

UNIVERSIDADE FEDERAL DE SERGIPE
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA
COMPUTAÇÃO

**Construção e Avaliação de uma Abordagem para Apoiar a
Compreensão e a Manutenção de Software baseada em
Mídias Dinâmicas**

Dissertação de Mestrado

Anne Caroline Melo Santos

São Cristóvão - Sergipe
2020

UNIVERSIDADE FEDERAL DE SERGIPE
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA
COMPUTAÇÃO

Anne Caroline Melo Santos

**Construção e Avaliação de uma Abordagem para Apoiar a
Compreensão e a Manutenção de Software baseada em
Mídias Dinâmicas**

Dissertação de mestrado
apresentada ao Programa de Pós-
Graduação em Ciência da
Computação (PROCC) da
Universidade Federal de Sergipe
(UFS) como parte do requisito para
obtenção do título de Mestre em
Ciência da Computação.

Orientador: Dr. Methanias Colaço Rodrigues Júnior

São Cristóvão - Sergipe
2020

**FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL
UNIVERSIDADE FEDERAL DE SERGIPE**

S237c Santos, Anne Caroline Melo
Construção e avaliação de uma abordagem para apoiar a compreensão e a manutenção de software baseada em mídias dinâmicas / Anne Caroline Melo Santos ; orientador Methanias Colaço Rodrigues Júnior. – São Cristóvão, SE, 2020.
129 f. : il.

Dissertação (mestrado em Ciências da Computação) –
Universidade Federal de Sergipe, 2020.

1. Sistemas multimídia. 2. Engenharia de software. 3. Software -
Manutenção. I. Rodrigues Júnior, Methanias Colaço, orient. II. Título.

CDU 004.41.01



UNIVERSIDADE FEDERAL DE SERGIPE
PRÓ-REITORIA DE PÓS-GRADUAÇÃO E PESQUISA
COORDENAÇÃO DE PÓS-GRADUAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Ata da Sessão Solene de Defesa da Dissertação do
Curso de Mestrado em Ciência da Computação-UFS.
Candidato: ANNE CAROLINE MELO SANTOS

Em 27 dias do mês de novembro do ano de dois mil e vinte, com início às 16h00min, realizou-se na Sala virtual <https://meet.google.com/kis-njhp-xaz>. A Sessão Pública de Defesa de Dissertação de Mestrado do candidato **Anne Caroline Melo Santos**, que desenvolveu o trabalho intitulado: "Construção e Avaliação de uma abordagem para apoiar o desenvolvimento e a manutenção de software baseada em recursos multimídia", sob a orientação do Prof. Dr. **Methanias Colaço Rodrigues Júnior**. A Sessão foi presidida pelo Prof. Dr. **Methanias Colaço Rodrigues Júnior** (PROCC/UFS), que após a apresentação da dissertação passou a palavra aos outros membros da Banca Examinadora, Prof. Dr. **Manoel Carvalho Marques Neto** (IFBA - Instituto Federal da Bahia) e, em seguida, ao Prof. Dr. **Michel dos Santos Soares** (PROCC/UFS). Após as discussões, a Banca Examinadora reuniu-se e considerou o mestrando (a) _____ aprovado _____ "(aprovado/reprovado)". Atendidas as exigências da Instrução Normativa 01/2017/PROCC, do Regimento Interno do PROCC (Resolução 67/2014/CONEPE), Resolução nº 25/2014/CONEPE e da Portaria nº 413 de 27 de maio de 2020 (Banca por videoconferência) que regulamentam a Apresentação e Defesa de Dissertação, e nada mais havendo a tratar, a Banca Examinadora elaborou esta Ata que será assinada pelos seus membros e pelo mestrando.

Cidade Universitária "Prof. José Aloísio de Campos", 27 de novembro de 2020.

Prof. Dr. Methanias Colaço Rodrigues Júnior
(PROCC/UFS)
Presidente

Prof. Dr. Michel dos Santos Soares
(PROCC/UFS)
Examinador Interno

DocuSigned by:

Prof. Dr. Manoel Carvalho Marques Neto
(IFBA)
Examinador Externo

Anne Caroline Melo Santos
Candidato

AGRADECIMENTOS

Primeiramente, quero agradecer a Deus e a nossa senhora pela graça de concluir mais uma etapa importante em minha vida, mesmo diante de tantas adversidades, termino esse ciclo com a sensação de dever cumprido e com o sentimento de gratidão.

Quero agradecer aos meus pais e a minha irmã por sempre acreditarem no meu potencial e me darem todo o apoio e suporte necessário para que eu continue crescendo e correndo atrás dos meus objetivos. Vocês são as peças mais importantes da minha vida, e sempre estarão presentes em todas as minhas conquistas.

A Rebeca pelo apoio e compreensão nos momentos em que precisava dar vacância as atividades do mestrado, e por sempre me ajudar a enxergar as situações de uma perspectiva otimista.

Um agradecimento especial ao meu professor e orientador Methanias, que despertou em mim o interesse em fazer ciência, e que durante esse período, me ensinou diariamente de forma gentil e objetiva, como eu poderia melhorar enquanto cientista. Quero agradecer também pela presteza e rapidez com a qual respondia as minhas dúvidas, e por ter feito dessa jornada mais leve com seu bom humor e positividade. Obrigada professor!

Agradecer também a 3Tecnos por ter aberto as portas para que pudesse aplicar o experimento. Agradecer aos voluntários por terem disponibilizado parte do seu tempo e participarem dessa empreitada. Sem vocês nada disso seria possível. Agradecer a João Paulo por ter me ajudado na condução do experimento, reunindo todas as condições necessárias para sua realização. Sua cooperação foi inestimável.

A Edna e a Rafael por terem sido meus parceiros ao longo dessa trajetória, sendo peças fundamentais para construção da abordagem proposta neste trabalho.

Ao PROCC, na figura da querida Elaine, pela disponibilidade em responder a todos meus questionamentos.

Por fim, a todos que indiretamente ou diretamente, estiveram envolvidos nesse processo e contribuíram de alguma forma para que chegássemos até aqui. O meu mais sincero agradecimento.

RESUMO

Contexto: A Engenharia de Software (ES) tem utilizado documentação textual para representar os requisitos do usuário. Para algumas pessoas, estas descrições podem não ser suficientes para entender o que precisa ser desenvolvido, sendo necessária a utilização de outros meios de visualização. **Objetivo:** Este trabalho objetivou propor e analisar uma abordagem multimídia, para apoiar a compreensão e manutenção de software, como alternativa às técnicas tradicionais de documentação de requisitos, avaliando se a utilização do *plug-in CodeMedia* aumenta a efetividade da compreensão e manutenção de software. **Método:** Inicialmente, foi realizado um mapeamento sistemático para identificar as abordagens que promoviam o uso de multimídia na Engenharia de Requisitos (ER), como suporte aos processos de compreensão e manutenção de software. Por fim, foi realizado um experimento controlado na indústria para avaliar a efetividade da abordagem proposta. **Resultados:** Para o estado da arte, foi identificada uma grande variedade de abordagens que promoviam o uso de multimídia na ER, dentre elas: TRECE, MURMER, *Wiki System Multimedia*, *Storytelling*, UTOPIA, bem como abordagens sem nomes explícitos. Com a execução do processo experimental, evidenciou-se que a abordagem multimídia apresentou os melhores resultados em termos de eficácia e nível de satisfação do cliente. No que se refere à média de tempo de codificação e ao nível de compreensão do código, a abordagem multimídia se mostrou menos eficiente. **Conclusão:** Após ser analisado o estado da arte, evidenciou-se que houve consenso favorável quanto ao uso de multimídia em ER. Os estudos selecionados demonstraram ser favoráveis à adoção de multimídia para persistir os requisitos do usuário. Com a avaliação experimental, foi constatado que a abordagem multimídia foi mais eficaz em termos de acertos na codificação e nível de satisfação do cliente com o produto final, tornando-a uma opção a ser considerada para Levantamento, Registro, Validação e Verificação dos requisitos de usuário.

Palavras-chave — Multimídia, Engenharia de Requisitos, Documentação de Software, Compreensão e Manutenção de Software e Engenharia de Software Experimental.

ABSTRACT

Context: Software Engineering (SE) has used textual documentation to represent the user requirements. For some people, textual descriptions may not be enough to understand what needs to be developed, requiring the use of other means of visualization. **Objective:** This work aimed to propose and analyze a multimedia approach, to support the comprehension and maintenance of software, as an alternative to the traditional requirements documentation techniques, evaluating whether the use of the CodeMedia plug-in increases the effectiveness of understanding and maintaining systems. **Method:** Initially, a systematic mapping was carried out to identify the approaches that promoted the use of multimedia resources in Requirements Engineering, as a support in the process of comprehension and maintaining software. Finally, a controlled experiment was carried out in the industry to assess the efficiency and effectiveness of the proposed multimedia approach. **Results:** For the state of the art, a wide variety of approaches were identified that promoted the use of multimedia in RE, among them: TRECE, MURMER, Wiki System Multimedia, Storytelling, UTOPIA, as well as approaches without explicit names. With the execution of the experimental process, it became evident that the multimedia approach presented the best results, especially in terms of effectiveness and level of customer satisfaction. Regarding the average coding time and the level of comprehension of the code, the multimedia approach proved to be less efficient. **Conclusion:** After analyzing the state of the art, it was evidenced that there was a favorable consensus regarding the use of multimedia in RE. The selected studies have shown to be favorable to the adoption of multimedia to persist software requirements. With the experimental evaluation, it was found that the multimedia approach was more effective in terms of correct coding and level of customer satisfaction with the final product, making it an option to be considered for Survey, Registration, Validation and Verification of user requirements.

Keywords - Multimedia, Requirements Engineering, Software Documentation, Software Comprehension and Maintenance and Experimental Software Engineering.

LISTA DE FIGURAS

| | |
|---|----|
| Fig 1. The Steps of a Systematic Mapping Process. | 31 |
| Fig 2. Results obtained from the search process..... | 37 |
| Fig 3. Distribution of primary studies by Evaluation Method. | 44 |
| Fig 4. Distribution of primary studies by Type of Publication..... | 45 |
| Fig 5. Distribution of articles by country. | 46 |
| Fig 6. Distribution of articles by publication year. | 47 |
| Fig 7. Distribution of articles by author. | 48 |
| Figure 1. Operating Architecture of the Multimedia Approach. | 75 |
| Figure 2. Menu for Adding Documentation. | 76 |
| Figure 3. Adding Files..... | 77 |
| Figure 4. File Selection Screen. | 78 |
| Figure 5. Link File to Requirement..... | 79 |
| Figure 6. Multimedia Documentation. | 79 |
| Figure 7. Reproduction and Visualization of Multimedia Resources..... | 80 |
| Figure 8. Folder Structure. | 81 |
| Figure 9. Age. | 88 |
| Figure 10. Gender. | 89 |
| Figure 11. Academic Training..... | 89 |
| Figure 12. Preferred Learning Mode. | 89 |
| Figure 13. Methods for Requirements Documentation..... | 90 |
| Figure 14. Profession/Position..... | 90 |
| Figure 15. Area of Operation..... | 91 |
| Figure 16. Years of experience in the language used in the experiment. | 91 |
| Figure 17. Number of Systems. | 92 |

LISTA DE TABELAS

| | |
|--|-----|
| Table 1. EMPIRICAL METHODS | 32 |
| Table 2. PICO MODEL | 32 |
| Table 3. RESEARCH QUESTIONS | 33 |
| Table 4. PICO MODEL CATEGORIES AND TERMS IDENTIFIED FOR BIBLIOGRAPHIC SEARCH | 34 |
| Table 5. SEARCH STRING..... | 34 |
| Table 6. SEARCH STRATEGY..... | 35 |
| Table 7. SELECTION CRITERIAS | 35 |
| Table 8. CLASSIFICATION..... | 36 |
| Table 9. APPROACHES..... | 38 |
| Table 10. IMPACTS | 42 |
| Table 11. DISTRIBUTION OF PRIMARY STUDIES BY IMPACT | 44 |
| Table 12. SCENARIOS | 46 |
| Table 13. OUTCOMES..... | 48 |
| Table 1. Software Maintenance Tasks. | 85 |
| Table 2. Time spent to execute tasks without the support of the multimedia approach. | 99 |
| Table 3. Time spent to execute tasks supported by the multimedia approach..... | 100 |
| Table 4. Average hits for each task implemented without the multimedia approach support..... | 102 |
| Table 5. Average hits for each task implemented with the multimedia approach support. | 104 |

LISTA DE ABREVIATURA E SIGLAS

| | |
|------|---|
| RE | Requirements Engineering |
| JBCS | Journal of the Brazilian Computer Society |
| SE | Software Engineering |
| PRS | Preferred Representation Systems |
| IRT | Item Response Theory |
| RS | Representation Systems |
| PICO | Population, Intervention, Control, and Outcomes (Results) |
| UML | Unified Modeling Language |
| OOP | Object-oriented programming |
| BPMS | Business Process Management System |
| JSON | JavaScript Object Notation |
| GQM | Goal, Question-Metric |
| TAM | Technology Acceptance Model |
| CDOC | Crypted Document |

SUMÁRIO

| | |
|--|-----------|
| 1. INTRODUÇÃO | 14 |
| 1.1. Problemática e Hipóteses | 15 |
| 1.2. Justificativa..... | 18 |
| 1.3. Objetivos da Pesquisa..... | 18 |
| 1.3.1. Objetivo Geral..... | 19 |
| 1.3.2. Objetivos Específicos | 19 |
| 1.4. Metodologia | 19 |
| 1.5. Organização da Dissertação | 20 |
| 2. DISCUSSÃO..... | 22 |
| 3. MAPEAMENTO SISTEMÁTICO | 27 |
| 3.1. Introduction..... | 29 |
| 3.2. Related works | 30 |
| 3.3. Methodology | 31 |
| 3.3.1. Research Secondary Questions and Terms Selection | 31 |
| 3.3.2. Search String and Selection Strategy | 34 |
| 3.3.3. Data Extraction and Analyses | 36 |
| 3.4. Results and Discussion | 37 |
| 3.4.1. Research Questions..... | 38 |
| 3.4.2. Approaches Overviews | 51 |
| 3.5. Syntheses and analysis, and lessons learned..... | 56 |
| 3.6. Threats do validity..... | 59 |
| 3.7. Conclusions | 59 |
| 3.8. References..... | 61 |
| 4. AVALIAÇÃO EXPERIMENTAL..... | 68 |
| 4.1. Introduction..... | 69 |
| 4.2. Methodology | 71 |
| 4.3. Related works | 72 |
| 4.4. CodeMedia Tool | 73 |
| 4.4.1. Architecture | 74 |
| 4.4.2. Features..... | 76 |
| 4.4.2.1. Attach Multimedia Resources to the Code..... | 76 |

| | | |
|----------|--|-----|
| 4.4.2.2. | Reproduction of Multimedia Resources..... | 80 |
| 4.4.2.3. | Storage and Maintenance of Multimedia Requirements | 80 |
| 4.4.2.4. | Folder Structure..... | 81 |
| 4.5. | Experimental Evaluation | 82 |
| 4.5.1. | Goal Definition | 82 |
| 4.5.2. | Planning | 83 |
| 4.5.2.1. | Context Selection..... | 83 |
| 4.5.2.2. | Hypothesis Formulation | 83 |
| 4.5.2.3. | Variables Selection..... | 85 |
| 4.5.2.4. | Description of Tasks in the Experiment..... | 85 |
| 4.5.2.5. | Selection of Participants and Objects..... | 87 |
| 4.5.2.6. | Experimental Design..... | 92 |
| 4.5.2.7. | Instrumentation | 94 |
| 4.6. | Experiment Steps..... | 96 |
| 4.6.1. | Preparation | 96 |
| 4.6.2. | Execution..... | 97 |
| 4.6.2.1. | Data Collection..... | 97 |
| 4.6.3. | Data Validation | 98 |
| 4.7. | Results and Discussion | 98 |
| 4.8. | Threats to Validity..... | 108 |
| 4.8.1. | Threats to Construction Validity..... | 108 |
| 4.8.2. | Threats to Internal Validity..... | 109 |
| 4.8.3. | Threats to External Validity..... | 110 |
| 4.9. | Conclusion and Future Works..... | 110 |
| | References..... | 112 |
| 5. | CONCLUSÃO | 116 |
| 5.1. | Resultados e Contribuições | 117 |
| 5.2. | Trabalhos Futuros | 119 |
| | REFERÊNCIAS..... | 120 |
| | APÊNDICES | 123 |
| | APÊNDICE A - Formulário de Caracterização da Amostra | 123 |
| | APÊNDICE B – Pergunta de Avaliação do Nível de Compreensão do Código | 126 |
| | APÊNDICE C - Questionário de Avaliação de Nível de Satisfação do Cliente. | 127 |

APÊNDICE D – Questionário de Avaliação de Nível de Compreensão de Software

129

1. INTRODUÇÃO

A Engenharia de Requisitos (ER) é uma atividade fundamental no processo de desenvolvimento de software. Por meio desta atividade, é possível organizar a base teórica para a construção de qualquer sistema, oferecendo suporte às fases iniciais do seu ciclo de vida (CALAZANS & MARIANO, 2016). Um dos papéis da ER é a identificação das necessidades e requisitos do produto de software, o qual depende do atendimento a estes e da satisfação das necessidades do usuário para obter sucesso (MENTEN, SCHEIBMAYR & KLIMPKE, 2010).

Tradicionalmente, a ER, assim como outras atividades da Engenharia de Software (ES), têm usado recursos textuais para catalogar e registrar os artefatos produzidos durante o desenvolvimento de software, apesar de existirem outros recursos para enriquecer a documentação, como imagens, áudio e vídeo (BRUNI *et al.*, 2012). A resistência em adotar outros meios para armazenar e apresentar a documentação de software acaba sendo prejudicial para a construção de sistemas, dado que, para algumas pessoas, os recursos textuais podem não ser suficientes para entender o que precisa ser desenvolvido, prejudicando o entendimento, acompanhamento e, conseqüentemente, a execução das tarefas. As pesquisas baseadas em psicologia embasam essa tese, ao afirmar que indivíduos, em contextos específicos, podem ter canais preferenciais para os processos de compreensão e aprendizagem (COLAÇO JÚNIOR *et al.*, 2017).

Dado o exposto, é necessário levar em consideração, na hora da elaboração da documentação de software, que as pessoas possuem diferentes formas de percepção da informação. Isso faz toda a diferença na comunicação e aprendizagem, visto que, a depender das preferências cognitivas, algumas pessoas podem compreender melhor por meio da gravação de áudio ou vídeo, do que por meio da leitura de um documento, e vice-versa (CHEN *et al.*, 2020). Desta forma, o uso de multimídia pode ser uma forma de oferecer às pessoas diferentes canais cognitivos para observar e interpretar a documentação, atuando como um catalisador para a interação rápida das partes interessadas e entendimento do que precisa ser desenvolvido (GARTNER & SCHNEIDER, 2012). Em outras palavras, a multimídia pode preencher a lacuna entre as origens, a terminologia e a educação das partes interessadas, como também, proporcionar percepções mais próximas da realidade (CREIGHTON, OTT & BRUEGGE, 2006).

Neste trabalho, multimídia pode ser entendida como uma forma eficaz de expressar as informações, usando a tecnologia digital para lidar com o texto, voz e imagem (CHANGBAI, 1998); um tipo de campo relevante para a integração de texto, gráficos, imagens estáticas ou em movimento, som de animação e outras mídias (FLUCKIGER, 1997); uma combinação de uma ampla gama de recursos audiovisuais como meios de comunicação.

Em (KARRAS, KIESLING & SCHNEIDER, 2016), foi verificado que quando os requisitos foram persistidos em formato multimídia, ao invés de notações textuais, houve um aumento de 80 pontos percentuais na qualidade dos requisitos. Embora as cenas gravadas em vídeo possam proteger o viés do analista, que realiza a gravação e a edição de vídeo, estas ainda são mais objetivas do que a dependência de atas escritas ou mesmo da memória pessoal do analista (HAUMER *et al.*, 2000). Em resumo, independentemente da quantidade de canais cognitivos ou da ordem utilizada para estimular estes canais, a possibilidade de apresentar os requisitos de usuário por meio de vários meios de comunicação possibilita um incremento ao processo de compreensão, permitindo que os desenvolvedores optem pela ordem preferida para se comunicar e aprender (COLAÇO JÚNIOR *et al.*, 2017).

Por todos esses aspectos, o objetivo deste trabalho é apresentar o resultado da construção e experimentação de uma abordagem multimídia acoplada ao código fonte, para apoiar a compreensão e manutenção de software, como alternativa às técnicas tradicionais de documentação de requisitos. O escopo da abordagem foi construído tomando como referência as lacunas das ferramentas identificadas durante a realização de um mapeamento sistemático, cujo objetivo foi identificar e caracterizar as abordagens que promoviam o uso de multimídia na ER.

Para automatizar a abordagem, foi desenvolvido um *plug-in* denominado *CodeMedia*, integrado ao *VisualStudio*, que permite a vinculação direta de trechos de código a mídias dinâmicas (áudio e vídeo) capturadas durante o desenvolvimento do software, especialmente durante a atividade de ER. Este *plug-in* foi avaliado em um processo experimental, do ponto de vista de eficiência e eficácia, seguindo as diretrizes de (BASILI, 1996), (SANTOS, COLAÇO JÚNIOR & SOUZA, 2018) e (OLIVEIRA & COLAÇO JÚNIOR, 2018).

Na próxima seção, será introduzida a problemática e hipóteses relacionadas à pesquisa em questão.

1.1. Problemática e Hipóteses

Os artefatos tradicionais da ER (documento de requisitos, casos de uso, atas ou outros registros das reuniões, diagramas UML de vários tipos, etc.) têm sido utilizados como prática comum para Levantamento, Registro, Validação e Verificação dos requisitos, funcionando como direcionamento durante o ciclo de desenvolvimento e manutenção de software.

Estes artefatos podem não fornecer resultados concretos podendo prover requisitos incorretos, pouco claros, ambíguos e não verificáveis (BRILL, SCHNEIDER & KNAUSS, 2010). Essa lacuna de comunicação, associada à dificuldade de compreensão apresentada por alguns profissionais na interpretação da documentação textual, prejudica o entendimento do que está sendo desenvolvido (COLAÇO JÚNIOR *et al.*, 2017), e ameaça a usabilidade da meta de produto de software, consistindo na produtividade de subobjetivos (em termos de eficiência), eficácia e satisfação (KARRAS *et al.*, 2017).

Para lidar com estas limitações, alternativas vêm sendo propostas. Dentre estas, foi averiguada a existência de abordagens que promovem o uso de multimídia na ER, para apoiar a compreensão e manutenção de software.

Diante deste cenário, este trabalho propõe o desenvolvimento e experimentação de uma abordagem para aumentar a eficácia e a eficiência da compreensão e manutenção de software. Será feita uma avaliação da compreensão e manutenção de software, perfazendo experimentos controlados feitos na indústria, para analisar a efetividade de uma ferramenta que encapsula e automatiza parte da abordagem, o *plug-in CodeMedia*. Este *plug-in* mesclará as características citadas, provendo a visualização da documentação do software por meio de mídias dinâmicas (áudio e vídeo) acopladas ao código-fonte.

Tendo em vista os fatos mencionados, foram elaboradas as seguintes questões principais de pesquisa:

- **RQ1** - A utilização da abordagem multimídia, automatizada pelo uso do *plug-in CodeMedia*, pode reduzir o tempo de codificação dos programadores no processo de manutenção de software?
- **RQ2** - A utilização da abordagem multimídia, automatizada pelo uso do *plug-in CodeMedia*, pode reduzir erros na codificação para manutenção de software?
- **RQ3** - A utilização da abordagem multimídia, automatizada pelo uso do *plug-in CodeMedia*, pode aumentar o nível de compreensão de software pelos programadores no processo de manutenção de software?

- **RQ4** - A utilização da abordagem multimídia, automatizada pelo uso do *plug-in CodeMedia*, pode aumentar o nível de satisfação do cliente com a solução desenvolvida, do ponto de vista de usabilidade da tarefa entregue?

A fim de avaliar tais questões, foram utilizadas as seguintes métricas: média de tempo de codificação das demandas; média de acertos para cada demanda implementada; nível de compreensão de software; e nível de satisfação do cliente.

Dessa forma, para a primeira questão de pesquisa elencada, a hipótese que foi testada e sua respectiva hipótese alternativa foram:

Hipótese 1

- Hipótese nula H0: A codificação para a manutenção de software, com e sem a utilização da abordagem multimídia, têm a mesma eficiência.

- H0: $\mu(\text{tempoCodificaçãoComAbordagemMultimídia}) = \mu(\text{tempoCodificaçãoSemAbordagemMultimídia})$.

- Hipótese alternativa H1: A codificação para manutenção de software, com a utilização da abordagem multimídia, é mais eficiente do que a codificação realizada sem a utilização da abordagem.

- H1: $\mu(\text{tempoCodificaçãoComAbordagemMultimídia}) < \mu(\text{tempoCodificaçãoSemAbordagemMultimídia})$.

Para a segunda questão de pesquisa elencada, a hipótese que foi testada e sua respectiva hipótese alternativa foram:

Hipótese 2

- Hipótese nula H0: A codificação para manutenção de software, com e sem a utilização da abordagem multimídia, têm a mesma eficácia.

- $\mu(\text{médiaAcertosCodificaçãoComAbordagemMultimídia}) = \mu(\text{médiaAcertosCodificaçãoSemAbordagemMultimídia})$.

- Hipótese alternativa H1: A codificação para manutenção de software, com utilização da abordagem multimídia, é mais eficaz do que a codificação realizada sem a utilização da abordagem.

- $\mu(\text{médiaAcertosCodificaçãoComAbordagemMultimídia}) > \mu(\text{médiaAcertosCodificaçãoSemAbordagemMultimídia})$.

Para a terceira questão de pesquisa elencada, a hipótese que foi testada e sua respectiva hipótese alternativa foram:

Hipótese 3

- Hipótese nula H0: O nível de compreensão de software dos desenvolvedores e a utilização da abordagem multimídia não possuem uma correlação.

- $r = 0$

- Hipótese alternativa H1: O nível de compreensão de software dos desenvolvedores e a utilização da abordagem multimídia possuem uma correlação.

- $r \neq 0$

Para a quarta questão de pesquisa elencada, a hipótese que foi testada e sua respectiva hipótese alternativa foram:

Hipótese 4

- Hipótese nula H0: O nível de satisfação do cliente com a solução entregue não está correlacionado com a utilização da abordagem multimídia.

- $r = 0$

- Hipótese alternativa H1: O nível de satisfação do cliente com a solução entregue está correlacionado com a utilização da abordagem multimídia.

- $r \neq 0$

É importante frisar que as hipóteses nulas (H0) são as hipóteses que pretendia-se refutar e as hipóteses alternativas (H1) são aquelas que, dentro do contexto do experimento, não seriam rejeitadas.

1.2. Justificativa

As abordagens existentes para Levantamento, Registro, Validação e Verificação dos requisitos de usuário têm se mostrado insuficientes para registrar os requisitos de forma completa, consistente e correta. Estudos realizados mostraram que 40% dos defeitos identificados em projetos de software são oriundos do registro incorreto dos requisitos. Abordagens vêm sendo propostas para lidar com as questões mencionadas, incluindo técnicas baseadas em vídeo (BOULILA, HOFFMANN & HERRMANN, 2011).

Por conseguinte, faz-se necessário experimentar outras abordagens que permitam que os interessados no produto usem diferentes canais de aprendizagem e comunicação, possibilitando um melhor diálogo, divulgação, entendimento e apresentação das informações, dentro de um contexto de compreensão e manutenção de software.

1.3. Objetivos da Pesquisa

Para realização desta pesquisa, têm-se os seguintes objetivos geral e específicos.

1.3.1. Objetivo Geral

Este trabalho objetivou propor e analisar uma abordagem multimídia, para apoiar a compreensão e manutenção de software, como alternativa às técnicas tradicionais de documentação de requisitos, avaliando se a utilização do *plug-in CodeMedia*, criado para viabilizar a abordagem, aumenta a efetividade da compreensão e manutenção de sistemas.

1.3.2. Objetivos Específicos

Para possibilitar a realização do objetivo geral, podemos enumerar os seguintes objetivos específicos:

- Mapeamento Sistemático para identificar e caracterizar o conjunto de estudos primários publicados, os quais abordam o uso de multimídia na Engenharia de Requisitos (ER), como suporte para a compreensão e manutenção de software;
- Requisitos da abordagem multimídia, baseados em resultados e lacunas encontrados na literatura;
- *Plug-in CodeMedia*, para visualização da documentação do software por meio de mídias dinâmicas (áudio e vídeo), acopladas diretamente ao código-fonte;
- Experimento controlado *in vivo*, para avaliar o uso do *CodeMedia* como facilitador no processo de compreensão e manutenção de software.

1.4. Metodologia

Este trabalho é um estudo experimental que avaliou o desempenho de uma Abordagem Multimídia para Levantamento, Registro, Validação e Verificação dos requisitos de usuário, usando mídias dinâmicas (áudio e vídeo) integradas ao código-fonte. A fim de avaliar tais questões, foram utilizadas as seguintes métricas: (i) média de tempo de codificação das demandas, (ii) média de acertos para cada demanda implementada, (iii) nível de compreensão de código e (iv) nível de satisfação do cliente.

Além do ponto de vista experimental, este trabalho também se caracteriza como exploratório, uma vez que, inicialmente, foi realizado um mapeamento sistemático da literatura, publicado em (SANTOS, M. C. A; COLAÇO JÚNIOR, M; ANDRADE, C. E, 2020), com o

objetivo de encontrar pesquisas sobre o uso de multimídia no processo de desenvolvimento e manutenção de software, em especial na Engenharia de Requisitos (ER). A análise desses estudos orientou a construção da abordagem multimídia proposta neste trabalho, bem como demonstrou a ausência da aplicação de uma metodologia rigorosa e experimental nas avaliações publicadas. Afastando-se desta lacuna, nosso experimento consistiu no planejamento, instrumentação, parceria com a indústria e seleção de participantes, preparação do ambiente, execução, coleta de dados e validação estatística dos resultados.

Na execução do experimento, os participantes foram submetidos a dois tipos de tratamento: codificação de tarefas de manutenção reais com e sem o suporte da abordagem multimídia. O design experimental pode ser visualizado na seção 4.5.2.6. Por conseguinte, para auxiliar os cálculos e verificar se havia diferenças significativas na eficácia e eficiência dos tratamentos, foi utilizada análise de correlação e três testes estatísticos: *Shapiro-Wilk*, *Wilcoxon* e Teste T.

Em resumo, este trabalho conduziu um experimento, o qual tem o seu método descrito de forma autocontida, no seu planejamento, detalhado no capítulo 4, com 4 etapas macro: planejamento com seleção de recursos e treinamento; operação com execução do experimento e entrevistas, coleta de dados e validação; comparação dos tratamentos baseada na significância estatística; e análise dos resultados.

1.5. Organização da Dissertação

Este documento está organizado de acordo com a Instrução Normativa Nº 05/2019/PROCC, a qual permite que a Dissertação seja “uma compilação de artigos científicos submetidos ou publicados em veículos com *Qualis*, desde que seja contextualizada com seções de Introdução, Discussão, Conclusão e Referências, não limitada a estas”. São 5 capítulos que fornecem uma base conceitual e experimental para o entendimento sistêmico. Os tópicos a seguir descrevem o conteúdo de cada um dos capítulos:

- O Capítulo 1 apresenta esta Introdução, explicando as justificativas, juntamente com as hipóteses levantadas;
- O Capítulo 2 traz uma síntese narrativa do Mapeamento Sistemático, juntamente com uma discussão do experimento;
- O Capítulo 3 replica um Mapeamento Sistemático aceito e publicado no *Journal of Software: Evolution and Process*;

- O Capítulo 4 apresenta um artigo submetido ao *Journal of the Brazilian Computer Society (JBCS)*, resumindo todo o trabalho efetuado nesta pesquisa. São descritos o Planejamento, Operação e Resultados do Experimento;
- Finalmente, no capítulo 5, é apresentada uma compilação de conclusões, contribuições e sugestões de trabalhos futuros.

2. DISCUSSÃO

Neste capítulo, será apresentada uma discussão dos resultados obtidos durante a realização do Mapeamento Sistemático e da Avaliação Experimental.

Com relação ao Mapeamento Sistemático, foi verificado que a utilização de multimídia na ER é considerada desde 1992. Isso indica que essa linha de pesquisa já vem sendo estudada há algum tempo. Nesse período, diversas soluções têm sido propostas para incorporar mídias dinâmicas ao processo de solicitação e manutenção dos requisitos, porém, o número de estudos publicados ainda é baixo. Isso pode indicar duas coisas: (1) que a área já produziu resultados definitivos; ou (2) que a área está falhando em atingir seus objetivos. O segundo caso é o mais provável, uma vez que multimídia e evolução de software são tópicos muito importantes na moderna engenharia de software; e multimídia para ER é uma área de pesquisa ampla e relativamente jovem, que tem muito a oferecer à comunidade de pesquisa de Engenharia de Software.

Os resultados também mostraram que as publicações referentes ao uso de mídias dinâmicas na ER envolvem profissionais de diferentes nacionalidades e são desenvolvidas em diferentes países. Isso denota que a busca por melhorias na documentação de requisitos e na comunicação entre os stakeholders, na construção e manutenção de software, apresenta-se como uma preocupação de diferentes partes do mundo.

No que concerne aos métodos de pesquisa utilizados para avaliação, estudo exploratório (31,92%) e experimento controlado (27,66%) aparecem com as principais opções entre os trabalhos selecionados. Mesmo sendo utilizado em 27.66% dos trabalhos, poucas abordagens encontradas validaram suas soluções por meio de experimentos controlados, mostrando a necessidade de aumentar o uso do método científico nesta área, com replicações de estudos que permitirão avaliar se outros pesquisadores, de forma independente, surgirão com os mesmos resultados. Mesmo aqueles que validaram, realizaram uma validação parcial.

Qualquer abordagem deve focar no cliente final e validar sua utilidade em experimentos ou estudos de caso bem executados. Isso está longe de propor uma associação de multimídia aos requisitos e de executar alguns estudos de viabilidade sobre isso. Pesquisadores e profissionais devem se concentrar em responder a perguntas como: Minha abordagem se adapta a softwares maiores do mundo real? Como meus resultados se generalizam para outros clientes, domínios e sistemas? Caso contrário, sempre haverá um problema com a validade externa da abordagem proposta e será difícil passar do estado da arte para o estado da prática

em engenharia de software. Ainda neste contexto, a adoção efetiva de multimídia em ambientes industriais de engenharia de software é muito baixa.

Na verdade, há pouco trabalho colaborativo na área. A maior parte dos trabalhos analisados busca desenvolver novas abordagens, com poucos trabalhos de validação das soluções já existentes. A validação e as atividades cooperativas levariam a uma melhoria mais rápida das abordagens existentes e a um entendimento mais profundo da área. Os pesquisadores nunca compararam profundamente seus resultados com outras abordagens.

Nessa linha de generalização de resultados, a maioria dos estudos cita o entendimento compartilhado e a qualidade das especificações como os principais impactos da incorporação da multimídia na ER. Esta descoberta mostra que os requisitos multimídia podem ajudar a reduzir problemas comuns encontrados em ER, tais como requisitos ambíguos, não claros e não verificados.

Desta forma, uma variedade de soluções para registrar, preservar, vincular e revisar requisitos usando multimídia foi usada e testada, como alternativa às técnicas tradicionais utilizadas na ER. No que diz respeito aos ambientes nos quais mídias dinâmicas podem ser utilizadas como facilitadoras do desenvolvimento e manutenção de software, o cenário é bastante favorável, sendo explorada em diferentes contextos. Essa abordagem, por exemplo, tem sido utilizada no desenvolvimento de softwares complexos, que representam riscos elevados (por exemplo, aeroportos, saúde e sistemas financeiros). Todos esses fatores enfatizam a viabilidade da abordagem proposta neste trabalho e mostram que o uso de mídias dinâmicas pode melhorar o processo de compreensão do código, diminuindo custos de evolução e manutenção.

Em relação às lacunas e à análise da evolução do software, a integração com o código-fonte e a fase de construção (programação) precisam ser aprimoradas. A falta de integração estreita com o código impede que ferramentas multimídia possam ser usadas com eficácia para analisar e compreender os dados produzidos durante a evolução do software. Além de gravar ou filmar as entrevistas com clientes e fornecê-las ao programador, para que possam entender melhor seus requisitos, um programador deve ser capaz de clicar em um link no código-fonte e ver ou ouvir as entrevistas das partes interessadas, bem como as explicações do código gravadas por um colega de trabalho (COLAÇO JÚNIOR *et al.*, 2017). Neste ponto, as explicações podem incluir incrementos e evoluções.

Outra característica a ser aprimorada e considerada pelas ferramentas multimídia são as estratégias de análise da evolução. As estratégias temporais, por exemplo, podem retratar a evolução considerando todas as versões disponíveis para análise. Dadas n versões v_1, v_2, \dots ,

vn, (ou um subconjunto sequencial considerável destas), este tipo de análise leva em consideração tudo o que aconteceu da versão v1 à versão vn. Isso ajudaria, por exemplo, a analisar mudanças entre diferentes artefatos e mudanças de negócios ao longo do ciclo de evolução do software. Em outras palavras, o versionamento do conteúdo multimídia deve estar alinhado ao versionamento do software, permitindo que a filmagem ou explicação de um requisito do negócio, por exemplo, seja associada à evolução das métricas do código. Em outra dimensão, uma questão a ser avaliada seria o registro ou filmagem de explicações sobre a evolução de cada versão do software, permitindo que as principais mudanças e decisões-chave sejam documentadas.

Em relação à manutenção de software, os estudos não abordam especificadamente ou profundamente o uso de multimídia aplicada à manutenção e compreensão de software. Em nossa pesquisa, foi proposto e implementando um acoplamento mais próximo ao código, explicações técnicas do código e novas formas de encontrar trechos específicos de áudios e vídeos.

Vale ressaltar que as mídias digitais podem dar suporte a diversas atividades de ER, por meio da gravação de entrevistas, reuniões com stakeholders, histórias de usuários e discussões com clientes. A gestão desse conteúdo pode ser feita por meio de iniciativas próprias, ou por meio de ferramentas disponíveis no mercado.

Encerrando a análise do estado da arte, o uso de multimídia mais intimamente ligada ao código e às estratégias de análise da evolução podem ajudar a responder a duas perguntas importantes (SANTOS, COLAÇO JÚNIOR & ANDRADE, 2020): (1) Onde vejo meu negócio no código? (2) Quais pontos evoluíram no código em linha com a evolução do meu negócio? As respostas a essas perguntas podem ajudar a aumentar a precisão da localização do código a ser evoluído e alterado, aumentando a eficácia da manutenção, reduzindo custos e proporcionando a perspectiva de impacto no código-fonte, com base na evolução do negócio.

Do ponto de vista experimental, analisando os resultados do experimento executado nesta pesquisa, constatou-se que quando a abordagem multimídia foi utilizada, o tempo de codificação foi um pouco maior em relação ao tempo sem a abordagem. Porém, por meio da aplicação do teste estatístico de *Wilcoxon*, constatou-se que não houve significância estatística para rejeitar a hipótese de igualdade entre os dois tratamentos. Assim, H_0 não é rejeitada. Portanto, a codificação para manutenção de software, com e sem o uso da abordagem multimídia, tem a mesma eficiência.

Dentre os fatores que podem ter contribuído para o aumento do tempo de utilização da abordagem, pode-se citar o estilo de programação apresentado por cada participante. Alguns

tinham um estilo mais direto, focando exatamente no que a tarefa exigia, o que reduzia significativamente o tempo de codificação. Enquanto outros realizaram uma análise mais profunda da tarefa, indo além do que a tarefa precisa. Isso acabou gerando refatoração de código e, conseqüentemente, aumentando o tempo de resolução da tarefa. Nesse ponto, vale destacar mais uma vantagem da utilização de multimídia na ES, ou seja, a explicação em áudio ou vídeo do código pode ter evidenciado um débito técnico, estimulando positivamente a refatoração. Por fim, outro ponto que pode ter influenciado no maior tempo despendido na resolução de tarefas, quando foi utilizado o *plugin CodeMedia*, foi o tempo necessário para reproduzir e compreender o conteúdo multimídia anexado ao código.

Para segunda questão de pesquisa elencada, foi possível verificar que, quando utilizada a abordagem multimídia, o resultado foi promissor, ou seja, a média de acertos por tarefa foi maior, quando comparada à média de acertos sem a utilização da abordagem. Porém, ao aplicar o teste estatístico de *Wilcoxon*, constatou-se que ainda não havia significância estatística para rejeitar a hipótese de igualdade entre os dois tratamentos. Ou seja, mesmo com a melhora na média de respostas corretas, para programadores com o perfil psicológico e com o nível de experiência dos avaliados, as eficácias com e sem a abordagem ainda precisam de mais replicações de experimentos como este para descobrir suas diferenças.

Para a terceira questão de pesquisa listada, que aborda a correlação entre a compreensão do código e o uso da ferramenta, houve um baixo coeficiente de correlação negativo, segundo as respostas ao questionário qualitativo, de -0,16. Ou seja, surpreendentemente, quando a ferramenta foi utilizada, o entendimento, segundo os programadores, caiu na proporção de apenas 16%. Porém, apesar de 32 amostras, ou seja, um número maior que 30, o que permite uma aproximação da distribuição da amostra por uma distribuição normal, o valor T calculado, com 30 graus de liberdade ($n-2$), ficou -0,88, acima do valor do T crítico, que, para um nível de significância de 0,05, é -1,96. Assim, não há significância estatística para rejeitar a hipótese nula (H_0) de que não há correlação entre a compreensão do código e o uso da ferramenta.

Dentre os fatores que podem ter influenciado neste resultado, pode-se citar: a familiaridade dos desenvolvedores com o código; a qualidade e assertividade do conteúdo multimídia vinculado às tarefas; falta de compreensão da questão de compreensão do código; e o fato de a maior parte do material multimídia anexado às tarefas persistir em formato de áudio. Conforme sugerido em (COLAÇO JÚNIOR, MENEZES, CORUMBA, MENDONÇA & SANTOS, 2017), os áudios com explicações do código podem ser interessantes, para permitir ao programador ter ajuda “*on the fly*” de seus colegas de trabalho, previamente gravada e

disponível no código, porém, os desenvolvedores podem ter ordem de preferência de sistemas de representação.

Nesse sentido, como pode ser observado na Figura 12, a maioria dos participantes não possui a primeira ou a segunda preferência auditiva. Isso pode ter impactado a compreensão do código e também pode ter um efeito transitivo na satisfação do cliente, uma vez que tarefas bem executadas dependem de um bom entendimento. Em outras experiências e também como lição profissional aprendida, o conteúdo multimídia deve ser criado levando-se em consideração as proporções das preferências da equipe, o que pode gerar resultados ainda melhores a favor da documentação multimídia. Um programador com preferências cinestésicas irá preferir um vídeo mostrando o funcionamento do sistema a cada linha de código executada e, se possível, uma cena usando um protótipo. Finalmente, a ferramenta pode ser mais útil para programadores com menos experiência na linguagem utilizada no experimento.

Para a quarta questão de pesquisa listada, que trata da correlação entre o uso da ferramenta e a satisfação do cliente, houve um coeficiente de correlação positivo, segundo as respostas ao questionário qualitativo, de 0,07. Ou seja, quando a ferramenta foi utilizada, a satisfação do cliente, segundo ele, aumentou 7%. O valor de T calculado, com 30 graus de liberdade ($n-2$), foi de 0,36, abaixo do valor do T crítico, que, para um nível de significância de 0,05, é de 1,96. Assim, embora o resultado seja promissor, ainda não foi possível rejeitar a hipótese nula de que não há correlação entre o uso da ferramenta e a satisfação do cliente. De qualquer forma, dependendo do contexto atual de acirrada disputa de mercado, aumentar a satisfação do cliente em 7% pode ser uma vantagem competitiva.

Os fatos apresentados ajudam a justificar e corroborar os resultados obtidos no experimento. Percebeu-se, por meio do questionário de caracterização da amostra, que a maioria dos participantes era experiente na linguagem avaliada e já havia mantido um número considerável de sistemas, conforme pode ser observado nas figuras 16 e 17, respectivamente. É provável que, se o experimento tivesse sido aplicado com programadores menos experientes, sem muita familiaridade com a linguagem utilizada, os resultados teriam sido diferentes e mais favoráveis à abordagem multimídia. Nesse contexto, é necessário aplicar o experimento com diferentes perfis de programador, para saber como a abordagem multimídia se comporta em diferentes cenários, e, assim, é possível fazer previsões mais gerais sobre o uso de conteúdo multimídia no processo de compreensão e manutenção de software.

Apresentada a seção de Discussões, no próximo capítulo será apresentado o Mapeamento Sistemático realizado para construção da base teórica deste trabalho.

3. MAPEAMENTO SISTEMÁTICO

Neste capítulo, será apresentado o Mapeamento Sistemático realizado para identificar e caracterizar o conjunto de estudos primários publicados, que abordam o uso de multimídia na ER, como suporte para a compreensão e manutenção de software.

Multimedia Resources as a Support for Requirements Engineering and Software Maintenance

Anne Caroline M. Santos
Federal University of Sergipe
Aracaju/Sergipe/Brazil
anne.santos@dcomp.ufs.br

Methanias Colaço Júnior
Federal University of Sergipe
Aracaju/Sergipe/Brazil
mjrse@hotmail.com

Edna de C. Andrade
Federal University of Sergipe
Aracaju/Sergipe/Brazil
ednacarvalhosempre@gmail.com

Abstract—Context: Textual documentations are frequently used in the software development process to outline features and behaviors of an application. For some people, textual descriptions may not be enough to understand what is being developed. In this scenario, multimedia resources appear as an option for software documentation, providing other ways to observe and interpret information. **Objective:** To identify and characterize the approaches which promote the use of multimedia in Requirements Engineering (RE) to support software development and maintenance. **Method:** A systematic mapping was conducted to find the primary studies in the literature and collect evidence for directing future research. **Results:** Only 27.66% of the approaches found validated their solutions through controlled experiments, showing the need to increase the use of scientific method in this area, with replications of studies that will allow to evaluating if other researchers independently will come up with the same results. In this context, the approaches/techniques identified were TRECE, MURMER, Wiki System Multimedia, Storytelling, Virtual World Environment, VisionCatcher, PRESTO4U, ReqVidA, CrowdRE, AVW, The Software Cinema Technique, Dolli Project, UTOPIA, and approaches without explicit names, which, as a rule, use multimedia resources as an additional support. **Conclusions:** There was a favorable consensus regarding the use of multimedia in RE. The selected studies demonstrated to be favorable to the adoption of media to persist and store

the requirements of a system. Moreover, multimedia resources can improve the process of understanding the code and decrease evolution and maintenance costs.

General Terms

Design, Documentation, Experimentation, Human Factors

Multimedia, Reliability, Software Engineering, Verification and Security.

Keywords—Multimedia Resources, Requirements Engineering, Software Documentation, Software Evolution and Software Comprehension.

3.1. Introduction

Understanding the requirements of a software is among the most difficult tasks faced by a software engineer [1]. To ensure that the user requirements are always complete, consistent, relevant and up-to-date, able to properly support professionals during and after the software development cycle, it is necessary to use systematic and repeatable techniques [2].

Historically, requirements engineering (RE) has always used a lot of structured and unstructured text (use cases, diagrams, and prototypes) to describe the requirements of a system [3]. According to [4], the way software engineers process these requirements affects the success of this processing for both text and diagrams, impacting understanding of what should be developed.

For some people, textual descriptions may not be sufficient to understand what is being developed. There are groups that can better understand through audio or video recording, along with the textual description or individually [5].

The ability to present software requirements across multiple communication channels allows for an incremental understanding process, empowering developers to choose their preferred means of communicating and learning. In this scenario, multimedia resources can be exploited to improve understanding and descriptions of requirements in the software development process. The combination of textual notes and video can offer greater potential for obtaining and crafting more readable, accurate and objective requirements that represent user needs [5].

Regarding Software Evolution, whereas, for some systems, the cost devoted to evolution and maintenance now accounts for more than 90% of total software costs [6] and software maintainers spend approximately 50% of their time in the process of understanding the code [7], multimedia tools integrated into the code can help people to understand software through the use of explanatory audio and video, and it can be effectively used to analyze and understand the data produced during software evolution. Generally, these tools do not have a closely integration with the code and allow analyzing indirectly the evolution of the software with respect to a set of software maintenance related questions [8][9][10]. For this reason, some

researchers have been proposing associate audio or video explanations to software architecture, design of code artifacts and source code [5], evidencing an aspect to be observed by the multimedia-based requirements tools.

This article presents a systematic mapping aimed to identify and characterize the approaches and techniques that promote the use of multimedia resources in RE to support software development and maintenance.

After answering the research questions, it was identified approaches such as TRECE, MURMER, Wiki System Multimedia, Storytelling, Virtual World Environment, VisionCatcher, PRESTO4U, ReqVidA, CrowdRE, AVW, The Software Cinema Technique, Dolli Project, UTOPIA, and approaches without explicit names, which, as a rule, use multimedia resources as an additional support. In the context of the types of study, "Exploratory Study", with 15 (31.92%) publications, far outperformed the other analyzed types. Regarding countries, Germany (17), United States (15), and Canada (7) lead the ranking of publications on the subject. Among the main channels, conferences stood out with 33 (70.22%), while periodicals reached 14 (29.78%).

The remainder of this paper is structured as follows: section 2 presents the related works. Section 3 describes how systematic mapping was planned. In section 4, the results obtained during the study is showed. Section 5 shows the discussion about the results. Section 6 presents threats to validity. Finally, in section 7, the conclusion is presented.

3.2. Related works

Some articles served as reference for the construction of the research line approached in this work. [5] took the first step towards the study of Preferred Representation Systems (PRS) in RE. They used survey combined with an analysis of Item Response Theory (IRT) data to show that software engineers have preferred representation systems and point out that these preferences can be identified. These findings spurred research on multimedia (video and audio) capabilities to help build and understand software, and to support scientists and managers create better ways to communicate and manage software engineers.

[11] describes that the way that software engineers process resources like diagrams and non-conventional visualization metaphors impacts on the success of that processing for both, text and diagrams. The article points out that other studies do not evaluate what types of representational systems (RS) are the preferred by software engineers. This discovery is

important to define what kind of resources would be useful to the software engineer, in a specific context.

As these papers point out, there is improvement in software understanding when developers can choose different ways of interacting with project information. This highlights a gap and the need to compile all the related works, identifying the current state of the area and what may be done.

3.3. Methodology

Some researchers have been working to provide stable methods for systematic literature review process. In this study, we explore the approach defined by [12][13], in order to define the procedure and the methodology to be used.

In this context, it was conducted a mapping study because it allows to analyzing the primary studies and answering research questions in a broader way, collecting evidence for directing future research. In addition, both data extraction and analysis are largely concerned with classification of the available studies. The steps performed in this mapping are detailed in Fig 1.

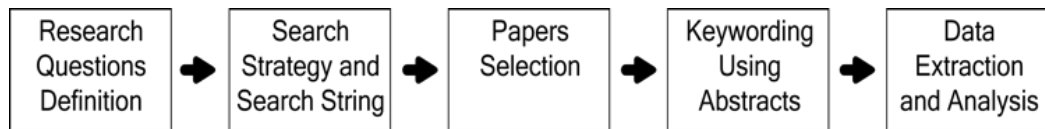


Fig 1. The Steps of a Systematic Mapping Process.

Considering these steps, the goal of this study was to identify and characterize the set of primary studies published that address the use of multimedia resources in RE. It is presented the main research question of this study:

“What is the state of the art in the use of multimedia resources in Requirements Engineering to support software development and maintenance? ”

The definition of the research secondary questions, search strategy and selection criteria will be described in the following sections.

3.3.1. Research Secondary Questions and Terms Selection

The research questions were developed with the purpose of presenting an overview of the area, highlighting key aspects of primary studies.

For this study, the research questions attempt to provide a specific insight into the relevant aspects about the use of multimedia resource in RE. These includes questions about which approaches and techniques are used to promote the use of media in RE to support software development; impacts of the use of media in RE processes; scenarios where media is used for software development; the kind of forum where papers have been published (conferences and periodicals); and the type of research carried out (empirical method). The empirical method is classified in one of the following types [14][15], depending on the purpose of the evaluation and how researchers described their evaluated methods (see Table 1).

Table 1. EMPIRICAL METHODS

| Empirical Methods | |
|--------------------------|--|
| Survey | Empirical strategy to gather data from a population sample and to achieve an understanding of that population [14]. |
| Case study | Research methodology that studies a phenomenon in its natural context [15]. |
| Controlled experiment | Controlled experiment provides a formal, rigorous and controlled investigation in which an intervention is introduced to observe its effects [16]. |
| Feasibility Study | In this type of study, authors use the proposed approach over a software as a feasibility study and it only presents a proof of concept [17] |
| Exploratory Study | It focuses on exploring a specific context and does not intend to discuss final and conclusive solutions, but helps researchers to have a better understanding of the problem and generating ideas and hypotheses for new research [15]. |
| Comparison | The authors performed a comparison based on checklist among their approach and another related tool [18]. |

As a guarantee of quality, PICO model was used in order to prove research questions characterization and classification capacity. PICO, an acronym for Population, Intervention, Control, and Outcomes (Results), aims to highlight the effects of an intervention on a given population [19]. Table 2 illustrates the PICO model .

Table 2. PICO MODEL

| Category. | Description |
|------------------|--|
| Population | Publications by researchers considering Requirements Engineering approaches. |

| Category. | Description |
|--------------|--|
| Intervention | Context of approaches and techniques using multimedia resources to perform specific tasks such as requirements specification, business process modeling, and communication between stakeholders and developer team. |
| Control | <p>Traditional techniques (e.g., use cases, diagrams, and analysis document) used in elicitation and requirements management were compared with approaches that use multimedia to support software development and maintenance. However, no empirical comparison is made at this stage, approaches are identified and characterized.</p> <p>Control papers that obey the intervention:</p> <ul style="list-style-type: none"> • Using different communication media on group performance in requirements engineering [20]. • Automating requirements traceability: Beyond the record & replay paradigm [21]. • Videos vs. use cases: Can videos capture more requirements under time pressure? [22]. • Interactive multimedia storyboard for facilitating stakeholder interaction: supporting continuous improvement in IT-ecosystems[23]. |
| Results | Requirements Engineering providing management of high-quality requirements, better cognition, documentation, understanding, and choice of personal preferences for Software Engineers. |

Table 3 describes the research questions of this work.

Table 3. RESEARCH QUESTIONS

| ID. | Research Question |
|-----|--|
| RQ1 | Which are approaches used to capture and record user requirements using multimedia resources? |
| RQ2 | What is the impact of using media in software development and maintenance? |
| RQ3 | What research methods have been used in research into the media use in requirements engineering? |
| RQ4 | What types of publications or forums have dealt with the issue of media in requirements engineering? |

| ID. | Research Question |
|-----|---|
| RQ5 | In which scenarios media are used for software modeling, development and maintenance? |
| RQ6 | What are the countries that have more researchers who published on this field? |
| RQ7 | Which years have had the most publications in this area? |
| RQ8 | Who are the researchers that have been publishing in this field? |

Table 4 shows the PICO-based terms that were selected before they were refined (see next section).

Table 4. PICO MODEL CATEGORIES AND TERMS IDENTIFIED FOR BIBLIOGRAPHIC SEARCH

| Category. | Description |
|--------------|--|
| Population | Software Engineering, Requirements Engineering, Requirements Analyses, and Requirements Elicitation. |
| Intervention | Multimedia, Audio and Video. |
| Control | No strings |
| Results | Software documentation, software comprehension. |

3.3.2. Search String and Selection Strategy

To perform the search in the databases, a search string was created using English terms and several synonyms, associated with the studies that are contained in the areas of computing that deal with the use of multimedia (audio and video) to support software development and maintenance. These terms (see table 4) were identified with the help of PICO model and control papers, described in Table 2. After refinement, the adjusted terms were used to construct the final search string presented in Table 5.

Table 5. SEARCH STRING

| Database | Search String |
|-------------|--|
| IEEE Xplore | ("Document Title":"multimedia" OR "Document Title":"audio" OR "Document Title":"video") AND ("Publication Title":"requirements engineering" OR "Document Title":"requirements elicitation" OR |

| | |
|----------------|---|
| | "Document Title": "requirements analysis" OR "Document Title": "software requirements" OR "Document Title": "software documentation" OR "Document Title": "software design" OR "Document Title": "software engineering" OR "Document Title": "software comprehension" OR "Document Title": "software maintenance" OR "Document Title": "software construction") |
| Scopus | TITLE("Multimedia") OR TITLE("Audio") OR TITLE("video") AND TITLE("requirements elicitation" OR "requirements engineering" OR "software documentation" OR "software construction" OR "requirements analyses" OR "software comprehension" OR "software requirements" OR "software maintenance" OR "software engineering" OR "software design") |
| Science Direct | ((("Multimedia" OR "Audio" OR "video") AND ("requirements elicitation" OR "requirements engineering" OR "software documentation" OR "software construction" OR "requirements analyses" OR "software design" OR "software comprehension" OR "software maintenance" OR "software engineering" OR "software requirements")) |
| Web of Science | (TI=((requirements elicitation OR requirements engineering OR requirements analysis OR software design OR software comprehension OR software maintenance OR software construction OR software engineering OR software requirements OR software documentation) AND multimedia)) AND IDIOMA: (English) AND TIPOS DE DOCUMENTO: (Article) |

In table 6, the search strategy is presented.

Table 6. SEARCH STRATEGY

| | |
|-----------------------|---|
| Databases | Scopus; IEEEExplore; Science Direct and Web of Science. |
| Academic Publications | Journal Papers; Conference Paper. |
| Search applied to | Title; Abstract; Keywords. |
| Language | Papers written in English |
| Publication period | No period has been defined |

In order to filter relevant articles for this systematic mapping, the inclusion and exclusion criteria were defined. The selection criteria are presented in table 7.

Table 7. SELECTION CRITERIAS

| |
|-------------------------------------|
| Inclusion/Exclusion Criteria |
|-------------------------------------|

| | |
|--|--|
| Inclusion Criteria | Paper written in English; Academic journal and conference; The Articles must specify approaches that propose the use of multimedia resources to support RE processes OR describe the impact to use media to software development OR cite the scenarios where multimedia can be act as facilitator to software development; The papers must be available for online consultation; The articles must contain the theme of this study in the title, abstract or keywords. |
| Exclusion Criteria for titles and abstract | Duplicated articles; Unavailable articles; Personal blogs or web pages; Paper that do not focus on the use of media in RE processes. |
| Exclusion criteria for full text | Research which does not treat the media use to support software development and maintenance |

3.3.3. Data Extraction and Analyses

The studies found were classified in five categories corresponding to each of the research questions of the systematic mapping. Three authors built the classification used to categorize the selected articles. The categories are presented in table 8.

Table 8. CLASSIFICATION

| Categories | Description |
|----------------------------|--|
| <i>Approaches</i> | To classify the studies according to approaches used to capture and record user requirements using multimedia resources in order to support the software development and maintenance. |
| <i>Impacts</i> | To list the positive e negative impacts of using media resources in software lifecycle, pointing key success factors that may generalize to other interdisciplinary software modeling environments. |
| <i>Research Method</i> | To identify whether the proposed approach has been evaluated through empirical methods, and if so, which method was used. It was considered that a study has an empirical evaluation if it brings at least one section with some discussion dedicated to this topic. |
| <i>Kind of publication</i> | To identify the kind of publications where the articles have been published. |

| Categories | Description |
|------------------|--|
| <i>Scenarios</i> | To identify and characterize the scenarios where media are used as support to requirements analyze, in order to know in which contexts it can be successfully applied. |

3.4. Results and Discussion

In this section, the results of the primary studies obtained in each step of the search process are presented in Fig 2.

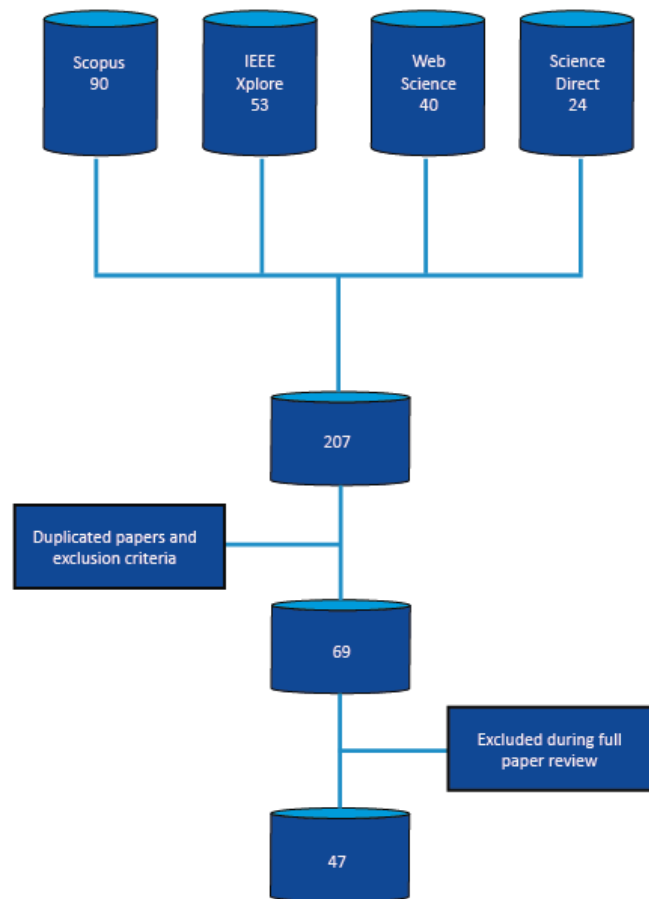


Fig 2. Results obtained from the search process.

The initial search returned 207 publications. Duplicate researches were removed, then, the selection criteria were applied, and 138 studies were rejected, leaving 69 works for full paper review. 22 papers were discarded after full reading and analyses. Thus, 47 publications were selected for the data extraction phase.

3.4.1. Research Questions

The first considered aspect in the classification was the approaches (**RQ1**) identified. The distribution of studies in terms of approaches are shown in Table 9.

Table 9. APPROACHES

| Ref. | Name | Description | Workflow and Multimedia Type |
|-------------|---|--|---|
| [24] | TRECRE | The TRECRE framework involves a wide variety of technologies for recording, preserving, linking and reviewing requirements. | The framework is structured in 4 layers: Domain models & Requirements specifications; Semi-structured index descriptions; Indices and Metadata; and, Multimedia source materials (video or audio). Software engineers thus navigate up and down the layers to find and review the information they desire. |
| [25] | MURMER | This method represents requirements for multimedia document types containing audio, video, image, or textual data. | MURMER represents content and sequence timelines with three Formal Sets: 1) a Storyboard (PPT) of the proposed presentation, 2) a Content File list of audio files, video files, image files, and text files, and 3) one or more Finite State Machines showing the timeline of frames. Together, these three elements represent at an abstract level all of the necessary elements for a multimedia presentation such as a simple tutorial. |
| [8] | No name was specified for this approach | Modern mobile devices typically provide multimedia features like microphones or digital cameras. Therefore, using such devices to enrich requirements descriptions with audio notes or videos is an obvious opportunity. | As stakeholders cannot directly interact with the actors they have been developing the Mobile Scenario Presenter (MSP), a mobile scenario-based tool augmenting ART-SCENE (scenario-driven technique for requirements discovery and documentation) in several iterations has been proposed. Using the MSP mobile, stakeholders can discover and document requirements systematically in the work context using structured scenarios generated by ART-SCENE. |
| [9] | The Software Cinema | This paper presents a novel technique for the video analysis of scenarios, relating the | The Software Cinema technique uses film as a semiformal representation of software models. They employ it to bridge the rhetoric gap between end-users and engineers. This addresses two |

| | | | |
|------|---|--|---|
| | Technique | use of video-based requirements to process models of software development. | issues: First, providing a model of reality that all stakeholders can understand equally well. Second, giving all involved developers a rich base of reference for what the complete system is intended to achieve. |
| [10] | No name was specified for this approach | A framework that is designed to improve the requirements elicitation process as well as the traceability of requirements throughout the complete life cycle of a project. | To accomplish this task, requirements elicitation sessions are recorded using rich media and subsequently the stakeholder requests, that describe a requirement, are extracted and stored in short multimedia clips. Those clips are then stored in the Sysiphus database, where they can be linked to the extracted requirements or to UML model elements. |
| [26] | No name was specified for this approach | A theoretical framework through which any decision about the use of different media for recording requirements-related information can take place in an informed manner. | It distinguishes between abstract media and physical media and uses this distinction to clarify the nature of multimedia. They suggest to use these distinctions to define a number of canonical media transformations which, in turn, reveal how information can be lost or gained as it is translated between media and hence between evolving requirements descriptions. |
| [27] | Dolli Project | The DOLLI project was a large-scale educational student project course with a real customer. They experimented with a shift from a traditional life-cycle to an agile process during the project, and used video techniques for defining requirements and meeting capture. | They used video to visualize the requirements and to teach soft skills. All reviews (with and without the customer) were filmed to provide feedback. The videos were automatically converted and made accessible on the team specific web pages. To visualize the project requirements, they used a technique called Video-based Requirement Engineering. |
| [28] | Wiki System | The method uses collaborative | The method uses interviews for the requirements elicitation. Further, a software tool – which is still |

| | | | |
|------|---|--|---|
| | Multimedia | technologies (a wiki system) and audio recordings to allow multiple stakeholders the joint elicitation and documentation of the requirements. | to be developed – is used to capture the audio information of the interviews and notes of the requirements elicited. The audio information and the requirements are linked to enable the traceability of the rationales and discussions in subsequent development steps. |
| [22] | No name was specified for this approach | They show that Ad-Hoc videos can work comparably or better than use cases for avoiding misunderstandings in the early phases of a project. | Use cases vs. ad-hoc videos are used to document elicited requirements, and to present them to the customers for validation. Counting recognized requirements is afforded by using lists of explicit requirements as a reference. |
| [29] | Storytelling | Storytelling as a technique was used for knowledge management. It uses Stories to pass along knowledge and record requirements. | It is based on group storytelling, where stakeholders tell stories about current and past systems that support a given activity. The stories are merged to form a single story. Stories are then transformed into scenarios, and from scenarios to Use Cases. The solution consists of a knowledge model based on stories about the system, a collective construction method, and a tool to support interactions. |
| [30] | A Virtual World (VW) environment | A Virtual Dynamic 3D Environment is a setting that provides realism in testing and conveying requirements for projects, products, and services | A representation of a real world (RW) scene is constructed first, based on interviews with stakeholders. The real setting is modeled at a reduced level of detail that focuses on the most central and common elements without distractions from minor visual details. For a new product or service, the virtual setting should be familiar to users and stakeholders |
| [23] | VisionCatcher | It focuses on direct interaction with stakeholders and employs several multimedia technologies for documentation of user requirements. | VisionCatcher supports the stakeholder in expressing the vision of a new system or an innovative idea for improving an already running system. Visions are depicted as a special kind of video, which are enhanced by multimedia technologies. |

| | | | |
|------|---|--|--|
| [31] | PRESTO 4U | European Technology for Digital Audiovisual Media Preservation in order to ensure development and use of high-quality software both by technology and service providers as well as media owners. | The Presto4U aggregation tasks are: metadata mapping and validation, audio-visual material storage, information extraction, manual content annotation, rights management and preservation platforms. |
| [32] | ReqVidA | This article proposes the combination of textual minutes and video with a software tool. | The objective is connecting textual notes with the corresponding part of the video. By highlighting relevant sections of a video and attaching notes that summarize those sections, a more useful structure can be achieved. This structure allows an easy and fast access to the relevant information and their corresponding video context |
| [33] | No name was specified for this approach | This article proposes a framework that provides a real time application for recording the conversation and converting it into a text transcript. | A text mining program is used to describe requirements and customized additional words. To capture verbal discussion and translate it into a text transcript, a real time audio-to-text conversion software is used. To capture the action items that are within the text transcript, they choose to use Fillmore's case theory. |
| [34] | CrowdRE | This article proposes an approach of video as a by-product of digital prototyping to specify and document scenarios. | They propose video as a medium for communicating problems, solution alternatives, and arguments effectively within a mixed crowd of officials, citizens, children and elderly people. |
| [35] | AVW | The findings show that when learners engage in commenting on videos and rating others' comments, their understanding about requirements increases. | AVW offers Personal Space and Social Space. Initially students watch and comment on videos individually in the Personal Space, using aspects to tag their comments. The system time-stamps comments (i.e. the time elapsed from the start of video). The student can watch videos multiple times, including rewinding or skipping parts. |
| [36] | No name was | This paper reports on a study that compared the | They designed a laboratory experiment which uses a scenario that illustrates the conflict between |

| | | | |
|------|---|--|--|
| | specified for this approach | performance of groups in face-to-face and distributed requirements negotiation meetings, paying special consideration to the socio-psychological aspects of group interaction in both communication media. | requirements scope and development time constraints. The multimedia meeting system used in the distributed settings was Microsoft's NetMeeting, a system that allowed video-conferencing and real-time sharing of an editor for the manipulation of requirements. |
| [37] | No name was specified for this approach | This paper describes an automatic speech recognition technique for capturing the non-functional requirements spoken by stakeholders at open meetings and interviews during the requirements elicitation process. | This approach uses a context-free grammar to boost recognition accuracy, segment the stakeholders' utterances and finally to classify the recognized statements by quality type. |
| [38] | UTOPIA | This paper proposes the use of multimedia requirements as a means to increase the empathy of software designers with older users. | The older people's real experiences and the findings were distilled by the scriptwriter into narratives which encapsulated a range of issues within an engaging, cohesive and 'dramatic' storyline. These were then produced as short vignettes using professional actors, director, and crew. |

The second considered aspect in the classification was the **impact (RQ2)** of using media in software development and maintenance. Elements in the studies, which describe their findings about the use of media to support RE processes, were found. In this way, a classification to categorize the aspects identified was created.

Table 10. IMPACTS

| | Description |
|---|---|
| <i>Communication support capability</i> | It supports the designer communication over simple and complex scenarios. |

| | Description |
|---|---|
| <i>Decision support</i> | Decision support during requirements analysis; focus on the discussion and analysis of the artifact and not the creation and referencing notes. |
| <i>Abstraction as close to reality as possible</i> | Video record provides more contexts than pure transcript. |
| <i>Faster familiarizing with a scenario</i> | Video allows capturing the dynamic aspect of interaction, facilitating the recognize the stakeholder's needs |
| <i>Shared understanding and specification quality</i> | It improves understanding and validation of a formal specification; clarifying ambiguous requirements; stakeholders understand multimedia videos; the sharable artifacts. |
| <i>Traceability</i> | The ability to verify the history, location, or application of an item by means of documented recorded identification. |

Among the elements and aspects analyzed by publications in the area, the technique proposes by [39] had the advantage of reducing complicated stories into a set of sequential, cooperating episodes. The multi-agent story capability supports the used designer communication over simple and complex scenarios, thus sequential and parallel episodes can be easily observed and discussed for further refinements.[40] indicates that a multimedia architecture to support requirements analysis is desirable and effective, allowing the user to focus on the discussion and analysis of the artifact and not on the creation and reference of notes.

From the point of view of problems and limitations, according to [41], the use of real world scenes is restricted to the re-engineering of systems in which system usage is observable and recordable. This limitation can be solved by the persistent capture of context in the form of real world scenes recorded in multimedia. In this sense, [42] list key success factors that may be generalize to interdisciplinary modeling environments include: the choice of a common abstraction as close to reality as possible, and an artifact sharable by all members of the team.

[10] have identified other problems: a system which films people influences them simply by the fact the people know they are observed (it can influence the result). On the other hand, according to [43], the use of video, photo and audio acts as a catalyst for fast-paced stakeholder interaction, improving the classical stakeholder-meeting situation. Moreover, [32]

have identified that requirements quality is about 80 percentage points higher by using tool-supported video analysis instead of a common technique like textual. In this same line, [44] explain that a textual scenario can be faster understood with the support of a dynamic video, generated by their approach, than with the support of static mockups. Video allows capturing the dynamic aspect of interaction and provides developers the benefit of familiarizing themselves faster with a scenario.

After this compilation of critical aspects and scenarios in the area, the studies were classified considering the categories mentioned in table 10. Table 11 shows the results of this classification.

Table 11. DISTRIBUTION OF PRIMARY STUDIES BY IMPACT

| Impact. | Studies |
|--|--|
| Communication support capability | [39], [20], [45], [36], [46], [47], [9], [10], [38], [48], [49], [43], [23], [50], [33] |
| Decision support | [42], [25], [51], [52], [43], [34] |
| Abstraction as close to reality as possible | [42], [53], [41], [24], [54], [25], [8], [55], [9], [10], [26], [38], [22], [30], [56] |
| Faster familiarizing with a scenario | [36], [46], [21], [8], [9], [57], [27], [44] |
| Shared understanding and specification quality | [58], [45], [21], [47], [25], [8], [9], [10], [57], [48], [28], [59], [29], [23], [31], [32], [35] |
| Traceability | [41], [58], [21], [60], [24], [8], [26], [51], [47], [53] |

Fig. 3 shows the distribution of the primary studies according to the followed **research method (RQ3)**. Exploratory study is the main research method used for construction of the papers.

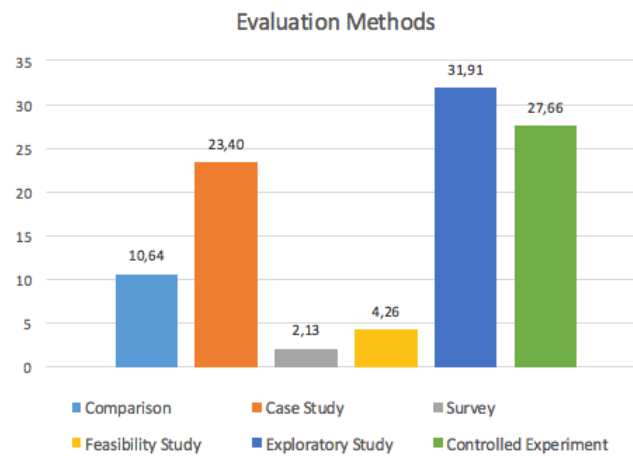


Fig 3. Distribution of primary studies by Evaluation Method.

The fourth considered aspect was the **type of publication (RQ4)**. Fig. 4 shows that 33 (70.22%) of primary studies were published as conference proceedings; 14 (29.78%) were papers published in periodicals.



Fig 4. Distribution of primary studies by Type of Publication.

The fifth aspect considered was **scenarios (RQ5)**. There is a plurality of contexts where multimedia resources can be used to software development and maintenance. [27],[35] [54, 61] have proposed the use of multimedia resources to requirements elicitation in learning and *teaching environments*. [39, 53] proposed to capture current system usage using rich media (e.g., video, speech, pictures, etc.) and to interrelate those observations with the goal definitions. This technique was used in real world scenes to development simple and complex software.

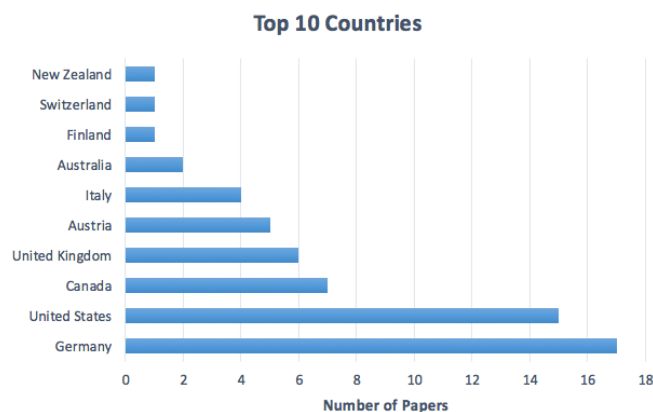
The articles [21, 51, 53], for instance, have proposed approach to track software's artifacts using video-on-demand to deal with requirements traceability (RT) in real-world scenarios. [42] presented a tool implemented as part of a process-integrated modeling environment to attend demands in the Automotive Industry. [55] proposed techniques for capturing artistic context to implement video games and the pre-production phase of video development.

[43][49] used video techniques for defining requirements and capture meetings in development of Software-Intensive Systems. [27] proposed to use video presentation in teaching innovation to develop software-intensive systems. They had to solve a real problem that was posed by a real customer: the Munich Airport. Finally, [22] proposed to use Ad-Hoc videos as a concrete representation of early requirements in the development embedded systems. Table 12 shows the distribution of studies by scenarios cited.

Table 12. SCENARIOS

| Scenarios | Studies |
|---|---|
| Learning and teaching enviroment | [61], [54], [27], [35] |
| Simple and complex software Development | [39], [53] |
| Requirements traceability | [47], [60], [41], [21], [51], [53], [58], [24], [8], [26] |
| Automotive Industry Software | [42] |
| Video games development | [55] |
| Software-Intensive Systems | [49], [43], [27] |
| Embedded Systems development | [22] |

Fig. 5 shows the distribution of the papers according to **country that have more researchers who published on this field (RQ6)**. Only the top 10 countries in this research area were considered. These countries (number of papers is in parentheses) were: Germany (17); United States (15); Canada (7); United Kingdom (6); Austria (5); Italy (4); Australia (2); Finland (1); Switzerland (1) and New Zealand (1). The countries where the research was conducted were considered.

**Fig 5. Distribution of articles by country.**

Some articles have the participation of more than one country, which justifies the numerical divergence in the totalizer of studies by country.

Fig. 6 shows the distribution of the studies according to the **year that they were published (RQ7)**. The year with the highest number of publications was 2006, with 10

published articles. There is a noticeable decrease in the number of publications from 2006 to 2017 and many oscillations over the years. After the conclusion of this study, which includes papers until mid-2018, in the journal review phase, the search string was executed again in the scientific databases, considering papers published between 2018-2020. 16 studies were identified: Web of Science (6), IEEE (4), ScienceDirect (1), and Scopus (5).

After applying exclusion criteria, 9 papers were removed and 7 studies were selected: [62][63] describe approaches already mentioned in this study (CrowdRE and AVW); [64] proposes a videogame as a tool to reinforce students' skills, in terms of identification and classification of requirements; [65] proposes creating a central repository of programming videos, enabling analyzing and annotating videos to illustrate specific behaviors of interest, such as asking and answering questions, employing strategies, and software engineering theories; [66] conducted a survey to investigate if software professionals perceive video as a medium that can contribute to RE; and [67][68] provide guidance how to create useful videos for visual communication to enable software professionals to produce good videos at moderate costs, yet sufficient quality. Three papers were published in 2018 and four in 2019, maintaining the trend previously mentioned.

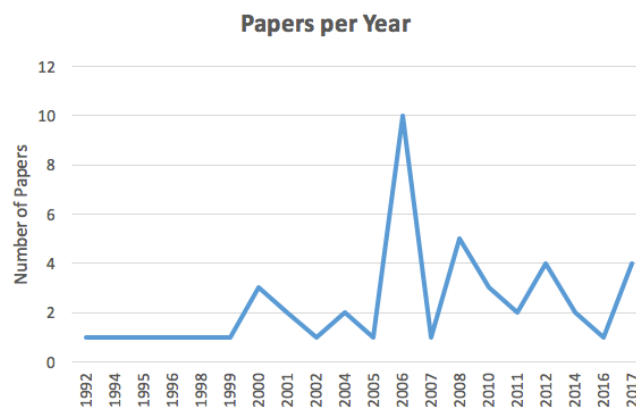


Fig 6. Distribution of articles by publication year.

Fig. 7 presents the top 10 authors (RQ8) who appeared in 2 papers or more, and they (number of papers in parentheses) were: Schneider, K. (7); Damian, D E H. (5); Grunbacher, P. (4); Bruegge, B. (4); Pohl, K. (3); Haumer, P. (3); Jarke, M. (3); Gaines, B R. (3); Karras, O. (3) and Egyed, A. (3).

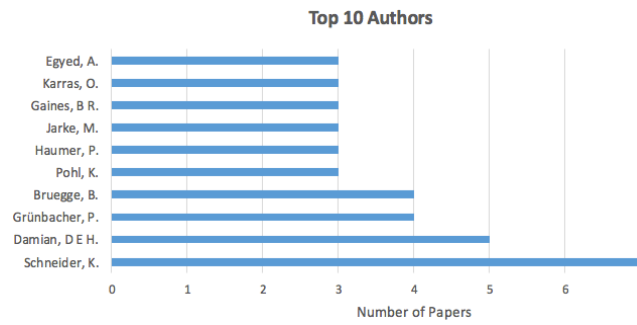


Fig 7. Distribution of articles by author.

Table 13 provides a mapping of the studies considering the following aspects: RE activities supported by the use of multimedia resources; type of content persisted in multimedia; and tools used to manage the audio, video or image files, which contain the user requirements.

Table 13. OUTCOMES

| Id | Ref. | RE activities where it was applied | Type of content described in the audio or video or image | Tools used to support the management of the media materials |
|-----------|-------------|---|---|--|
| 1 | [24] | Requirements elicitation, specification and traceability. | Conversations with customers, users and other experts. | They used the TRECE framework to store and manage the recorded files. |
| 2 | [25] | Requirements specification. | User stories, objectives, stakeholder profile. | To support this process, they built a method called MURMER. The method is simple as it relies on well-known tools such as PowerPoint and state machines to represent the entire presentation. Specific tools were not mentioned for recording multimedia requirements. |
| 3 | [8] | Requirements specification. | Interviews conducted with end-users in the workplace. | The integration of the Mobile Scenario Presenter with Microsoft Pocket Word. |
| 4 | [9] | Requirements specification. | Discussion between designers and developers. Discussion with the end-users. | To support this process, they built a custom tool for video-based requirements called Xrave. |

| | | | | |
|----|------|---|---|---|
| 5 | [10] | Requirements elicitation, specification and traceability. | Requirements elicitation sessions are recorded with stakeholder requests. | Requirements elicitation sessions are recorded using rich media and subsequently the stakeholder requests, that describe a requirement, are extracted and stored in short multimedia clips. Those clips are then stored in the Sysiphus database. |
| 6 | [26] | Requirements elicitation, specification and traceability. | Interviews with stakeholders. | They proposed a theoretical framework for managing the artifacts |
| 7 | [27] | Requirements specification. | Meetings with stakeholders. | The teams recorded their meetings themselves with cameras attached to a Podcast Producer system. With the Podcast Producer technology, the videos were automatically converted and made accessible on the team specific web pages. |
| 8 | [28] | Requirements elicitation and specification. | Interviews with a stakeholder | The method uses a collaborative software development platform as a tool infrastructure. Also part of the tool is a wiki which is used to identify and document requirements. |
| 9 | [22] | Requirements elicitation, validation and negotiation. | Discussion with customers. | Specific tools were not mentioned for recording and managing multimedia requirements. |
| 10 | [29] | Requirements elicitation and specification. | Stakeholders' stories about current and past systems that support a given activity. | They developed a tool called Storytelling that was used for knowledge management. |

| | | | | |
|----|------|---|--|--|
| 11 | [30] | Requirements elicitation, specification and validation. | Interviews with stakeholders. | They used a Virtual Dynamic 3D Environment to describe the requirements. |
| 12 | [23] | Requirements elicitation and specification. | Interactions with stakeholders. | They employed several multimedia technologies for documentation of user requirements. |
| 13 | [31] | Requirements elicitation. | Interviews and questionnaires with stakeholders. | They developed their own technology called Presto4U. |
| 14 | [32] | Requirements elicitation. | Workshops with stakeholders and team develop. | They developed their own technology to support analysis of an enriched video. |
| 15 | [33] | Requirements elicitation and specification. | Interviews with stakeholders. | In order to capture verbal discussion and translate it into a text transcript, a real time audio-to-text conversion software, such as Dragon NaturallySpeaking Premium, is used. A text mining program is used to describe requirements and customized additional words. |
| 16 | [34] | Requirements elicitation, specification and distribution. | Meetings with stakeholders. | They proposed their own application called CrowdRE. |
| 17 | [35] | Requirements elicitation and specification. | Meetings with stakeholders. | AVW offers Personal Space and Social Space to share interviews with stakeholders. |
| 18 | [36] | Requirements distribution, negotiation and specification. | Meetings with stakeholders. | They used Microsoft's NetMeeting, a system that allowed video-conferencing and real-time sharing of an editor for the manipulation of requirements. |

| | | | | |
|----|------|---|--|--|
| 19 | [37] | Requirements elicitation. | Meetings and interviews during the requirements elicitation process. | This paper developed an automatic Speech-based detection tool for capturing the non-functional requirements spoken by stakeholders at open meetings and interviews during the requirements elicitation process. This approach uses a context-free grammar to boost recognition accuracy, segment the stakeholders' utterances and finally to classify the recognized statements by quality type. |
| 20 | [38] | Requirements elicitation and specification. | User Stories. | Specific tools were not mentioned for recording and managing multimedia requirements. |

Most studies cite requirements specification and elicitation as the main RE activities supported by the use of multimedia resources. The articles also cite the use of audio and video for requirements validation, negotiation and distribution. The results also point to the use of media to store interviews and meetings with stakeholders, user stories, discussions with customers and end-users. To store and manage the recorded files, some articles developed their own tools, such as [9][24][25][31]. Some studies used tools already available on the market, such as [36][33][10][30].

3.4.2. Approaches Overviews

In this section, the numbering presented corresponds to the ids in table 13.

- 1) This article proposes use multimedia resources to record the discussions with stakeholders, and saving the sketches, images and other artifacts. They will then provide means for efficiently searching and browsing all of these materials, and as automatically as possible, link media to related domain models and requirements specifications. Thus, they are proposing a multimedia traceability framework to address the capture and review of

requirements source materials, aiming to complement, not modify, existing requirements methods and artifacts.

- 2) The MURMER method represents a starting point for representing simple multi-media tutorials that merge audio, video, text, and images. Its primary function is that of a risk mitigator, so that risky or costly components such as video (or animation) can be postponed and designed around at a very early stage. The method is simple as it relies on well-known tools such as PowerPoint and state machines to represent the entire presentation. MURMER also strives to be customer-oriented, since it is simple and accessible to customers, can be done at the early (requirements specification).
- 3) The focus of this paper is to present and analyze two options supporting capturing multimedia descriptions of requirements with a mobile scenario tool. They explore which option is more amenable to different user types. As little is known about capturing multimedia requirements descriptions using mobile RE tools, this comparison can stimulate further research in this field.
- 4) This paper presents a novel technique for the video analysis of scenarios, relating the use of video-based requirements to process models of software development. It uses a knowledge model—an RDF graph—based on a semiotic interpretation of film language, which allows mapping conceptual into formal models. It can be queried with RDQL, a query language for RDF. The technique has been implemented with a tool which lets the analyst annotate objects as well as spatial or temporal relationships in the video, to represent the conceptual model. The video can be arranged in a scenario graph effectively representing a multi-path video. It can be viewed in linear time order to facilitate the review of individual scenarios by end-users. Each multi-path scene from the conceptual model is mapped to a UML use case in the formal model.
- 5) This article proposes a framework that is designed to improve the requirements elicitation process as well as the traceability of requirements throughout the complete life cycle of a project. To accomplish this task, requirements elicitation sessions are recorded using rich media and subsequently the stakeholder requests, that describe a requirement, are

extracted and stored in short multimedia clips. These clips are then stored in the Sysiphus database, where they can be linked to the extracted requirements or to UML model elements.

- 6) This article distinguishes between abstract media and physical media and uses this distinction to clarify the nature of multimedia and recording requirements. The interview session is guided by a pre-written questionnaire, so the abstract media is text and the physical media is paper. The questions are delivered by the interviewer using speech (abstract media) carried on sound waves (physical media). The interviewee's response is communicated using spoken natural language (speech as abstract media), recorded together with any background noise (sound as abstract media). Once the interview session has been recorded, the subsequent activities to determine requirements are performed.
- 7) This paper experimented with a shift from a traditional lifecycle to an agile process during the project, and used video techniques for defining requirements and meeting capture. The task of the film team was to create scripts and shoot three films: A scenario film that visualizes the functional requirements, a making-of film that describes the interaction and the atmosphere between the project participants, and a trailer that can be used to market the project. To visualize the project requirements, they used a technique called Video-based Requirement Engineering.
- 8) This article proposes a method, which supports communication processes between stakeholders of a software development project in order to elicit requirements more efficiently and effectively. Thus, their focus lies on the first phase of the requirements engineering process. The method uses interviews for the requirements elicitation. Further, a software tool is used to capture the audio information of the interviews and notes of the requirements elicited. The audio information and the requirements are linked to enable the traceability of the rationales and discussions in subsequent development steps.
- 9) In this work, they investigate whether videos can replace textual requirements representations. They compare ad-hoc videos with use cases as a widely used textual representation of requirements. Firstly, they compare the efficiency of creating videos and textual requirements

descriptions by subjecting the analysts to time pressure. Secondly, they investigate the effectiveness, i.e., whether customers can distinguish valid from invalid requirements when they see them represented as use cases, or in videos. These videos are created based on the requirements from elicitation meetings.

- 10) This research investigates the usage of the Storytelling technique in eliciting requirements. A small-scale experiment was conducted to gain insights on the practice of telling stories related to Requirements elicitation context. In the experiment, Storytelling was used to collect and discover the maximum number of requirements in forms of short stories revealing personal experiences with the ticket machine. The stories included narrated real experiences, the good as well as bad ones, and anecdotes.
- 11) This work proposes the use of a virtual dynamic 3D environment, with realistic settings and multiple remotely located participants, to bring value for Requirements Engineering. The visual expressive power, distant collaboration, artifact permanence and availability, and the richness of realistic 3D simulations bring added power and clarity. The ability to simulate and experience user satisfaction and pain are of great value in predicting the utility and acceptance of developments.
- 12) In this article, a new kind of dedicated multimedia storyboard is presented. The storyboard VisionCatcher supports the stakeholder in expressing the vision of a new system or an innovative idea for improving an already running system. Visions are depicted as a special kind of video, which are enhanced by multimedia technologies. In a meeting, the engineer constructs these videos in tight collaboration with the customer. Each action is depicted as a piece of multimedia (video-clip, photo, and audio-clip). Actions are stepwise concatenated in order to form a so-called VisionVideo. This way, requirements are documented in an expressive and highly concrete representation.
- 13) In this paper, a software quality model customizable for the audio-visual context has been performed. An experiment of requirements elicitation has been developed for the identification of some functional and nonfunctional requirements. Crucial work of this paper, is the definition of a software product quality evaluation process, which is customizable for tools used in

the audio-video preservation context. These tools cover different tasks: they can be either tools for metadata mapping or for audiovisual files archiving and restoring or for evaluating the quality of the various contents, as images and sounds, etc.

- 14) This paper presents a software tool, called ReqVidA – Requirements Video Analyzer. This software tool allows support and guidance of a requirements engineer in the role of a scribe during a workshop and afterwards. In general, this tool enables the creation and elaboration of annotations and minutes in both views (Recorder and Analyzer). The export of artifacts, e.g. minutes or a list of annotations, is also available at any time, during a workshop and afterwards.
- 15) In this paper, they propose a solution to remedy the problem of capturing verbal requirements communication by presenting a framework for recording, transcribing, and mining verbal communication to assist developers in better describing and further refining requirements. During the meeting or discussion, the application constantly sends the captured audio to the desktop application, which converts it into a text transcript. Once the translation is complete, the transcript is saved, and the text mining tool retrieves the file. Next, the text mining program outputs the extracted requirements in terms of verbs and nouns.
- 16) In this paper, they propose this new perspective as a timely opportunity for the spatial planning domain – and as an increasingly important application domain of CrowdRE. CrowdRE starts from the assumption that there is a crowd of participants who are able and willing to communicate via electronic media. Thus, they can receive electronic messages at short notice and have the technical infrastructure for responding.
- 17) In this study, the Active Video Watching (AVW) system was developed. AVW is a controlled video watching environment designed for self-study. It can be customized by the teacher who defines a list of aspects that serve as scaffolds for learning with videos. The student can watch videos multiple times, including rewinding or skipping parts. The main aim was to elicit requirements for intelligent support to improve learning with AVW.
- 18) This paper reports on a study that compared the performance of groups in face-to-face and distributed requirements negotiation meetings. The

multimedia meeting system used in the distributed settings was Microsoft's NetMeeting, a system that allowed video-conferencing and real-time sharing of an editor for the manipulation of requirements.

- 19) This paper describes an automatic speech recognition technique for capturing the non-functional requirements spoken by stakeholders at open meetings and interviews during the requirements elicitation process. Their approach uses a context-free grammar to boost recognition accuracy, segment the stakeholders' utterances and finally to classify the recognized statements by quality type.
- 20) This paper discusses the efficacy of narrative video to communicate some of the fundamental differences between older users of ICT interfaces and the interface designers who tend not to be familiar with the general perspectives and user requirements of this and other 'nontypical' target groups. A series of narrative is produced based videos to illustrate the kinds of problems that many older people face with ICT. These data and experiences were distilled by the scriptwriter into narratives which encapsulated a range of issues within an engaging, cohesive and 'dramatic' storyline.

3.5. Syntheses and analysis, and lessons learned

In this section, in addition to the discussion of impacts, scenarios and raw results, made in the previous section, a discussion of the main aspects and lessons learned on possible improvements for software evolution analysis, which can be developed for the analyzed tools and new researches, is presented.

The use of multimedia in the RE has been considered since 1992. This indicates that this line of research has been studied for some time. In this period, several solutions have been proposed to incorporate media in the process of requesting and maintaining requirements, however, the number of published studies is still low. This may indicate two things: (1) that the area has already produced definitive results; or (2) that the area is failing to reach its goals. Second case is the most likely, since multimedia and evolution of software are very important topics in modern software engineering; and multimedia for RE is a broad and relatively young research area, which has much to offer to the software evolution research community.

The results also show that the publications referring to the use of media in the RE involve professionals of different nationalities and are developed in different countries. This denotes that the search for improvements in requirements documentation and communication between stakeholders in the software construction and maintenance presents itself as a global concern being studied in different parts of the world.

Exploratory Study (31.92%) and Controlled Experiment (27.66%) are the main research methods used for evaluating. Despite the second-place finish, these numbers show that few approaches found validated their solutions through controlled experiments, showing the need to increase the use of scientific method in this area, with replications of studies that will allow evaluating if other researchers independently will come up with the same results. Even those that validated, performed a partial validation.

Any approach should focus on the end user and validate its usefulness in well-executed experiments or case studies. This is far from proposing a new multimedia resources association to the requirements and executing a few feasibility studies over it. Researchers and practitioners should focus on answering questions like: Does my approach scale to larger real-world software? How do my results generalize to other users, domains, and systems? Otherwise, there will be always an issue with the external validity of the proposed approach and it will be difficult to move from the state of the art to the state of the practice in software engineering. Still in this context, the actual adoption of multimedia resources in software engineering industrial environments is very low.

There is actually little collaborative work in the area. Most of the analyzed work tries to develop new approaches as oppose as to validate or add value to existing ones. Validation and cooperative activities would lead to faster improvement of existing approaches and to a deeper understanding of the area. Researchers never benchmarked deeply their results against other approaches.

Most studies cite shared understanding and specification quality as the main impact of media incorporation in RE. This finding shows that media requirements can help reduce common issues encountered in RE, such as ambiguous, unclear and unchecked requirements.

A wide variety of solutions for recording, preserving, linking, and reviewing requirements using media have been used and tested, as alternative to traditional techniques for requirements elicitation. With respect to environments where the media can be used as a facilitator in software development and maintenance, the scenario is highly favorable, and the use of the media has been exploited in different contexts. It has been used in the development of complex software, which represents high risks (e.g. airport, health and financial systems).

These factors emphasize the reliability of the method approached in this work and show that multimedia resources can improve the process of understanding the code, decreasing evolution and maintenance costs.

Regarding gaps and software evolution analysis, integration with the source code and the construction (programming) phase needs to improve. The lack of closely integration with the code prevents multimedia tools can be effectively used to analyze and understand the data produced during software evolution. In addition to recording or filming the interviews with clients and provide them to the programmer so they can better understand their requirements, a programmer should be able to click on a link in the source code and to see or listen to stakeholders' interviews and code explanations recorded by a co-worker. At this point, explanations could include increments and evolutions.

Another feature to be improved and considered by the multimedia tools are the strategies of evolution analysis. Temporal strategies, for example, can portray the evolution considering all the versions available for analysis. Given n versions v_1, v_2, \dots, v_n , (or a sizeable sequential subset of them), this type of analysis takes into account everything that has happened from version v_1 to version v_n taking in consideration all the intermediate versions. That would help, for example, to analyze changes between different artifacts and business changes over the software evolution cycle. In other words, the versioning of multimedia resources must be aligned with the versioning of the software, allowing the filming or explanation of a business requirement, for example, to be associated with the evolution of metrics in the code. In another dimension, an issue to be evaluated would be the recording or filming of explanations about the evolution of each software version, allowing the main changes and key decisions to be documented.

Regarding software maintenance, the studies do not specifically or deeply address the use of multimedia resources applied to software maintenance and comprehension. In our research, we are proposing [5], implementing and considering closer coupling to the code, code technical explanations and new ways to find specific excerpts from audios and videos.

It is worth mentioning that the multimedia resources can support various RE activities, by recording interviews, meetings with stakeholders, user stories, and discussions with customers. The management of this content can be done through own initiatives, or through tools available on the market.

Finally, multimedia resources more closely linked to the code and strategies of evolution analysis can help answer two important questions: (1) Where do I see my business in the code? (2) What points have evolved in the code in line with the evolution of my business?

The answer to these questions can help increasing location accuracy of the code to be evolved and changed, increasing the effectiveness of maintenance, reducing costs and providing the prospect of impact on the source code, based on the business evolution.

3.6. Threats do validity

Threats to validity may limit the ability to interpret and/or describe results from the data obtained. Therefore, there is no way to disregard the following threats found in this study.

- **Construct Validity:** The search string and search secondary questions used may not cover the new area that includes use of multimedia resources to develop or maintain software. To mitigate this threat, a short string was developed, in order to increase sensitivity and retrieve all interest documents, with terms that could be used in the area and several synonyms. These terms were identified and refined with the help of control articles, using studies that were of interest to the research, and false positives, in order to calibrate the search string. In addition, the opinions of three researchers were considered.

- **Internal Validity:** (1) **Data Extraction:** Three researchers were responsible for extracting and classifying data from each publication. Therefore, biases or problems in data extraction may threaten the validity of data characterization. (2) **Selection Bias:** Initially, papers were included or excluded according to the researchers' own judgment. Consequently, some studies may have been incorrectly categorized. To mitigate these threats (1 and 2), selection and extraction reviews were made by all researchers involved and disagreements found were resolved in a final vote. (3) **Classification Bias:** Some selected articles did not make clear the methodology in detail, i.e., the research, evaluation or validation strategy. To mitigate this bias, these articles were read completely by the three researchers, in order to find characteristics that fitted a research type.

- **External Validity:** Although the researched scientific bases have thousands of journals and millions of conferences, it cannot be stated that the results of this systematic mapping covered the entire researches in software engineering and multimedia. However, this work presented evidence of the main techniques used, identifying gaps to be explored, and serving as a guide for future work in this line.

3.7. Conclusions

In this paper, a mapping study has been conducted to characterize the set of primary studies that address the use of multimedia resources in RE to support software comprehension and maintenance. 47 primary studies were selected and analyzed, identifying trends in this area.

The number of publications fluctuates a lot, the oldest publication dates back to 1992. The year with the most publications was 2006, with 10 papers published. Finally, the most studies cite shared understanding and specification quality as the main impact of media incorporation in RE.

Among the main channels, conferences stood out with 33 (70.22%), while periodicals reached 14 (29.78%). These results and the low number of controlled experiments indicate that the area still needs to mature. Journals are strict and experiments allow performing replications to consolidate and validate the tools, as well as to use more rigorous protocols, which allow such replications.

A gap found in the articles was the parsimony in the description of the approaches, being restricted to the description of some characteristics of functioning. The source code and implementation details were not made available. This fact makes it difficult to verify the effectiveness of the proposed solutions and the proliferation of techniques.

In addition, integration with the source code and the construction (programming) phase needs to improve. The lack of closely integration with the code prevents effective mapping of software entities (such as packages, classes, and methods in OOP) and their attributes (metrics, properties, and features) to multimedia resources (audio and video) in a way that they can be easily explored by software engineers and also allow to analyze the software evolution with respect to a set of software maintenance related questions.

The results obtained in this work demonstrate that it is an area of interest for researchers worldwide and can be explored more deeply for software maintenance, with closer coupling to the code and new ways to locate specific excerpts of audios and videos. In this sense, this study is relevant to the software companies, and universities, fostering the need for interdisciplinary research between the areas of Multimedia and Software Engineering.

As future work, the research group that planned and executed this research will publish the experimental evaluation of an open tool, under development, which can be coupled with an integrated software development environment. The plug-in will focus on four main aspects: recording requirements using multimedia resources; storage and maintenance of multimedia requirements; direct link with code; and event-based playback of files.

The approach intends for multimedia resources to be integrated into the software construction process, giving programmers different channels for representing information and

the ability to ascertain the software evolution from the point of view of changing requirements and the emergence of new ones.

3.8. References

1. SOMMERVILLE, I.: Engenharia de Software. (2011).
2. Emília, R., Teixeira, Z., Prado, G., Giglio, D.M.: Estudo do desenvolvimento de uma ferramenta de Gerência de Requisitos para apoio aos processos de manutenção de software . Cad. Estud. em Sist. Informação. 1, (2017).
3. Moody, D.L.: The “ Physics ” of Notations : Toward a Scientific Basis for Constructing Visual Notations in Software Engineering. IEEE Trans. Softw. Eng. 35, 756–779 (2009).
4. Fredericks, M., Basili, V.R., Park, C.: Detecting Defects in Object Oriented Designs : Using Reading Techniques to Increase Software Quality. ACM SIGPLAN Not. 34, 47–56 (1999).
5. Colaço Junior, Methanias; de Fatima Menezes, Maria; Corumba, Daniela; Mendonca, Manoel and Santos, B.: Do Software Engineers Have Preferred Representational Systems? J. Res. Pract. Inf. Technol. (2017).
6. L. Erlikh: Leveraging legacy system dollars for e-business. IT Prof. 2, 17–23 (2000).
7. Fjeldstad, R; Hamlen, W.: Application program maintenance - report to our respondents. Tutor. Softw. Maint. 13–27 (1983).
8. Rabiser, R., Seyff, N., Grünbacher, P., Maiden, N.: Capturing multimedia requirements descriptions with mobile RE tools. In: First International Workshop on Multimedia Requirements Enineering, MeRE’06 (2006).
9. Creighton, O., Ott, M., Bruegge, B.: Software cinema - Video-based requirements engineering. In: Proceedings of the IEEE International Conference on Requirements Engineering. pp. 106–115 (2006).
10. Gall, M., Bruegge, B., Berenbach, B.: Towards a framework for real time requirements elicitation. In: First International Workshop on Multimedia Requirements Enineering, MeRE’06 (2006).
11. Colaço Júnior, M., Mendonça, M., Farias, M., Santos, P.H. dos.: OSS Developers Context-Specific Preferred Representational Systems : A Initial Neurolinguistic Text Analysis of the Apache Mailing List. In: 7th IEEE Working Conference on Mining Software Repositories, Cape Town, SA, 2010, pp. 126-129, doi: 10.1109/MSR.2010.5463339.

12. Kitchenham, B.: Procedures for performing systematic reviews. Keele University Technical Report TR/SE-0401, UK, Keele Univ. 33, 28 (2004).
13. Petersen, K., Feldt, R., Mujtaba, S., Mattsson, M.: Systematic Mapping Studies in Software Engineering. 12Th Int. Conf. Eval. Assess. Softw. Eng. 17, 10 (2008).
14. Punter, T., Ciolkowski, M., Freimut, B., John, I.: Conducting on-line surveys in software engineering. Proc. - 2003 Int. Symp. Empir. Softw. Eng. ISESE 2003. 80–88 (2003).
15. Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empir. Softw. Eng. 14, 131–164 (2009).
16. Sjøberg, D.I.K., Hannay, J.E., Hansen, O., Kampenes, V.B., Karahasanović, A., Liborg, N.K., Rekdal, A.C.: A survey of controlled experiments in software engineering. IEEE Trans. Softw. Eng. 31, 733–753 (2005).
17. Fernandez, A., Abrahão, S., Insfran, E.: A systematic review on the effectiveness of web usability evaluation methods. IET Semin. Dig. 2012, 52–56 (2012).
18. Novais, R.L., Torres, A., Mendes, T.S., Mendonça, M., Zazworka, N.: Software evolution visualization: A systematic mapping study. Inf. Softw. Technol. 55, 1860–1883 (2013).
19. James, K.L., Randall, N.P., Haddaway, N.R.: A methodology for systematic mapping in environmental sciences. Environ. Evid. 1–13 (2016).
20. Herlea Damian, D.E., Eberlein, A., Shaw, M.L.G., Gaines, B.R.: Using different communication media in requirements negotiation. IEEE Softw. 17, 28–36 (2000).
21. Egyed, A., Grünbacher, P.: Automating requirements traceability: Beyond the record & replay paradigm. In: Proceedings - ASE 2002: 17th IEEE International Conference on Automated Software Engineering. pp. 163–171. Institute of Electrical and Electronics Engineers Inc. (2002).
22. Brill, O., Schneider, K., Knauss, E.: Videos vs. use cases: Can videos capture more requirements under time pressure? Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics). 6182 LNCS, 30–44 (2010).
23. Pham, R., Meyer, S., Kitzmann, I., Schneider, K.: Interactive multimedia storyboard for facilitating stakeholder interaction: Supporting continuous improvement in IT-ecosystems. In: Proceedings - 2012 8th International Conference on the Quality of Information and Communications Technology, QUATIC 2012. pp. 120–124 (2012).
24. Richter, H., Gandhi, R., Liu, L., Lee, S.-W.: Incorporating multimedia source materials into a traceability framework. In: First International Workshop on Multimedia Requirements Engineering, MeRE'06 (2006).

25. Witmer, K., Koss, R.B., Kasza, T.: A method for risk mitigation during the requirements phase for multimedia software systems. In: ACM SIGDOC 2006 - Proceedings of the 24th ACM International Conference on Design of Communication. pp. 19–22 (2006).
26. Gotel, O.C.Z., Morris, S.J.: Crafting the requirements record with the informed use of media. In: First International Workshop on Multimedia Requirements Engineering, MeRE'06 (2006).
27. Bruegge, B., Stangl, H., Reiss, M.: An experiment in teaching innovation in software engineering: Video presentation. In: Proceedings of the Conference on Object-Oriented Programming Systems, Languages, and Applications, OOPSLA. pp. 807–809 (2008).
28. Menten, A., Scheibmayr, S., Klimpke, L.: Using audio and collaboration technologies for distributed requirements elicitation and documentation. In: 2010 3rd International Workshop on Managing Requirements Knowledge, MaRK'10. pp. 51–59 (2010).
29. Boulila, N., Hoffmann, A., Herrmann, A.: Using Storytelling to record requirements: Elements for an effective requirements elicitation approach. In: 2011 4th Int. Workshop on Multimedia and Enjoyable Requirements Eng. - Beyond Mere Descriptions and with More Fun and Games, MERE'11 - Co-located with the 19th IEEE Int. Requirements Eng. Conf., RE'11. pp. 9–16 (2011).
30. Russell, S., Creighton, O.: Virtual world tools for Requirements Engineering. In: 2011 4th Int. Workshop on Multimedia and Enjoyable Requirements Eng. - Beyond Mere Descriptions and with More Fun and Games, MERE'11 - Co-located with the 19th IEEE Int. Requirements Eng. Conf., RE'11. pp. 17–20 (2011).
31. Biscoglio, I., Marchetti, E.: An experiment of software quality evaluation in the audio-visual media preservation context. In: da Silva A.R. da Silva A.R., M.R.J.B.M.A. (ed.) Proceedings - 2014 9th International Conference on the Quality of Information and Communications Technology, QUATIC 2014. pp. 118–123. Institute of Electrical and Electronics Engineers Inc. (2014).
32. Karras, O., Kiesling, S., Schneider, K.: Supporting Requirements Elicitation by Tool-Supported Video Analysis. In: Proceedings - 2016 IEEE 24th International Requirements Engineering Conference, RE 2016. pp. 146–155. Institute of Electrical and Electronics Engineers Inc. (2016).
33. Hollis, C., Bhowmik, T.: Automated support to capture verbal just-in-time requirements in agile development: A practitioner view. In: Proceedings - 2017 IEEE 25th International Requirements Engineering Conference Workshops, REW 2017. pp. 419–422. Institute of Electrical and Electronics Engineers Inc. (2017).

34. Schneider, K., Karras, O., Finger, A., Zibell, B.: Reframing societal discourse as requirements negotiation: Vision statement. In: Proceedings - 2017 IEEE 25th International Requirements Engineering Conference Workshops, REW 2017. pp. 188–193. Institute of Electrical and Electronics Engineers Inc. (2017).
35. Mitrovic, A., Dimitrova, V., Lau, L., Weerasinghe, A., Mathews, M.: Supporting constructive video-based learning: Requirements elicitation from exploratory studies. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*. 10331 LNAI, 224–237 (2017).
36. Damian, D.E.H., Eberlein, A., Shaw, M.L.G., Gaines, B.R.: Effects of communication media on group performance in requirements engineering. In: Proceedings of the IEEE International Conference on Requirements Engineering. p. 191. IEEE, Los Alamitos, CA, United States (2000).
37. Steele, A., Arnold, J., Cleland-huang, J.: Speech Detection of Stakeholders' Non-Functional Requirements Adam Steele, Jason Arnold, Jane Cleland-Huang. 0–7 (2006).
38. Carmichael, A., Newell, A.F., Morgan, M.: The efficacy of narrative video for raising awareness in ICT designers about older users' requirements. *Interact. Comput.* 19, 587–596 (2007).
39. Faro, A., Giordano, D.: From user's mental models to information system's specification and vice versa by extended visual notation. In: IEEE International Professional Communication Conference. pp. 44–48. IEEE, Piscataway, NJ, United States (1995).
40. Smith, J.D., Takahashi, K.: Multimedia architecture to support requirements analysis. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*. 1045, 15–31 (1996).
41. Haumer, P., Heymans, P., Jarke, M., Pohl, K.: Bridging the gap between past and future in RE: a scenario-based approach. In: Proceedings of the IEEE International Conference on Requirements Engineering. pp. 66–73. IEEE, Los Alamitos, CA, United States (1999).
42. Jarke, M., Miatidis, M., Schlüter, M., Brandt, S.: Media-assisted product and process traceability in supply chain engineering. In: R.H., S.J. (ed.) Proceedings of the Hawaii International Conference on System Sciences. pp. 1405–1414 (2004).
43. Gartner, S., Schneider, K.: A method for prioritizing end-user feedback for requirements engineering. In: 2012 5th International Workshop on Co-operative and Human Aspects of Software Engineering, CHASE 2012 - Proceedings. pp. 47–49 (2012).
44. Karras, O., Unger-Windeler, C., Glauer, L., Schneider, K.: Video as a by-product of digital prototyping: Capturing the dynamic aspect of interaction. In: Proceedings - 2017 IEEE

- 25th International Requirements Engineering Conference Workshops, REW 2017. pp. 118–124. Institute of Electrical and Electronics Engineers Inc. (2017).
45. Damian, D.: An empirical study of requirements engineering in distributed software projects: Is distance negotiation more effective? In: Proceedings of the Asia-Pacific Software Engineering Conference and International Computer Science Conference, APSEC and ICSC. pp. 149–152 (2001).
 46. Damian, D.E.H., Eberlein, A., Woodward, B., Shaw, M.L.G., Gaines, B.R.: An empirical study of facilitation of computer-mediated distributed requirements negotiations. *Proc. IEEE Int. Conf. Requir. Eng.* 128–135 (2001).
 47. Egyed, A., Grünbacher, P.: Identifying requirements conflicts and cooperation: How quality attributes and automated traceability can help. *IEEE Softw.* 21, 50–58 (2004).
 48. Damian, D., Lanubile, F., Mallardo, T.: On the need for mixed media in distributed requirements negotiations. *IEEE Trans. Softw. Eng.* 34, 116–132 (2008).
 49. Bruegge, B., Creighton, O., Reiß, M., Stangl, H.: Applying a video-based requirements engineering technique to an airport scenario. In: 2008 3rd International Workshop on Multimedia and Enjoyable Requirements Engineering, MERE'08 (2008).
 50. Khan, H., Ahmad, A., Alnuem, M.A.: Knowledge management: A Solution to requirements understanding in global software engineering. *Res. J. Appl. Sci. Eng. Technol.* 4, 2087–2099 (2012).
 51. Gotel, O.C.Z., Morris, S.J.: Macro-level traceability via media transformations. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*. 5025 LNCS, 129–134 (2008).
 52. Bruni, E., Ferrari, A., Seyff, N., Tolomei, G.: Automatic analysis of multimodal requirements: A research preview. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*. 7195 LNCS, 218–224 (2012).
 53. Haumer, P.: Requirements elicitation and validation with real world scenes. *IEEE Trans. Softw. Eng.* 24, 1036–1054 (1998).
 54. Cybulski, J.L., Parker, C., Segrave, S.: Touch it, feel it and experience it: Developing professional is skills using interview-style experiential simulations. In: ACIS 2006 Proceedings - 17th Australasian Conference on Information Systems (2006).
 55. Callele, D., Neufeld, E., Schneider, K.: Emotional requirements in video games. In: Proceedings of the IEEE International Conference on Requirements Engineering. pp. 292–295 (2006).

56. Kasurinen, J., Maglyas, A., Smolander, K.: Is requirements engineering useless in game development? *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*. 8396 LNCS, 1–16 (2014).
57. Yuhui, S., Lin, L., Fei, P.: Megore: Multimedia enhanced goal-oriented requirement elicitation experience in China. In: 2008 3rd International Workshop on Multimedia and Enjoyable Requirements Engineering, MERE'08 (2008).
58. Haumer, P., Jarke, M., Pohl, K., Weidenhaupt, K.: Improving reviews of conceptual models by extended traceability to captured system usage. *Interact. Comput.* 13, 77–95 (2000).
59. Primrose, M.C.: User experience grading via Kano categories. In: *Proceedings of the 2010 18th IEEE International Requirements Engineering Conference, RE2010*. pp. 331–336 (2010).
60. Egyed, A., Grünbacher, P.: Supporting software understanding with automated requirements traceability. *Int. J. Softw. Eng. Knowl. Eng.* 15, 783–810 (2005).
61. Dospisil, J., Polgar, T.: Conceptual modelling in the hypermedia development process. In: J.W., R. (ed.) *Proceedings of the 1994 Computer Personnel Research Conference on Reinventing IS: Managing Information Technology in Changing Organizations, SIGCPR 1994*. pp. 97–104. Association for Computing Machinery, Inc (1994).
62. Schneider, K., Bertoli, L.M.: Video Variants for CrowdRE: How to Create Linear Videos, Vision Videos, and Interactive Videos. In: *IEEE 27th International Requirements Engineering Conference Workshops (REW)*. pp. 186–192. IEEE (2019).
63. Galster, M., Mitrovic, A., Gordon, M.: Toward Enhancing The Training of Software Engineering Students and Professionals Using Active Video Watching. In: *40th International Conference on Software Engineering: Software Engineering Education and Training* Authorized licensed. pp. 2018–2021. ACM, Gothenburg, Sweden (2018).
64. Gabriel Elías Chanchí, G., María Clara Gómez, A., Wilmar Yesid Campo, M.: Proposal of an educational video game for the teaching-learning of the requirements classification in software engineering . *RISTI - Rev. Iber. Sist. e Tecnol. Inf.* 2019, 1–14 (2019).
65. Alaboudi, A., Latoza, T.D.: Supporting software engineering research and education by annotating public videos of developers programming. In: *Proceedings - 2019 IEEE/ACM 12th International Workshop on Cooperative and Human Aspects of Software Engineering, CHASE 2019*. pp. 117–118 (2019).
66. Karras O. (2018) Software Professionals' Attitudes Towards Video as a Medium in Requirements Engineering. In: Kuhrmann M. et al. (eds) *Product-Focused Software Process*

Improvement. PROFES 2018. Lecture Notes in Computer Science, vol 11271. Springer, Cham. https://doi.org/10.1007/978-3-030-03673-7_11

67. Karras, O., Schneider, K., Fricker, S.A.: Representing software project vision by means of video: A quality model for vision videos. *J. Syst. Softw.* 162, (2019).

68. Karras, O., Schneider, K.: Software Professionals are Not Directors : What Constitutes a Good Video ? In: 1st International Workshop on Learning from other Disciplines for Requirements Engineering. IEEE, Hannover, Germany (2018).

Uma vez apresentado o Mapeamento Sistemático, será explanado, no próximo capítulo, o Estudo Experimental que avaliou o desempenho da Abordagem Multimídia para Levantamento, Registro, Validação e Verificação dos requisitos de usuário, usando mídias dinâmicas (áudio e vídeo) integradas ao código-fonte.

4. AVALIAÇÃO EXPERIMENTAL

Neste capítulo, será apresentada a Avaliação Experimental realizada para avaliar a abordagem multimídia, no que concerne ao aumento da efetividade na compreensão e manutenção de software.

Construction and Evaluation of a Multimedia Approach to Software Maintenance and Comprehension

Anne Caroline M. Santos¹, Methanias Colaço Júnior², Edna de C. Andrade³, Rafael Meneses Santos⁴

^{1 2} Postgraduate Program in Computer Science - PROCC.
UFS – Federal University of Sergipe
São Cristóvão/SE – Brasil

^{2 3 4} Competitive Intelligence Research and Practice Group – NUPIC
Information Systems Department - DSI UFS – Federal University of Sergipe
Itabaiana/SE – Brasil

anne.santos@dcomp.ufs.br, mjrse@hotmail.com, ednacarvalorsempre@gmail.com,
rafaelmsse@gmail.com

Abstract.

Context: In Software Engineering, we still use textual documentation in most cases to represent software requirements. For some people, textual descriptions may be hard to understand, and they may need help of other representations and visualizations, like diagrams, video, audio conversations and so on. **Goal:** Propose and evaluate a multimedia approach to support the comprehension and maintenance of software, as an alternative to traditional techniques. **Method:** A controlled experiment was carried out in a real life setting to evaluate the efficiency and effectiveness of the proposed multimedia approach. **Results:** Our proposed approach showed best results in terms of effectiveness (average of correct answers for each task implemented) and level of user satisfaction. In the first case, our approach achieves an average of 4 correct answers per task, a number higher than the average of the case without using the approach; and the second, with a 7% increase in the level of customer satisfaction. Regarding the average coding time and the level of comprehension of the code, the multimedia approach proved to be less efficient. In this case, our approach achieves an average of roughly 49 minutes per task, a result higher than the average time without using the approach; and a decrease of 16% in the coefficient of code understanding. These results can be correlated with the

programming style and the experience time of the programmers. **Conclusion:** The multimedia approach is more effective in terms of correct coding and level of user satisfaction, making it a viable option for elicitation and registration of requirements. It is likely that, in a context with beginning programmers, the results will be much more promising regarding our approach. This is a new hypothesis to be tested. The project's most experienced programmers already know enough about the source code and don't need to consult our tool frequently.

4.1. Introduction

Requirements Engineering (RE) is a fundamental activity in the software development process. It is possible to organize the theoretical basis for the construction of any system through this activity, supporting the initial phases of its life cycle [1]. One of the roles of RE is to identify the needs and requirements of the users, so that the development team can effectively implement the ideal solution [2].

Software Engineering and other activities have more frequently used textual resources to catalog and register the artifacts produced during software development, although there are other methods to enrich the documentation, such as images, audio and video [3]. The resistance to adopt other means to store and present software documentation ends up being detrimental to the construction of systems, given that, for some people, the textual resources may not be sufficient to understand what needs to be developed. Psychology-based research supports this thesis by stating that individuals, in specific contexts, may have preferential input channels for understanding and learning subjects [4].

Given the above, it is necessary to consider, when preparing the software documentation, that people have different ways of perceiving information. This makes all the difference in communication and learning, since, depending on cognitive preferences, some people can understand better listening an audio or watching video interview, than by reading a document, and vice versa [5]. Thus, multimedia resources (video, images and audio) can be a way of offering people different cognitive channels to observe and interpret software documentation, improving interaction of interested parties and understanding of what needs to be developed [6]. In other words, multimedia can bridge the gap between the origins, terminology and education of stakeholders, as well as providing insights closer to reality [7].

In previous studies, it was found that when the requirements were persisted in multimedia format, instead of textual notations, there was an increase of 80% in quality of the requirements artifacts [8]. Although video recordings can prevent analyst's bias, these are still

more objective than relying on written minutes or even on the analyst's personal memory [9]. In summary, regardless of the number of cognitive channels or the order used to stimulate these channels, the possibility of presenting software requirements through various ways allows an increase in the understanding process, allowing developers to choose the preferred order for to communicate and learn [4].

For all these aspects, the objective of this article is to present the result of the construction and experimentation of a multimedia approach coupled with the source code, to support the comprehension and maintenance of software, as an alternative to the traditional techniques of requirements textual documentation. The scope of the approach was constructed taking as a reference the gaps in the tools identified in a systematic mapping, whose objective was to identify and characterize approaches and techniques that promoted the use of multimedia resources in RE.

To automate the approach, a plug-in called CodeMedia was developed, integrated with VisualStudio, which allows the direct linking of code snippets to multimedia resources captured during software development, especially during RE activity. This plug-in was evaluated in an experimental process, from the point of view of efficiency and effectiveness, following the guidelines of [10], [11] and [12]. After performing the experimental evaluation, we found that the multimedia approach obtained more promising results than the traditional development approaches, especially with regard to the number of correct answers in developing tasks, with an average of approximately 4 correct answers per task implemented; and customer satisfaction with the final product, considering aspects such as usability, with a 7% increase in the level of satisfaction.

Regarding the coding time, the time was longer when the approach was used, with an average of approximately 49 minutes on task, while the average time, without the approach, was approximately 34 minutes per task. This increase in coding time can be correlated to several factors, among which, the time required for reproduction of multimedia content and the programming style of the volunteers. The level of code understanding, surprisingly, also decreased when the plug-in was used, making a 16% reduction in the coefficient of code comprehension. It is possible to attribute this result to an incorrect interpretation of the question regarding the level of comprehension of the code, the quality of the multimedia content attached to the tasks or the experience of the developers.

In view of the aforementioned facts, the approach proved to be effective in terms of reducing coding errors in the development of tasks and increasing customer satisfaction with

the final product. In this way, the multimedia approach presents itself as a promising alternative to be combined with the traditional techniques of elicitation and documentation of software requirements, mainly with new developers in the project. In addition, we confirm the results found in the literature, which indicate the use of media as a real option for recording business rules, but now with application in a real development environment and a greater proximity to the code.

For a better understanding of how we obtained the results of the experiment, the work was structured as follows. Section 2 describes the methodology adopted in this work. In section 3, we describe and compare some related works. Section 4 presents the CodeMedia tool and the approach used for its design. In section 5, we find the experimental evaluation. In Section 6, the operation of the experiment is presented. Section 7 contains the results of the experiment. In Section 8, threats to validity are presented. Finally, Section 9 presents the conclusion and future works.

4.2. Methodology

This article is an experimental study that evaluated the performance of a Multimedia Approach for eliciting, registering and reproducing the requirements of a system, using multimedia resources integrated with the source code. In order to answer these questions, the following metrics were used: (i) average time coding tasks, (ii) average hits for each implemented task, (iii) level of code comprehension and (iv) level of customer satisfaction.

In addition to the experimental point of view, this work is also exploratory, since, initially, a systematic mapping of the literature was published in (SANTOS, COLAÇO JÚNIOR, & ANDRADE, 2020), with the goal of finding studies on the use of multimedia resources in the software development and maintenance process, especially in Requirements Engineering (RE). The analysis of these studies guided the construction of the multimedia approach proposed in this work and demonstrated the absence of a rigorous experimental methodology in the published studies. Our experiment consisted of planning, instrumentation, industrial partnership, selection of participants, preparation of the environment, execution, data collection and statistical validation of the results.

In carrying out the experiment, the participants underwent two types of treatment: coding of real maintenance tasks with and without the support of the multimedia approach. The experimental design can be seen in section 4.5.2.6. We apply three statistical tests: Shapiro-

Wilk, Wilcoxon and Test T, to check if there were statistically significant differences between the approaches.

In summary, this work describes an experiment detailed in section 5, with 5 macro steps: planning, experimentation, interviews, data collection and validation.

4.3. Related works

In one of the first steps of our study, we search for related works, through a systematic mapping of the literature. In this context, 69 articles were found and 47 were selected for the data extraction phase. We identified a wide variety of approaches, which promote the use of multimedia resources in activities related to software engineering, especially in RE. Within this subset of articles, considering the works directly related to our experiment and the authors with the largest number of relevant publications in the area, we selected the related works that served as base for the research area addressed in this study.

In [13], a new type of dedicated multimedia storyboard is presented. The product focuses on direct interaction with stakeholders and employs various multimedia technologies to document user requirements. The main objective is to allow interested parties to understand the documentation without the need to have deep knowledge over any software notation. This enables participants to interact and understand requirements. This objective can be divided into:

- Enable the engineer to formulate requirements in a way that the interested party can understand, thus creating a common basis for discussion;
- Allow the interested party to express new requirements; and,
- Capture and document the expressed view as close as possible to the original view.

The views are persisted in a special type of video, which is enhanced by multimedia technologies. In a meeting, the engineer builds these videos in close collaboration with the client: each action is portrayed as a multimedia file (video clip, photo, screen capture, hand drawing, sketch and audio clip). The actions are concatenated in stages, in order to form a so-called VisionVideo. In this way, requirements are documented in an expressive way and using different types of representation. The multimedia requirements can be validated on site, as VisionVideos are watched with the client, allowing him to express feedback and corrections.

[14] proposed to analyze the existing know-how to learn how to build a good video for visual communication, identifying quality features of good videos, to obtain a quality model.

Software professionals can use this quality model as a guide for planning, filming, post-processing and viewing a video. The goal is to encourage and train software professionals to produce good videos at moderate costs, with sufficient quality to store the requirements of a system.

In [15], the video is presented as an alternative to view the requirements and teach social skills. All reviews (with and without the customer) are recorded to provide feedback. Videos are automatically converted and made available on specific web pages, with restricted access to the team. To view project requirements, they use a technique called Video-Based Requirements Engineering.

[16] presented a framework that provides a real-time application to record conversations and convert them into a text transcript. In order to capture the verbal discussion and translate it into a text transcript, they used a real-time audio-to-text conversion software, such as Dragon NaturallySpeaking Premium. Then, a text mining program can describe additional custom words and requirements.

The approach presented here differs from the aforementioned works, especially with regard to integration with the source code. Our multimedia approach proposes that multimedia resources are integrated into the software construction process [11], giving developers different channels of information representation and the ability to evaluate the evolution of software, from the point of view of changes in requirements and the emergence of new ones.

In summary, so far, no studies have been found that have proposed the presentation of multimedia requirements in the way we are proposing, since, unlike the approaches we found, our tool proposes the direct linking of multimedia resources to the source code, in order to support developers in the reproduction, handling and traceability of requirements. Multimedia resources more closely linked to code and software evolution analysis strategies can help answer two important business questions: (1) Where do I see and where is my business in the code? (2) What points have evolved in the code, aligned with the evolution of my business? The answers to these questions can help increase the accuracy of the location of the code to be evolved and changed, increasing maintenance effectiveness, reducing costs and providing the prospect of impact on the source code, based on the evolution of the business.

4.4. CodeMedia Tool

In this section, we present an approach to gain efficiency and effectiveness in the process of comprehension and maintaining software. The multimedia approach proposed in this study adopts multimedia resources for recording, maintaining and presenting user requirements.

The scope of the approach was defined considering the results found in the systematic mapping. A wide variety of solutions for recording, preserving, linking and reviewing requirements using media have been used and tested, as an alternative to traditional requirements elicitation techniques. However, the identified methods presented some limitations, mainly with regard to integration with the source code.

The lack of tight integration with the code prevents a more effective mapping of software entities (such as packages, classes and methods in OOP) and their attributes (metrics, properties and resources) to dynamic medias (audio and video), in a way that can be easily exploited by a developer. Thus, a solution was proposed that explored the limitations of the approaches we found, promoting the integration of multimedia resources with the source code and preserving the existing features, such as the registration, preservation, linking and presentation of the multimedia requirements.

A plugin, called CodeMedia, was developed to implement the approach and work together with VisualStudio. Although CodeMedia was chosen to implement the approach, this does not prevent it from being implemented by another tool. In the next topics, the architecture of the multimedia approach and the CodeMedia documentation will be presented.

4.4.1. Architecture

Software architecture is the structure (or structures) of the system, which is made up of software elements, the externally visible properties of these elements and the relationships between them; it is the abstraction of the system [17]. According to [18], architecture defines what the system is in terms of computational components, the relationships between these components, as well as the standards that guide its compositions and restrictions.

In addition to the choice of algorithms and data structures, the architecture involves: decisions about the structures that will form the system, control, communication protocols, synchronization and access to data, allocation of features to system elements, physical distribution of elements, scalability, performance, selection of design alternatives and other quality attributes [19].

Based on these principles, the architecture of our approach was developed based on the model presented in Figure 1, which shows the main components and their relationships. The model starts with the collection of requirements: any and all information about the system operation (business rules, roles, relationships, input and output parameters, users' stories, behavior of variables, constants, procedures, classes and methods) can be persisted in dynamic medias (audio and video).

In this context, CodeMedia is a multimedia communication channel, with the purpose of helping to understand the code and business rules, without having to leave the development environment.

The plugin was developed in C#, object-oriented programming language developed by Microsoft as part of the .NET platform. As an object-oriented language, C# supports encapsulation, inheritance and polymorphism. All variables and methods, including the Main method, the application's entry point, are encapsulated in class definitions [20].

It is worth noting that the multimedia resources must be linked, at least, to the requirements titles, which may be stored in a Requirements System and/or in a Business Process Management System (BPMS) or not, as long as a system versioning and/or a database list them with identification keys, which will also be stored in CodeMedia, for tracking requirements. Finally, also through the extension, the programmer will be able to view and reproduce the multimedia resources attached to the code.

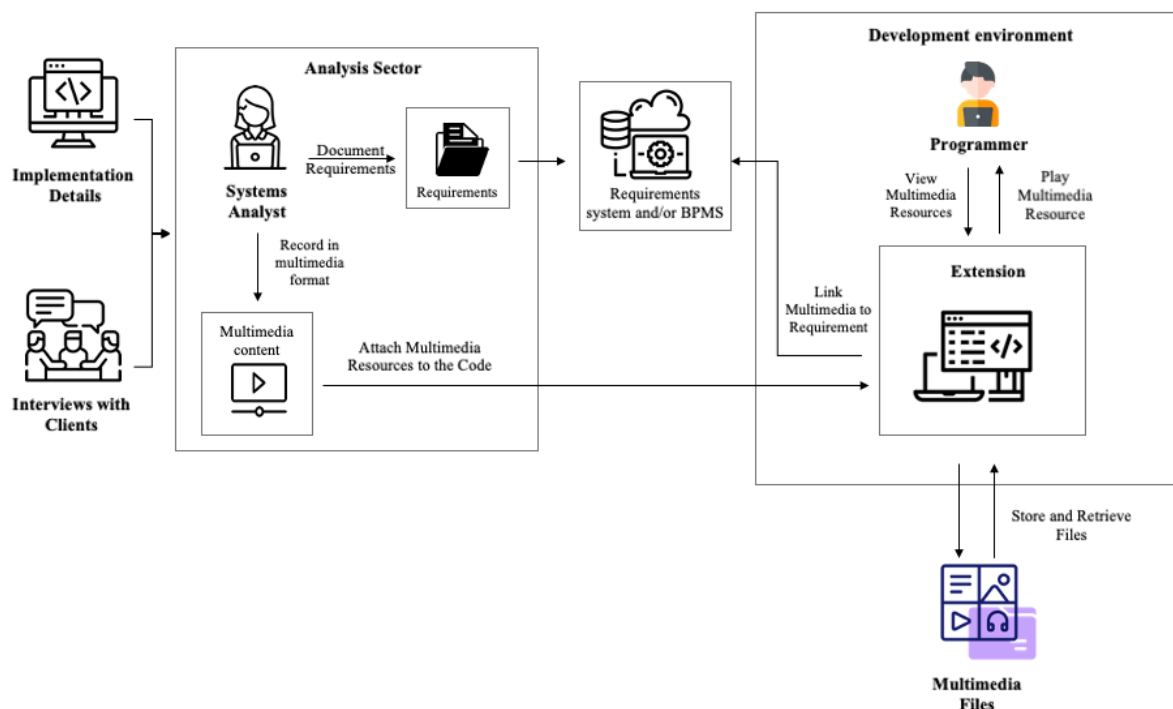


Figure 1. Operating Architecture of the Multimedia Approach.

4.4.2. Features

In the next subsections, the features offered by the extension will be presented.

4.4.2.1. Attach Multimedia Resources to the Code

After registering the requirements in multimedia format, the generated documents must be attached to the project. The files can be attached using two approaches: top-down, associating the files with the project as a whole; and, bottom-up, attaching the file to a specific piece of code in a class. In this sense, the extension allows the same resource to be associated with more than one component and can be used at different points in the project. In addition, the same component may have more than one associated file. Finally, the option to undo associations is also available, allowing the multimedia documentation to be constantly updated.

Figures 2, 3 and 4 represent the sequence of steps necessary to add multimedia documentation to the code. Initially, the developer must select the code snippet to which he wants to attach the files and select the option “Add Documentation” (see Figure 2).

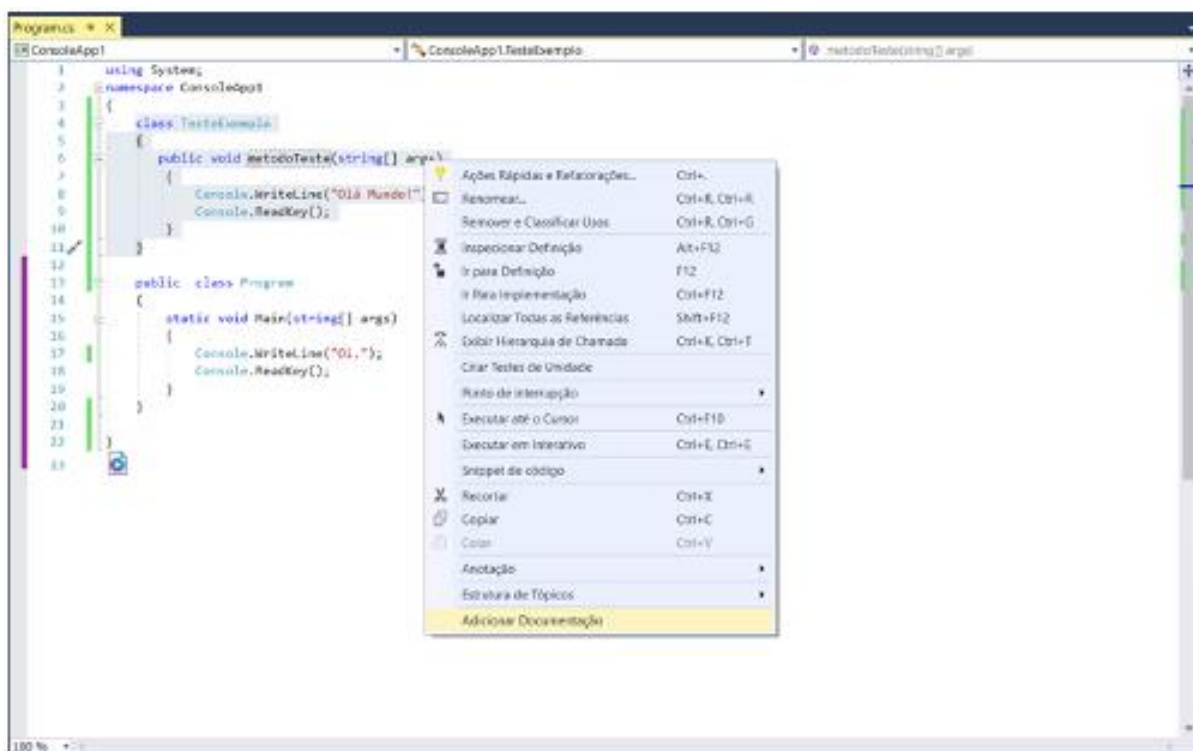


Figure 2. Menu for Adding Documentation.

When selecting the code and choosing the option “Add Documentation”, the CodeMedia window will be displayed in the development environment (see Figure 3).

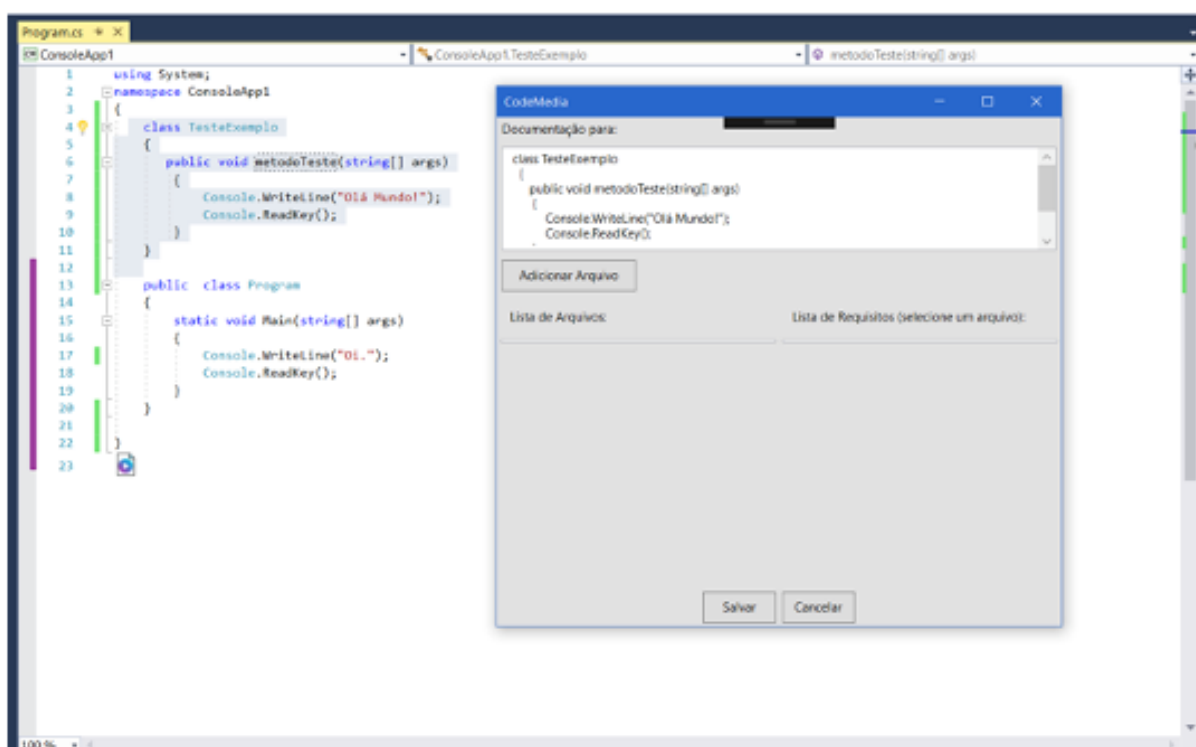


Figure 3. Adding Files.

To add multimedia resources, the developer must select the “Add File” option. In this way, a new window will be displayed (see Figure 4), which allows the addition of new files, using the “Add New” button. The analyst must choose the type of file to be uploaded: Image, Video or Audio; select one or more files in the file explorer and complete the operation by clicking on the “Save” button. At this point, it is important to note that for audio and video files, the user can specify the part of the file that should be played. For example, if an audio has 10 minutes, the user can define the start time and the end time, with only the track of interest being played.

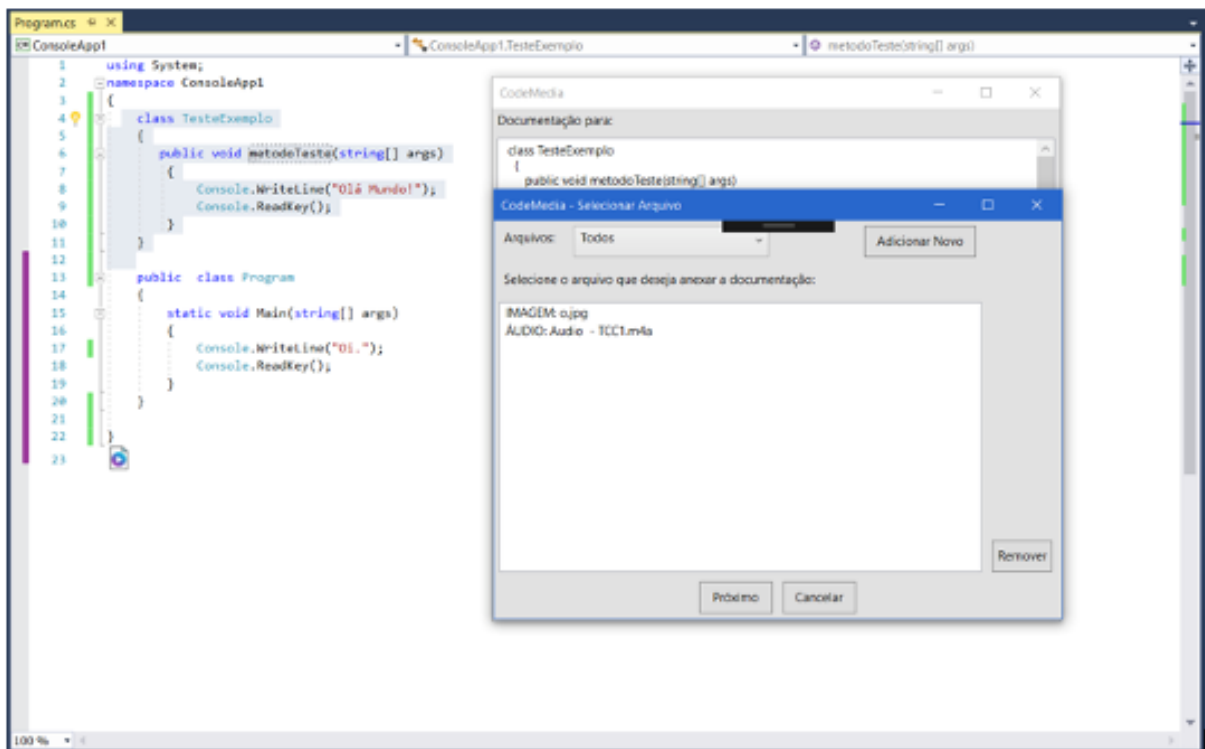


Figure 4. File Selection Screen.

After selecting the files to be attached to the code, the developer must link the files to the software requirements. At least one requirement, from the requirements engineering process, must be informed. The necessary information about the requirement is: id - a unique identifier that distinguishes it from the others on the list, which may come from a Requirements System or BPM System; and the description - a brief description of the requirement, which can also be copied from a system. The linking process can be seen in Figure 5.

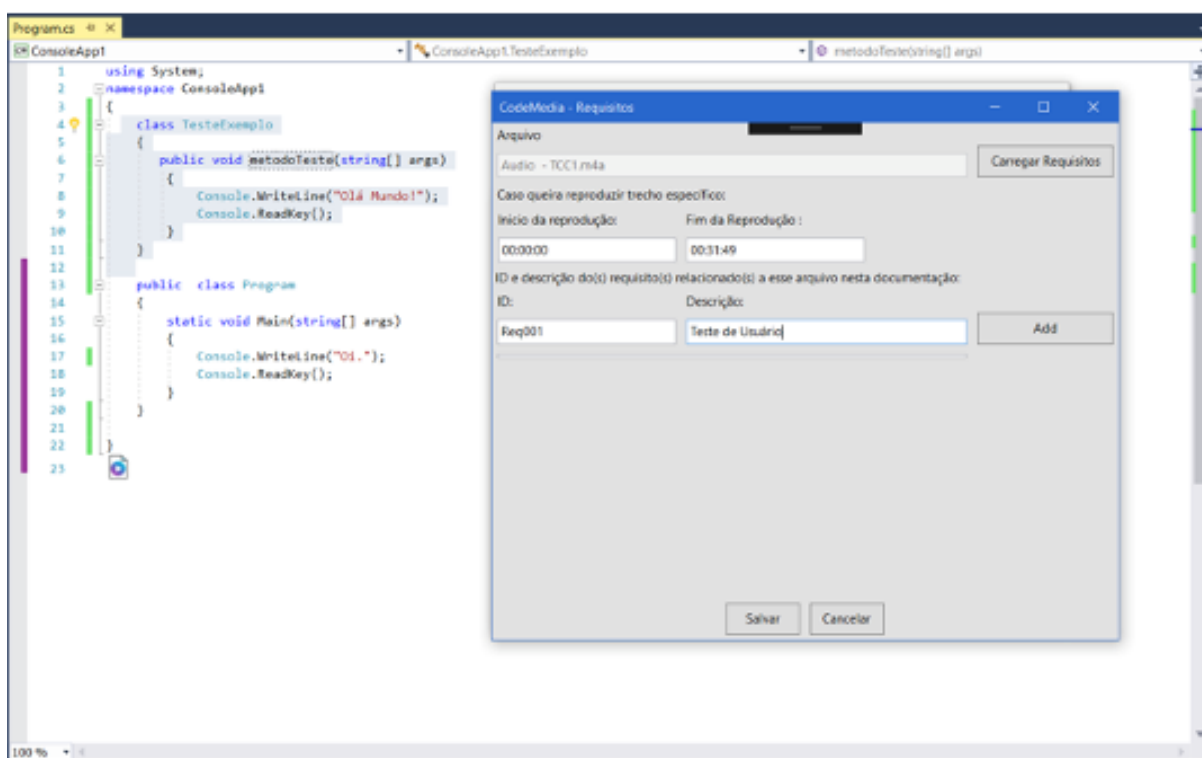


Figure 5. Link File to Requirement.

To complete adding the documentation to the code, the user must click on the “Save” button. In this case, a confirmation message will be displayed and an icon will be added to the code snippet, to which the documentation has been attached (see Figure 6).

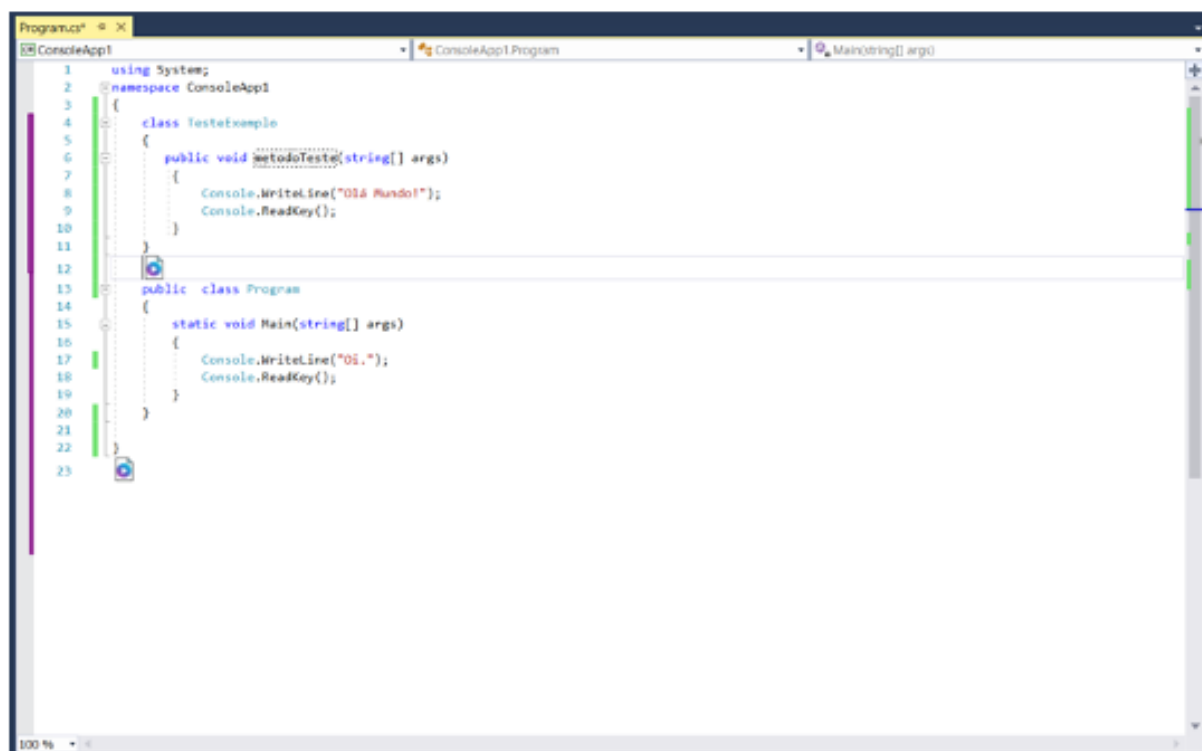


Figure 6. Multimedia Documentation.

4.4.2.2. Reproduction of Multimedia Resources

To view, reproduce, edit or remove the documentation, the user must click on the icon corresponding to the multimedia content. When doing this, a window will be displayed with the files linked to that code snippet, which can be selected and viewed, as shown in Figure 7. The viewing mode will vary depending on the format of the attached file. For audio and video files, the user will be able to play the content, similar to a traditional video player; for images and textual documents, the file will be displayed in a new window.

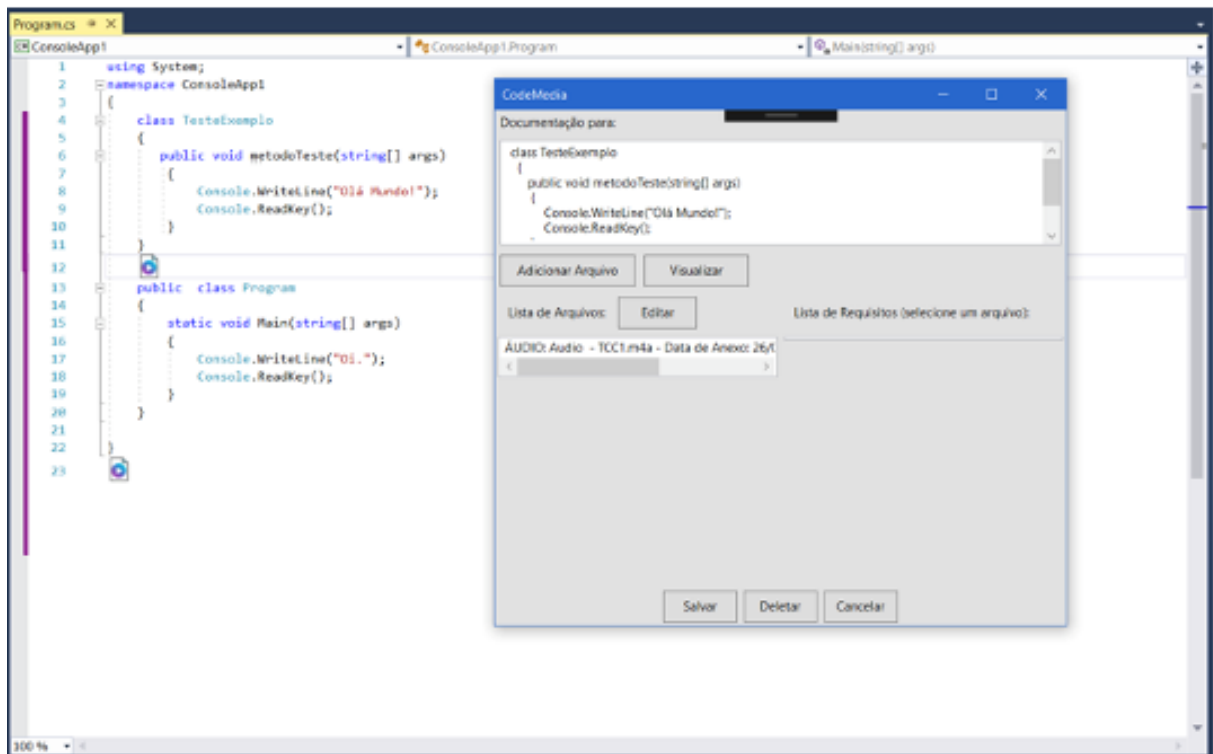


Figure 7. Reproduction and Visualization of Multimedia Resources.

4.4.2.3. Storage and Maintenance of Multimedia Requirements

The data generated by the extension is stored in two formats: JSON (JavaScript Object Notation) and CDOC (DigiDoc Crypted Document).

Files with the .cdoc extension store an object called “Documentations”, which is an array of objects. Each object is composed of 2 other objects, called “Code” and “DataArchives”. The “Code” represents the section of the source code where the documentation was added. It has three keys: “StartPosition” - contains the initial position of the code; “FinalPosition” - contains the final position of the code; and the “Text” - contains the textual description of the source code.

The “DataFile” object is an array of objects with information from the files that have been attached to the code. It consists of two elements: “FileName” - the name and extension of the file; and the “IdPart” - which contains the identifier of the specific segment of the multimedia file attached to the source code. Through “IdPart” it is possible to retrieve the information that is saved in files with the extension .json.

Files with the extension .json store three objects: “FileName” - contains the name and extension of the multimedia file; “FileType” - contains the type of the multimedia file (video, audio and image); and, “Snippets” - stores the attributes of the multimedia file.

The “Snippets” object holds the following information: “Id” - contains a unique segment identifier; “Start” - contains the starting point of the multimedia file playback; “End” - contains the stop point for the playback of the multimedia file; “TotalTime” - contains the original duration of the multimedia file; “Date” - contains the date that this section was created; and finally, “Requirements” - contains an array of objects that identifies the requirements related to the specified stretch. The requirements are identified by the attributes “Id” and “Description”.

When the multimedia file is an image type, the attributes “Start”, “End” and “TotalTime” will have a null value.

4.4.2.4. Folder Structure

The files used by the extension are stored in a folder within the project itself. Each project using the extension will have a folder, in the same directory as the solution, which will be created and managed by the plugin itself. The project directory, with the CodeMedia folder, can be viewed in Figure 8.









| | | |
|---|------------------|------------------------|
|  .vs | 26/03/2019 16:47 | Pasta de arquivos |
|  CodeMedia | 26/03/2019 16:47 | Pasta de arquivos |
|  Model | 26/03/2019 16:47 | Pasta de arquivos |
|  Negocio | 26/03/2019 16:47 | Pasta de arquivos |
|  packages | 26/03/2019 16:48 | Pasta de arquivos |
|  Persistencia | 26/03/2019 16:48 | Pasta de arquivos |
|  SistemaDelivery | 26/03/2019 16:48 | Pasta de arquivos |
|  SistemaDelivery.sln | 25/10/2018 07:44 | Visual Studio Solut... |

Figure 8. Folder Structure.

As persistence parameters, the following information is considered: project identifier; class identifier; file path; media type; description; start and end line of the mark (for code snippets), and the start and end time (capture the specific audio or video track, which should be considered for playback). The parameters may vary depending on the association approach chosen by the analysis team, responsible for feeding the extension with multimedia resources.

With the CodeMedia extension properly presented, the experimental evaluation of the multimedia approach to gain efficiency and effectiveness in the process of comprehension and maintaining the software will be explained in the next chapter.

4.5. Experimental Evaluation

In this section, we will describe the method used to evaluate the multimedia approach proposed in this work, with regard to efficiency gains (when the task is performed in the best possible way, with the least waste of time, effort and resources) and effectiveness (when the task is performed in order to achieve the desired or expected result). The experimental process adopted in this study is based on the guidelines of [10], [11] and [12], considering the necessary adaptations, imposed by pandemic caused by Covid 19 - coronavirus. The next sections will focus on the definition and planning of the experiment, execution and presentation of the experimental results obtained.

4.5.1. Goal Definition

This work aimed to propose and analyze a multimedia approach, to support comprehension and maintenance of software, as an alternative to the traditional techniques of requirements documentation, evaluating whether the use of the CodeMedia plug-in can increase comprehension of source code.

This objective was formalized using the GQM (Goal, Question-Metric) model proposed by [21] and presented by [22]: to analyze a multimedia approach for comprehension and maintaining software, using the plugin CodeMedia, with the purpose of evaluating, regarding the increase of effectiveness in the comprehension and maintenance of software, from the point of view of programmers, in the context of a private information technology company, which develops and maintains software for Public Management.

4.5.2. Planning

In this section, the experimental design will be detailed.

4.5.2.1. Context Selection

The experiment targeted programmers from a private Information Technology company, located in the state of Sergipe, and specialized in the development and maintenance of software for Public Administration.

4.5.2.2. Hypothesis Formulation

The following research questions were elaborated:

- **RQ1** - Can the use of the multimedia approach, using the CodeMedia plugin, reduce programmers' coding time in the software maintenance process?
- **RQ2** - Can the use of the multimedia approach, using the CodeMedia plugin, reduce code errors in software maintenance?
- **RQ3** - Can the use of the multimedia approach, using the CodeMedia plugin, increase the level of software comprehension by programmers in the software maintenance process?
- **RQ4** - Can the use of the multimedia approach, using the CodeMedia plugin, increase the level of customer satisfaction with the developed solution, from the point of view of usability of the task delivered?

In order to assess such issues, the following metrics were used: average time for coding tasks; average of correct answers for each implemented task; level of comprehension of software; and level of customer satisfaction.

Thus, for the first research question listed, the hypothesis that was tested and its respective alternative hypothesis were:

Hypothesis 1

- Null hypothesis H0: Coding for software maintenance, with and without the use of the multimedia approach, has the same efficiency.

$$\mu(\text{timeCodingWithMultimediaApproach}) = \mu(\text{timeCodingWithoutMultimediaApproach}).$$

- Alternative hypothesis H1: Coding for software maintenance, using the multimedia approach, is more efficient than coding performed without using the approach. $\mu(\text{timeCodingWithMultimediaApproach}) < \mu(\text{timeCodingWithoutMultimediaApproach})$.

For the second research question listed, the hypothesis that was tested and its respective alternative hypothesis were:

Hypothesis 2

- Null hypothesis H0: Coding for software maintenance, with and without the use of the multimedia approach, has the same effectiveness. $\mu(\text{averageHitsCodingWithMultimediaApproach}) = \mu(\text{averageHitsCodingWithoutMultimediaApproach})$.
- Alternative hypothesis H1: Coding for software maintenance, using the multimedia approach, is more effective than coding performed without using the approach. $\mu(\text{averageHitsCodingWithMultimediaApproach}) > \mu(\text{averageHitsCodingWithoutMultimediaApproach})$.

For the third research question listed, the hypothesis that was tested and its respective alternative hypothesis were:

Hypothesis 3

- Null hypothesis H0: The level of software comprehension of the developers and the use of the multimedia approach are not correlated.
- $r = 0$
- Alternative hypothesis H1: The level of software comprehension of the developers and the use of the multimedia approach have a correlation.
- $r \neq 0$

For the fourth research question listed, the hypothesis that was tested and its respective alternative hypothesis were:

Hypothesis 4

- Null hypothesis H0: The level of customer satisfaction with the delivered solution is not correlated with the use of the multimedia approach.
- $r = 0$

- Alternative hypothesis H1: The level of customer satisfaction with the delivered solution is correlated with the use of the multimedia approach.
- $r \neq 0$

It is important to emphasize that the null hypotheses (H0) are the hypotheses that were intended to be refuted and the alternative hypotheses (H1) are those that, within the context of the experiment, would not be rejected.

4.5.2.3. Variables Selection

In order to investigate the phenomenon in question, we considered the following variable:

- **Dependent Variables:** maintenance time with and without the adoption of the multimedia approach; level of comprehension of the code (grade given by programmers for tasks performed with and without using the approach); average of correct answers for each completed task; and level of customer satisfaction (grade given by the customer to tasks implemented with and without using the approach);
- **Independent Variables:** the object of the experiment (in this particular case, the multimedia approach defined in this work for comprehension and maintaining the software, using the CodeMedia plug-in); the activities that were developed and their complexities, described in Section 4.5.2.4; the quality of the multimedia content attached to the tasks, containing the task specification; and the professional experience of the participants.

4.5.2.4. Description of Tasks in the Experiment

The tasks were grouped into 4 different lists (see Table 1). The groups were built in order to facilitate segmentation and visualization of the tasks that would be made with and without the support of the multimedia approach.

Table 1. Software Maintenance Tasks.

| ID | Description | List | Media Type | Content Type | Module |
|----|--|------|------------|------------------------------------|-----------------|
| 1 | Allow printing financial files in the employee module. | 1 | Audio | Audio Requirements, Business Rule, | Human Resources |

| | | | | | |
|---|---|---|-------------------------------------|--|--------------------------|
| | | | | Implementation details. | |
| 2 | Link to the section on the Asset Transfer Screen. | 1 | Audio | Audio Requirements, Business Rule, Implementation details. | Patrimony |
| 3 | Create Accountability Tab on the Agreement Screen. | 1 | Video | Video Requirements, Business Rule, Implementation details. | Contracts and Agreements |
| 4 | Indicate on the Expense Request screen whether or not it will reserve Budget Balance. | 1 | Video | Video Requirements, Business Rule, Implementation details. | Bidding |
| 5 | Visual and functional adjustments in the Group Launch Registration Screen. | 2 | No multimedia content was attached. | No multimedia content was attached. | Human Resources |
| 6 | Create Summary Report by Type of Payroll in the Human Resources module. | 2 | No multimedia content was attached. | No multimedia content was attached. | Human Resources |
| 7 | Correct inconsistency in the Clearance Screen. The printer button is not working. | 2 | No multimedia content was attached. | No multimedia content was attached. | Human Resources |
| 8 | Add Legal Basis filter in the Annual Statement of Commitments. | 2 | No multimedia content was attached. | No multimedia content was attached. | Accounting |
| 9 | In the Online City, change the way the name of the entity is displayed. | 3 | No multimedia content was attached. | No multimedia content was attached. | Online City |

| | | | | | |
|----|---|---|-------------------------------------|--|--------------------------|
| 10 | Create modality filter in the Report of Minutes / Request for Commitment. | 3 | No multimedia content was attached. | No multimedia content was attached. | Bidding |
| 11 | Add filters: period, tax amount range, and sale amount range in the Property Transfer Report. | 3 | No multimedia content was attached. | No multimedia content was attached. | Tributes |
| 12 | Correct inconsistency after generating the second bank remittance for different paying accounts. | 3 | No multimedia content was attached. | No multimedia content was attached. | Human Resources |
| 13 | Limit the term of COVID-19 legally based contracts to a maximum of 6 months. | 4 | Audio | Requirements, Business Rule, Implementation Details. | Contracts and Agreements |
| 14 | Include a lock for when the server is going to launch maternity leave, do not allow more than 120 days to not be deducted in the Social Security Guide. | 4 | Audio | Requirements, Business Rule, Implementation Details. | Human Resources |
| 15 | Inserting the invoice number on the Summary of Engagement screen. | 4 | Vídeo | Requirements, Business Rule, Implementation details. | Accounting |
| 16 | Create limiter in the overtime event. When selecting the Court of Auditors link (overtime), a field should appear to limit the maximum overtime. | 4 | Audio | Requirements, Business Rule, Implementation Details. | Human Resources |

4.5.2.5. Selection of Participants and Objects

Once the hypotheses and variables to be analyzed were defined, the process of selecting participants and objects began. Initially, aiming at a result that was as close as possible to the day-to-day life of a programmer, it was decided to apply the experiment within a real

software maintenance context, with real problems and customers. Thus, programmers from a software company in Sergipe were invited to participate in this experiment, representing the population to carry out this study. The goal, despite the rare conveniences of access and partnership with a company in the market, has always been sampling by quota, in which the features of the population of interest are preserved.

A formal request was made to the company's board, questioning whether the experiment could be applied. After a positive response, the developers were invited to voluntarily participate in the experimental evaluation. From a total of 15 developers, 8 showed interest in participating in the referred study, which were characterized, considering the following aspects: age, gender, professional training, preferred learning mode (involving the auditory, visual and kinesthetic systems), profession/position, area of expertise (development, support and infrastructure), years of experience in the language used in the experiment, experience in the number of systems already maintained and known technologies. The tabulation of the results obtained with the characterization of the participants can be seen in the graphs.

With regard to age, 75% of the sample was aged between 19 and 40 years. Only 25% of the sample was aged between 41 and 50 years. None of the volunteers was over 50 years old.

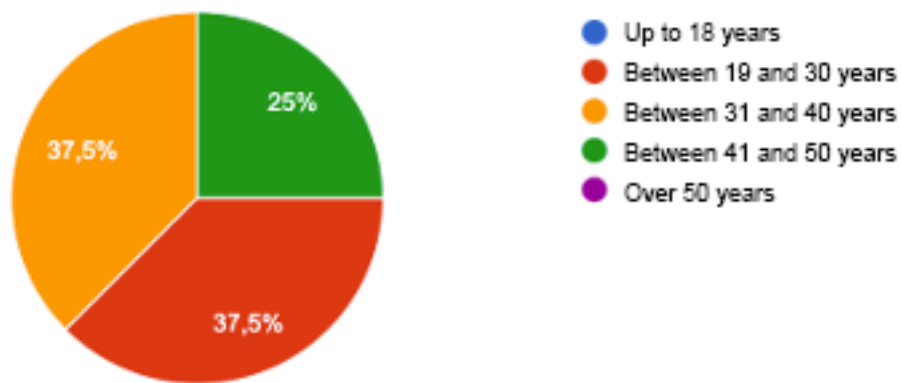


Figure 9. Age.

Regarding gender, there was unanimity, all volunteers declared themselves to be male.

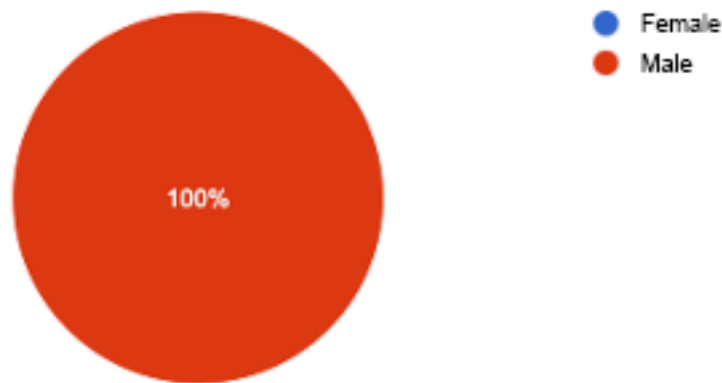


Figure 10. Gender.

With regard to academic training, 75% of the sample had a degree in higher education. 25% had a technical course or specialization.

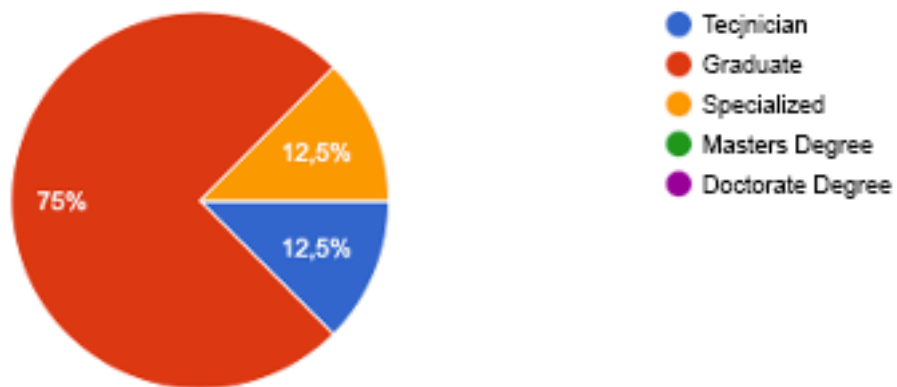


Figure 11. Academic Training

Participants were asked to indicate the order of preference, considering the preferred learning mode to absorb and interpret information. Visual and Kinesthetic were tied for first in order of preference of the programmers. Second, it appears visual; and third, the ear canal.

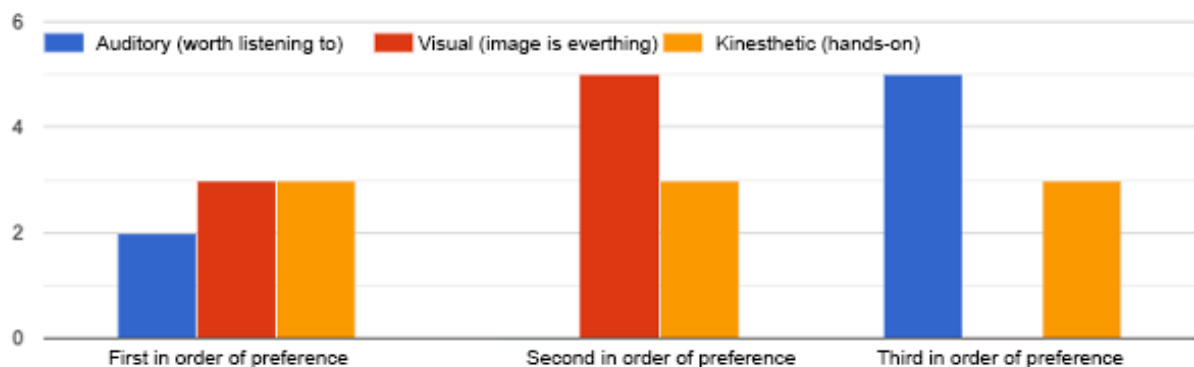


Figure 12. Preferred Learning Mode.

Developers were also asked if they used any techniques for documenting software requirements. 87.5% answered that they did not; only 12.5% cited the use of software tests and comments to record and manage requirements.

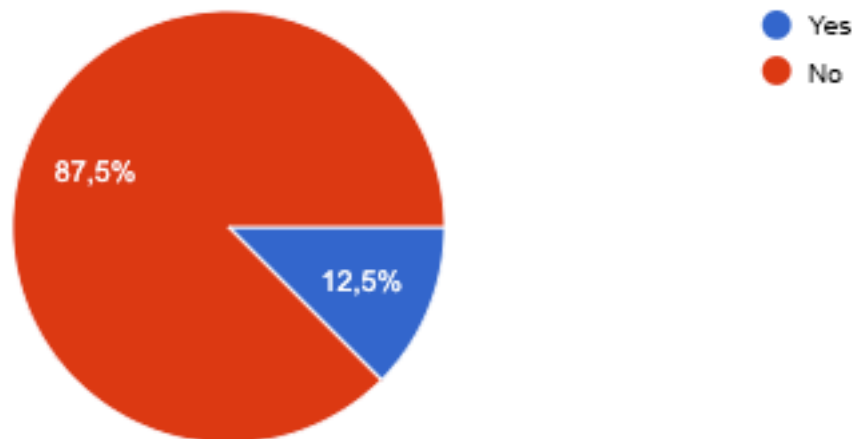


Figure 13. Methods for Requirements Documentation.

Regarding the profession/position, 63% declared themselves as programmers, while 37% called themselves as systems analysts. It is important to note that although there was a discrepancy in the name of the positions, the participants perform the same functions within the institution, where the experiment was applied.

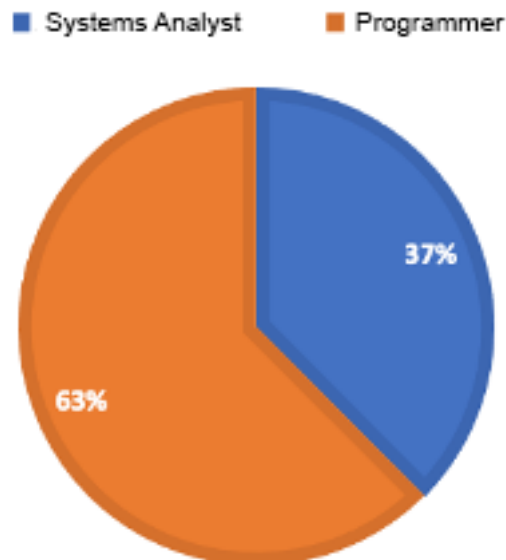


Figure 14. Profession/Position.

Regarding the area of operation, 75% had only worked with development and 25% had already worked in the areas of development, infrastructure and support.

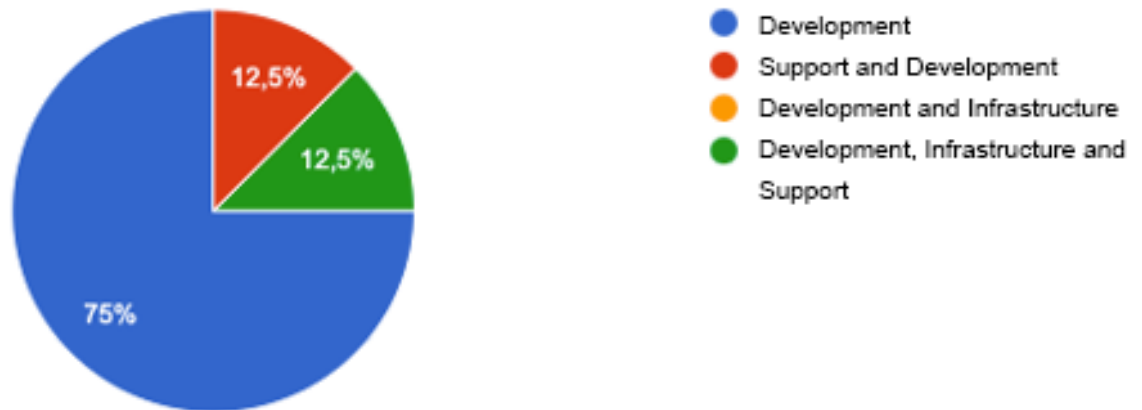


Figure 15. Area of Operation.

With regard to years of experience in the language used in the experiment, 75% of the volunteers had more than 10 years of experience. 25% had up to 3 years of experience.

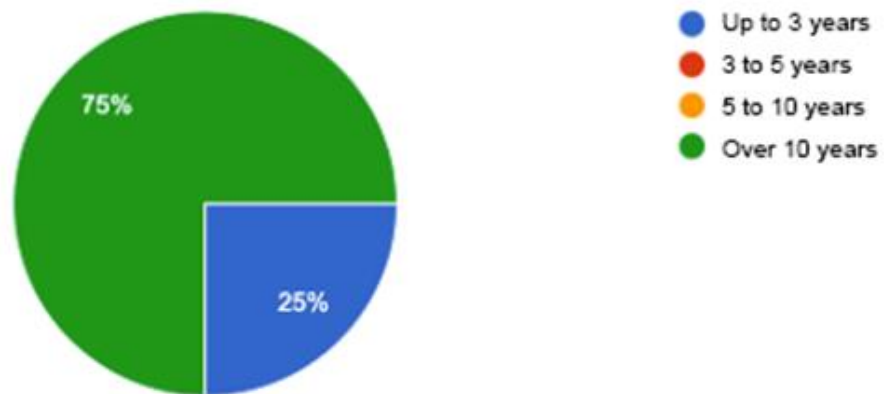


Figure 16. Years of experience in the language used in the experiment.

Regarding the number of systems which they are responsible for maintaining, 50% of the developers had already maintained between 6 to 10 systems; 25%, between 1 to 5; and, 25% had already maintained between 11 to 20 software or more.

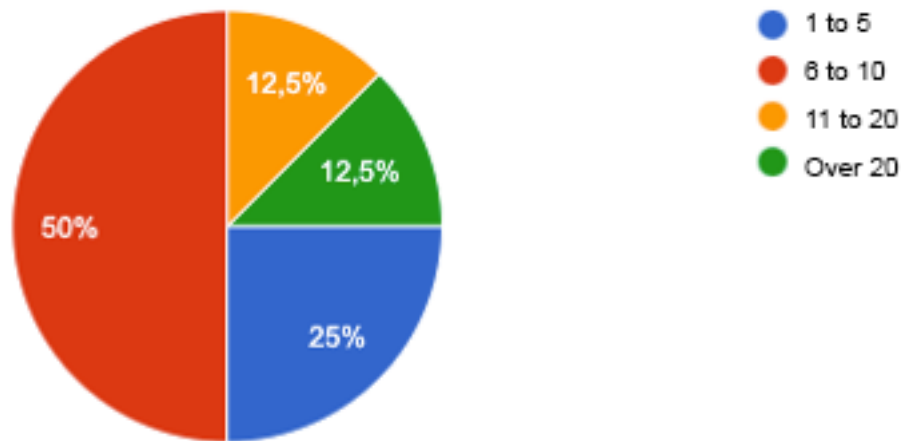


Figure 17. Number of Systems.

Regarding the number of known technologies, the developers demonstrated to know a wide variety of tools. In order of citation, follow the technologies mentioned by the volunteers: C #, Java, PHP, Python, Ruby, Typescript, Swift, C ++, JavaScript, Delphi, vb.net, Pascal, Scala, Lua, C and SQL.

4.5.2.6. Experimental Design

Taking as a reference the experimental design adopted by [23], the experiment was designed in a paired context, in which each group evaluated both approaches: coding with and without the support of the multimedia approach.

The experiment was performed in two days, with a daily duration of 4 hours, an interval of 10 minutes, totaling a total of 8 hours and 20 minutes. Four lists with maintenance tasks were used, totaling 16 tasks, comparable in terms of complexity.

Participants were divided into two groups which assessed both approaches. The division was made randomly, in which each group was left with 4 participants. On the first day, group 1 started the experiment, with option 1 followed by option 2. On the second day, group 2 continued the experimental process, starting with option 2 and ending with option 1. The inversion of options considered tiredness and boredom not to favor one of the two treatments. If one of these were always the last task, the participant's tiredness may influence the results. The options were:

Option 1: Coding of list 1 tasks, with support of the multimedia approach, and, soon after, coding of list 2 tasks, without the support of the approach; and,

Option 2: Coding of tasks in list 3, without the support of the multimedia approach and, soon after, coding of tasks in list 4, with the aid of the approach.

It is important to note that, before the experiment was carried out, the participants had to complete the characterization form. Then, a 30-minute training session was held to teach volunteers how to use the CodeMedia plugin. This training was carried out by someone without involvement in the research. Also, participants were not allowed to interact with each other during the application of the experiment.

Before starting a task, programmers were instructed to move the corresponding card to the “Doing” column. Then, the developer started the Activity timer, and the execution time started to be counted. After the completion of each task, the programmers informed that they had finished the task, stopped the timer and dragged the card to "In Validation" column. The test team, through functional tests and code review, were always evaluating the tasks according to the specification and requirements. Tasks which did not comply with the specification were “Failed” and subsequently counted as errors. The tasks which were in accordance with the specification were “Approved” and subsequently counted as hits. Tasks that were consistent with the specification, but that had reservations about implementation, were “Approved with remarks”.

To avoid errors in time count, in addition to the use of the stopwatch, a comparison was made with the times recorded by Trello, checking the date and time of each movement within the platform. With this verification, it was possible to monitor the time spent by each volunteer in executing the task and compare it with the time recorded using the stopwatch.

As the experiment was applied in a remote context, a compilation of the original Kanban method, proposed by [24], was used to manage the flow of software development tasks. The use of this model allowed visualization of the workflow during the execution of the experiment, with the creation of columns to illustrate where each task was in the workflow, facilitating the monitoring and control of the tasks.

For each completed task, programmers had to answer a question to assess the level of comprehension of the code. The time taken by the developers to answer the question was not counted as programming time. The question can be viewed in <https://docs.google.com/forms/d/e/1FAIpQLSdvUkeZwWWZ30yN63J53F5qvwvQFX2TQaA1u5pBy7pLNa6CZQ/viewform>. The completed tasks were also submitted to the final customer's evaluation. A questionnaire was used to assess the level of customer satisfaction with the solutions delivered by the programmers, with and without the use of the approach. The

questionnaire was applied individually to each task made and can be viewed in https://docs.google.com/forms/d/e/1FAIpQLScYjAjrqvI88N8A1IfV7O1UCzsql_fbpZ_uRAV1sLA17tO2yg/viewform. At the end of the experiment, the programmers had to answer the Questionnaire for Level of Comprehension of Software. Unlike the question on comprehension code, this questionnaire was applied within a general context, considering all the tasks performed. The questionnaire is available in https://docs.google.com/forms/d/e/1FAIpQLSdF3iq7W_7bHGXkfIV0i3srTHMAKI4i6-OBO83R7kmzO_kcrQ/viewform.

Finally, from the results that were collected, it was possible to assess whether the experiment design could identify evidence of the use of the multimedia approach as a facilitator in the process of comprehension and maintaining software.

4.5.2.7. Instrumentation

The instrumentation process took place, initially, with environment setting to carry out the experiment remotely, considering the home office regime in which the company was during the pandemic period.

The resources used were:

- Visual Studio (Community, Enterprise, Pro) versions 15.0 to 19.0: Microsoft's integrated development environment for software development specially dedicated to the .NET Framework and Visual Basic, C, C ++, C # and F # languages;
- CodeMedia plugin: extension for Visual Studio, which allows the indexing of multimedia files (audio, videos and images) to the code. The creation of this extension was necessary to implement the approach proposed in this work;
- Indexing of Multimedia Content to the code: The multimedia documentation containing the specification of the task was attached to the code through the CodeMedia extension;
- List of tasks listed in section 4.5.2.4;
- Trello: web-based collaboration tool that allows you to manage projects by grouping tasks, also called cards;

- Power Up Activity: stopwatch integrated with Trello, which was used to account for the time spent performing each task. In addition to the use of this resource, a comparison was made with the times recorded by Trello, in order to mitigate the threat of incorrect accounting of the time used to carry out the tasks;
- Zoom: remote conferencing service that combines video conferencing, online meetings, chat and mobile collaboration;
- Sample Characterization Form: a questionnaire was designed to define the profile of the participants, considering the following aspects: age, gender, professional training, preferred learning mode (auditory, visual and kinesthetic), profession / position, area of activity (development, support and infrastructure), years of experience in the language used in the experiment, number of systems maintained and known technologies. The form is available in https://docs.google.com/forms/d/e/1FAIpQLSdJivfjvJQLIGv5iXiAUqvrHu31B1mq_xSzHUGuCi8hIItWAw/formResponse;
- Question for Assessing the Level of Comprehension of the Code: a question was prepared to assess the level of comprehension of the code by the programmers, for each task performed. The question can be seen in <https://docs.google.com/forms/d/e/1FAIpQLSdvUkeZwWWZ30yN63J53F5qvwwQFX2TQaA1u5pBy7pLNa6CZQ/viewform>;
- Customer Satisfaction Level Assessment Questionnaire: a questionnaire was created to assess the level of customer satisfaction with the solutions delivered, considering usability. The form is available in https://docs.google.com/forms/d/e/1FAIpQLScYjAjrql88N8A1Ifv7O1UCzsql_fbpZ_uRAV1sLA17tO2yg/viewform; and,
- Software Comprehension Level Assessment Questionnaire: a questionnaire was developed to assess the level of comprehension of the participants when using or not using the multimedia approach proposed in this study. The form is available in

https://docs.google.com/forms/d/e/1FAIpQLSdF3iq7W_7bHGXkfIV0i3srTHMAKI4i6-OBO83R7kmzO_kcrQ/viewform.

The Software Comprehension Level and Customer Satisfaction Level questionnaires were created using the TAM (Technology Acceptance Model) model, proposed in [25].

4.6. Experiment Steps

This section describes each step of the experiment.

4.6.1. Preparation

The following preparation steps were carried out for the execution of the experiment:

1. Separation of tasks: We divided 16 tasks about real software maintenance problems comparable in terms of complexity;
2. Preparation of multimedia content: The analysis sector prepared the task specification from List 1 and 4, containing the software requirements linked to the task, business rules and implementation details and the content was persisted in multimedia content (audio or video). For tasks belonging to lists 2 and 3, the specification was presented in a traditional manner (through textual documentation). For all tasks, a brief introduction was made to the programmers to say what each task was about, as in a real work day. The conversation was limited to what happens in everyday life, to avoid any impact on the result of the experiment. The following is an example of multimedia content attached to the tasks, which were made using the CodeMedia extension:
https://drive.google.com/file/d/1qr63qhcyAzom0H_GIsgLGXxf0oRthxj/view?usp=sharing;
3. Accommodation of the participants: on a day and time previously scheduled, the programmers were invited to participate in a call, through the Zoom video conference tool, in which the guidelines of the experiment were explained, however, without mentioning the central objective of the experimental evaluation, which was the evaluation of the effectiveness of the multimedia approach in the process of comprehension and maintaining

software. After this introduction, the sample characterization form was made available to the participants, which also did not explain the final objective of the experimental evaluation;

4. CodeMedia extension training: as explained earlier, the multimedia approach proposed in this work was automated by the CodeMedia extension, developed just for this purpose. In order to reduce the impact of the extension in the experimental evaluation of the approach, it was decided to conduct a plug-in training, lasting 30 minutes, for the programmers to become familiar with the tool;
5. Group formation: the groups were defined according to the experimental design presented in Section 4.5.2.6; and,
6. Introduction to Trello: after defining the groups, participants were invited to join Trello, and to become members on the "Experiment" board. In this board, they visualized the tasks that would be developed during the experimental evaluation. The tasks were already divided into lists, as specified in topic 4.5.2.4. The programmers were also instructed on how to move the cards as they went about executing the tasks. It was not necessary to carry out specific training to explain the use of Trello, because it is daily used by developers.

4.6.2. Execution

After performing the steps already described, the experiment was started according to the design presented in Section 4.5.2.6.

4.6.2.1. Data Collection

During the experiment, the programmers answered the question for each task performed to assess the level of comprehension of the code. The final customer, in the role of the product owner, answered the questionnaire on the level of customer satisfaction, also for each task, considering usability aspects. The time taken by programmers to complete the code comprehension question was not counted as programming time.

At the end of the experiment, the developers were subjected to some questions, in which they had to answer the questionnaire to assess the level of software understanding, within a general context, considering all the tasks performed. In addition, the time spent executing each task, captured and stored using the Activity timer, as well as the approved and disapproved tasks, were recorded in Trello, for later analysis and accounting.

The results of this experimental evaluation, obtained through the collected data and their analysis and validation, will be presented, later, in the topic Results and Discussion.

4.6.3. Data Validation

For the experiment, a factor was considered - approach used to codify the tasks, with and without the support of the CodeMedia plugin. In this context, the grades attributed by the programmers to the comprehension of the code, grades attributed by the client to the tasks delivered, the number of tasks performed correctly, together with the time spent to perform each task, were collected through the use of questionnaires and tools time capture.

To assist in the analysis, interpretation and validation of the results, strength analysis of the correlations and three types of statistical tests were used: Shapiro-Wilk, Wilcoxon and Test T. The first verify that the collected data did not have a normal distribution. In view of the aforementioned fact, the Wilcoxon non-parametric test was applied to compare the time spent (in minutes) to perform the tasks and the average of correct answers (tasks performed according to specification) considering the two scenarios, with and without the use of the multimedia approach. The T test was used to assess the correlation hypotheses.

4.7. Results and Discussion

To answer the survey questions listed, the following dependent variables were analyzed: time in minutes spent to perform each task; average of correct answers in the execution of tasks; grade assigned by the programmer for comprehension the task code; and, average of the grade attributed by the client to the tasks made available for use.

For the first research question listed (RQ1), the Shapiro-Wilk test was applied to verify the normality of the data collected. Table 2 shows the time spent by each developer to execute the tasks without the support of the multimedia approach. Time was counted in hours and then converted into minutes. The sample can be seen in the table 2:

Table 2. Time spent to execute tasks without the support of the multimedia approach.

| Task | Group | Time in minutes | Time | Use (2) or non-use of the approach (1) |
|---|--------------|------------------------|-------------|---|
| Create modality filter in the Report of Minutes / Request for Commitment. | 1 | 39 | 00:39:00 | 1 |
| Visual and functional adjustments to the Group Launch Registration Screen. | 1 | 73 | 01:13:00 | 1 |
| Correct inconsistency in the Withdrawal Screen. The printer button is not working. | 1 | 20 | 00:20:00 | 1 |
| Correct inconsistency after generating the second bank remittance for different paying accounts. | 1 | 65 | 01:05:00 | 1 |
| Include Legal Basis filter in the Annual Statement of Commitments. | 1 | 10 | 00:10:00 | 1 |
| Include filters: period, tax amount range, and sale amount range in the Property Transfer Report. | 1 | 55 | 00:55:00 | 1 |
| In the Human Resources module, create Summary Report by Type of Sheet. | 1 | 92 | 01:32:00 | 1 |
| In the City Online, change the way the name of the entity is displayed. | 1 | 9 | 00:09:00 | 1 |
| Visual and functional adjustments to the Group Launch Registration Screen. | 2 | 16 | 00:16:00 | 1 |
| Correct inconsistency after generating the second bank remittance for different paying accounts. | 2 | 9 | 00:09:00 | 1 |
| Include Legal Basis filter in the Annual Statement of Commitments. | 2 | 7 | 00:07:00 | 1 |
| Include period filters, tax amount range, and sale amount range in the Property Transfer Report. | 2 | 17 | 00:17:00 | 1 |

| | | | | |
|--|---|----|----------|---|
| Correct inconsistency in the Withdrawal Screen. The printer button is not working. | 2 | 30 | 00:30:00 | 1 |
| Create modality filter in the Report of Minutes / Request for Commitment. | 2 | 5 | 00:05:00 | 1 |
| In the Municipality Online, change the way the name of the entity is displayed. | 2 | 7 | 00:07:00 | 1 |
| In the Human Resources module, create Summary Report by Type of Sheet. | 2 | 92 | 01:32:00 | 1 |

After applying the Shapiro-Wilk test to sample 1, the values of $p = 0.00614967$, the mean = 34.125000 and the median = 18.5 were obtained. As $p < 0.05$, it is assumed that the data in sample 1 is not normally distributed, and it is not possible to apply parametric tests.

After the non-normality of sample 1 was verified, the same statistical test was applied to sample 2, which brings the time spent by each developer to execute the tasks with the support of the multimedia approach. The sample can be seen in the table 3:

Table 3. Time spent to execute tasks supported by the multimedia approach.

| Task | Group | Time in minutes | Time | Use (2) or non-use of the approach (1) |
|---|-------|-----------------|----------|--|
| Create limiter in the overtime event. When selecting the Court of Auditors link (overtime), a field should appear to limit the maximum overtime. | 1 | 66 | 01:06:00 | 2 |
| Allow printing financial files by registering the server. | 1 | 46 | 00:46:00 | 2 |
| Indicate on the Expense Request screen whether or not it will reserve Budget Balance. | 1 | 87 | 01:27:00 | 2 |
| Include a lock for when the server is going to launch maternity leave, do not allow more than 120 days to not be deducted in the Social Security Guide. | 1 | 44 | 00:44:00 | 2 |
| Create Accountability Tab on the Agreement Screen. | 1 | 62 | 01:02:00 | 2 |

| | | | | |
|---|---|-----|----------|---|
| Inserting the invoice number on the Summary of Engagement screen. | 1 | 30 | 00:30:00 | 2 |
| On the Asset Transfer Screen, link only to the sector. | 1 | 30 | 00:30:00 | 2 |
| Limit the term of COVID-19 legally based contracts to a maximum of 6 months. | 1 | 11 | 00:11:00 | 2 |
| Allow printing financial files by registering the server. | 2 | 81 | 01:21:00 | 2 |
| Create limiter in the overtime event. When selecting the Court of Auditors link (overtime), a field should appear to limit the maximum overtime. | 2 | 104 | 01:44:00 | 2 |
| Include a lock for when the server is going to launch maternity leave, do not allow more than 120 days to not be deducted in the Social Security Guide. | 2 | 47 | 00:47:00 | 2 |
| In the Patrimony Transfer Screen, link only to the sector. | 2 | 39 | 00:39:00 | 2 |
| Indicate on the Expense Request screen whether or not it will reserve Budget balance. | 2 | 14 | 00:14:00 | 2 |
| Limit the term of COVID-19 legally based contracts to a maximum of 6 months. | 2 | 18 | 00:18:00 | 2 |
| Inserting the invoice number on the Summary of Engagement screen. | 2 | 52 | 00:52:00 | 2 |
| Create Accountability Tab on the Agreement Screen. | 2 | 58 | 00:58:00 | 2 |

After applying the Shapiro-Wilk test to sample 2, the values of $p = 0.797524$, the mean = 49.312500 and the median = 46.5 were obtained. As $p > 0.05$, it is assumed that the data in sample 2 is normally distributed. However, as normality failed when verified for sample 1, normality was disregarded for both samples. In view of this scenario, in order to validate the listed hypotheses, it was decided to use the non-parametric test, Wilcoxon.

Analyzing the results, it was found that when the multimedia approach was used, the coding time was slightly longer compared to the time without using the approach. However, through the application of the Wilcoxon statistical test, it was found that there was no statistical

significance to reject the hypothesis of equality between the two treatments. In other words, the efficiencies with and without the approach are the same. The p-value is equal to 0.910631. This means that the chance of error, rejecting the null hypothesis (H_0), is very high: 0.9106 (91.06%). The higher the p value, the more it supports H_0 . Thus, H_0 is not rejected. Therefore, coding for software maintenance, with and without the use of the multimedia approach, has the same efficiency.

Among the factors that may have contributed to the increase in time when the approach was used, it is possible to mention the programming style presented by each participant. Some had a more direct style, focusing on exactly what the task required, which significantly reduced the coding time. While others carried out a deeper analysis of the task, going beyond what the task need. This ended up generating code refactoring and, consequently, increasing the task resolution time. At this point, it is worth highlighting one more advantage of using multimedia in ES, that is, the audio or video explanation of the code may have evidenced a technical debt, positively stimulating refactoring. Finally, another point that may have influenced the longer time spent in solving tasks, when the CodeMedia plugin was used, was the time needed to reproduce and understand the multimedia content attached to the code.

For the second research question listed (RQ2), the Shapiro-Wilk test was also applied to verify the normality of the data samples. Sample 3 shows the average of correct answers for each task implemented without the support of the multimedia approach. “Failed” tasks received a score of zero; for “Approved” tasks, the average of the grades assigned by the client was used (the average, for each task performed with and without the tool), for “Approved with Disclaimers” tasks, the average of the grades assigned by the client was also used, however, for each negative exception, one point was discounted from the general average. Sample 3 can be seen in the table 4:

Table 4. Average hits for each task implemented without the multimedia approach support.

| Task | Group | Validation (Testing Team) | Average hits | Use (2) or non-use of the approach (1) | Average customer grade |
|--|-------|---------------------------------|-----------------|--|------------------------------|
| Create modality filter in the Report of Minutes / Request for Commitment. | 1 | Approved | 5 | 1 | 5 |
| Visual and functional adjustments to the Group Launch Registration Screen. | 1 | Approved | 5 | 1 | 5 |

| | | | | | |
|--|---|-----------------------|------|---|----------------------------------|
| Correct inconsistency in the Clearance Screen. The printer button is not working. | 1 | Approved with remarks | 3,33 | 1 | 4,333333333 |
| Correct inconsistency after generating the second bank remittance for different paying accounts. | 1 | Approved with remarks | 4 | 1 | 5 |
| Include Legal Basis filter in the Annual Statement of Commitments. | 1 | Approved | 5 | 1 | 5 |
| Include period filters, tax amount range, and sale amount range in the Property Transfer Report. | 1 | Failed | 0 | 1 | Not sent for customer validation |
| In the Human Resources module, create Summary Report by Type of Sheet. | 1 | Approved with remarks | 4,16 | 1 | 4,166666667 |
| In the Municipality Online, change the way the name of the entity is displayed. | 1 | Approved | 5 | 1 | 5 |
| Visual and functional adjustments to the Group Launch Registration Screen. | 2 | Failed | 0 | 1 | Not sent for customer validation |
| Correct inconsistency after generating the second bank remittance for different paying accounts. | 2 | Approved | 5 | 1 | 5 |
| Include Legal Basis filter in the Annual Statement of Commitments. | 2 | Approved | 5 | 1 | 5 |
| Include period filters, tax amount range, and sale amount range in the Property Transfer Report. | 2 | Approved with remarks | 3,66 | 1 | 4,666666667 |
| Correct inconsistency in the Clearance Screen. The printer button is not working. | 2 | Approved | 5 | 1 | 5 |

| | | | | | |
|---|---------|-----------------------|-----|---|-----|
| Create modality filter in the Report of Minutes / Request for Commitment. | 2 | Approved | 5 | 1 | 5 |
| In the Municipality Online, change the way the name of the entity is displayed. | 2 | Approved | 5 | 1 | 5 |
| In the Human Resources module, create Summary Report by Type of Sheet. | Grupo 2 | Approved with remarks | 2,5 | 1 | 4,5 |

After applying the Shapiro-Wilk test to sample 3, $p = 0.000114744$, mean = 3.915625 and median = 5 were obtained. As $p < 0.05$, it is assumed that the data in sample 3 is not normally distributed, and it is not possible to apply parametric tests.

After the non-normality of sample 3 was verified, the same statistical test was applied to sample 4, which brings the average of correct answers for each task implemented with the support of the multimedia approach. The sample can be seen in the table 5:

Table 5. Average hits for each task implemented with the multimedia approach support.

| Task | Group | Validation (Testing Team) | Average hits | Use (2) or non-use of the approach (1) | Average customer grade |
|--|-------|---------------------------|--------------|--|----------------------------------|
| Create limiter in the overtime event. When selecting the Court of Auditors link (overtime), a field should appear to limit the maximum overtime. | 1 | Failed | 0 | 2 | Not sent for customer validation |
| Allow printing financial files by registering the server. | 1 | Approved | 5 | 2 | 5 |
| Indicate on the Expense Request screen whether or not it will reserve Budget balance | 1 | Approved | 5 | 2 | 5 |
| Include a lock for when the server is going to launch maternity leave, do not allow more than 120 days to not be | 1 | Approved with remarks | 4 | 2 | 5 |

| | | | | | |
|---|---|-----------------------|------|---|-------------|
| deducted in the Social Security Guide. | | | | | |
| Create Accountability Tab on the Agreement Screen. | 1 | Approved with remarks | 3,66 | 2 | 4,666666667 |
| Inserting the invoice number on the Summary of Engagement screen. | 1 | Approved | 5 | 2 | 5 |
| In the Patrimony Transfer Screen, link only to the sector. | 1 | Approved with remarks | 3,5 | 2 | 4,5 |
| Limit the term of COVID-19 legally based contracts to a maximum of 6 months. | 1 | Approved with remarks | 4,66 | 2 | 4,666666667 |
| Allow printing financial files by registering the server. | 2 | Approved with remarks | 3,5 | 2 | 4,5 |
| Create limiter in the overtime event. When selecting the Court of Auditors link (overtime), a field should appear to limit the maximum overtime. | 2 | Approved with remarks | 2 | 2 | 5 |
| Include a lock for when the server is going to launch maternity leave, do not allow more than 120 days to not be deducted in the Social Security Guide. | 2 | Approved with remarks | 5 | 2 | 5 |
| In the Patrimony Transfer Screen, link only to the sector. | 2 | Approved with remarks | 3,5 | 2 | 4,5 |
| Indicate on the Expense Request Screen whether or not it will reserve Budget balance. | 2 | Approved | 5 | 2 | 5 |
| Limit the term of COVID-19 legally based contracts to a maximum of 6 months. | 2 | Approved with remarks | 5 | 2 | 5 |

| | | | | | |
|---|---|-----------------------|------|---|-------------|
| Inserting the invoice number on the Summary of Engagement screen. | 2 | Approved | 4,83 | 2 | 4,833333333 |
| Create Accountability Tab on the Agreement Screen. | 2 | Approved with remarks | 3,83 | 2 | 4,833333333 |

After applying the Shapiro-Wilk test to sample 4, the values of $p = 0.000800559$, the mean = 3.967500 and the median = 4.33 were obtained. As $p < 0.05$, it is assumed that the data in sample 4 is not normally distributed. As normality was disapproved for samples 3 and 4, in order to validate the hypotheses listed for the second research question, it was decided to use the non-parametric test, Wilcoxon.

Analyzing the results, it was possible to verify that when the multimedia approach was used, the result was promising, that is, the average of correct answers per task was higher, when compared to the average of correct answers without using the approach. However, by applying the Wilcoxon statistical test, it was found that there was still no statistical significance to reject the hypothesis of equality between the two treatments. That is, even with the improvement in the average of correct answers, for programmers with the psychological profile and with the level of experience of the evaluated, the efficacies with and without the approach still need more replications of experiments like this one to find out their differences. The p-value is equal to 0.570603. This means that the chance of error, rejecting the null hypothesis (H_0), is: 0.5706 (57.06%). The higher the p value, the more it supports H_0 . Thus, H_0 is not rejected.

For the third research question listed (RQ3), which addresses the correlation between comprehension the code and the use of the tool, there was a low negative correlation coefficient, according to responses to the qualitative questionnaire, of -0.16. That is, surprisingly, when the tool was used, understanding, according to the programmers, fell by a proportion of only 16%. However, despite 32 samples, that is, a number greater than 30, which allows an approximation of the sample distribution by a normal distribution, the calculated T value, with 30 degrees of freedom ($n-2$), was -0.88, above the value of the critical T, which, for a significance level of 0.05, is -1.96. Thus, there is no statistical significance to reject the null hypothesis (H_0) that there is no correlation between comprehension the code and using the tool.

Among the factors that may have influenced this result, it is possible to mention: the developers' familiarity with the code; the quality and assertiveness of the multimedia content

attached to the tasks; lack of understanding of the code comprehension question; and the fact that most of the multimedia material attached to the tasks was persisted in an audio format. As suggested in [4], the audios with explanations of the code can be interesting, to allow the programmer to have help “on the fly” from his co-workers, previously recorded and available in the code, however, this same reference points that the developers can have order of preference of representation systems.

In this sense, as can be seen in Figure 12, most participants do not have the first or the second hearing preference. This may have impacted the comprehension of the code and may also have had a transitive effect on customer satisfaction, since well-executed tasks depend on good understanding. In other experiments and also as a professional lesson learned, the multimedia content must be created taking into account the proportions of the team's preferences, which can generate even better results in favor of the multimedia documentation. A programmer with kinesthetic preferences will prefer a video showing the operation of the system to each line of code executed and, if possible, a scene using a prototype. Finally, the tool may be more useful for beginning programmers.

For the fourth research question listed (RQ4), which addresses the correlation between the use of the tool and customer satisfaction, there was a positive correlation coefficient, according to responses to the qualitative questionnaire, of 0.07. That is, when the tool was used, customer satisfaction, according to himself, rose by 7%. The calculated T value, with 30 degrees of freedom ($n-2$), was 0.36, below the value of the critical T, which, for a significance level of 0.05, is 1.96. Thus, although the result is promising, it has not yet been possible to reject the null hypothesis that there is no correlation between the use of the tool and customer satisfaction. Anyway, depending on the current context of fierce market dispute, increasing customer satisfaction by 7% can be a competitive advantage.

At the end of the experiment, some participants expressed subjective opinions about the experience. Some comments referred to the CodeMedia extension interface, giving suggestions on how it could have its usability improved and how it would facilitate the process of playing multimedia content. The developers suggested: creating a bookmark or shortcut to list all points in the code that have multimedia documentation attached; unlink the constructor extension from the class; making the extension interface more similar to the interface of music players, especially with regard to the reproduction of the media attached to the code; and, allow to hear more than one audio in sequence.

Although improvements were suggested in the extension interface, none of the participants associated the difficulty in solving tasks with the usability of CodeMedia.

They also mentioned that the component used in the experiment is good for situating the use of a certain part of the system and even how the implementations should be made, but it only works in a context, where there are analysts responsible for the project, capable of providing the necessary information. for implementation of tasks. Criticisms were also made of the quality of the multimedia content attached to the tasks. Some participants complained about the low volume of some audios that were attached.

The facts presented help to justify and corroborate the results obtained in the experiment. It was noticeable, through the sample characterization questionnaire, that most of the participants were experienced and had already maintained a considerable number of systems, as can be seen in figures 16 and 17, respectively. It is likely that, if the experiment had been applied with less experienced programmers, without much familiarity with the code, the results would have been different and more favorable to the multimedia approach. In this context, it is necessary to apply the experiment with different programmer profiles, to find out how the multimedia approach behaves in different scenarios, and, thus, it is possible to make more general predictions about the use of multimedia content in the process of comprehension and maintenance of software.

4.8. Threats to Validity

For the present study, the following are evidenced:

4.8.1. Threats to Construction Validity

According to [26], the construction validity considers the relationships between theory and observation, that is, if the treatment reflects the cause well and the result reflects the effect well. The most common problems with this type of validity are: Design of the Experiment, the incorrect definition of the theoretical basis or the experimental evaluation; and, Human (or social) factors, participants can base their behavior on assumptions about the hypothesis; the human being usually tries to look better than he is when he is being observed, and researchers can design the experiment with the results they expect (bias).

To mitigate construction threats, with regard to methodological definition, the experimental evaluation proposed in this work was developed following the guidelines proposed by [10] [12] and [11]. The general objective was formalized using the GQM model (Goal, Question-Metric), proposed by [21] and [22]. In order to establish the construction validity, predictions were generated based on hypothesis construction, and these predictions were tested to support the validity of the instrument.

With regard to human (or social) factors, the research hypotheses and objectives were not presented to the participants. The experiment was applied in a controlled and comfortable context for the developers, in order to avoid that they feel pressured. To reduce the bias of the researchers' involvement with the research object, some activities that involved direct contact with the programmers, such as training the tool and applying the experiment, were carried out by an individual without involvement with the research.

4.8.2. Threats to Internal Validity

According to [26], internal validity defines whether the observed relationship between treatment and the result is causal, and is not the result of the influence of another uncontrolled or measured factor. According to [27], the most common threats to this type of validity are: Instrumentation: the difference in results is the result of an incorrect measurement or an inadequate instrument; Testing: the design allows participants to learn from their own mistakes; it is related to the sequence of results collection versus the moment of the intervention, for example; Maturation: subjects can become more capable or become demotivated over time; Selection: the participants were not selected at random or the groups were not divided equally in both quantitative and qualitative aspects.

To mitigate the threats found, the following practices were adopted: the participants were randomly divided into two groups, in order to promote the miscegenation of the sample and thus not favor any of the two treatments (coding with and without using the approach). The tasks used during the experiment were selected so that they were comparable in terms of complexity and coding time. The definition of the tasks that would be made using the multimedia approach was also carried out at random. Each programmer executed 4 different tasks, in order to prevent them from learning from their mistakes. The instrumentation used during the experiment was tested and verified, in order to avoid unforeseen events that could

impact the results of the experiment. After application, the criteria were uniformly used for both treatments.

4.8.3. Threats to External Validity

The external validity, according to [26], defines conditions that limit the ability to generalize the results of an experiment to industrial practice. The most common threats to this type of validity are: Participants: selecting volunteers who are unrelated or do not reflect the behavior of the population; Experiment setup: perform the experiment in an environment very different from the ideal, or adopt instruments that are far from reality.

To reduce these threats, the following measures were adopted: the experiment was applied within a private software development company, with real questions and problems. Participants were programmers who work daily with maintenance tasks, from correcting errors to adapting pre-existing features. The cost of implementing this approach was also another point analyzed during the experiment, in order to verify whether in a real context, the multimedia approach would be sustainable for long periods.

Many of the strategies used to increase internal validity make the study more restricted, which ends up decreasing its external validity and, consequently, reducing the chances of application in a real environment. In view of the aforementioned fact, when planning the methodology adopted in this study, it was decided to promote a balance between the two validities. Finally, it is worth mentioning that the results are applicable for content production with auditory majority and for programmers with experience similar to that of the programmers who participated in the evaluation, who had, in their majority, an order of preference of the representation systems of the type: Kinesthetic - Visual - Auditory. In other words, the application of the same experiment with inexperienced programmers of first hearing preference can produce much better results.

4.9. Conclusion and Future Works

Understanding the requirements of a product is not a simple task, as delivering to the customer what he wants and needs is an activity that requires experience, attention and effort, to ensure that these requirements are always complete, consistent, relevant and up to date. To support this process, as it is an intensely human area, it is necessary to use increasingly

assertive and repeatable techniques, which explore the different cognitive channels of the people involved, giving them the possibility to reproduce, interpret, understand and learn information about the tasks to be automated, in a more natural and less costly way.

In this context, the present work presents important contributions to increase the effectiveness in the process of comprehension and maintaining software, through the use of multimedia resources to persist the user requirements. The multimedia approach proposed here was automated by the CodeMedia plug-in, which enables the integration of multimedia requirements into the source code, making an innovation for those who adopt this approach. Tight integration with the code can allow multimedia tools to be used effectively to analyze and understand the data produced during the evolution of the software. In addition to recording or filming customer interviews and providing them to the programmer, so that the programmer better understands the requirements, a programmer must have the facility to click on a link in the source code and see or hear the stakeholder interviews and explanations code recorded by co-workers who created the code. In this case, explanations can also include increments and evolutions.

In the analysis of the results, with regard to development time and code compression, the approach obtained indicators below expectations. These indexes may be correlated to the quality of the multimedia content attached to the code, the programming style of the developers, the lack of understanding of the question about comprehension the code, the familiarity of the developers with the source code and / or the time for reproduction and understanding multimedia content. Anyway, it was not possible to refute the hypothesis that this approach interferes, positively or negatively, in the coding time and in the comprehension of the code. Probably, especially in terms of efficiency, the numbers will be better in a context with less experienced developers.

In addition, despite the need for more replications of the experiment for greater statistical significance, the results related to effectiveness were promising. The average number of correct answers per task was greater, compared to the average number of correct answers without using the approach. From a business perspective, when the tool was used, customer satisfaction rose by 7%. From a commercial point of view and considering a real software maintenance context, increasing customer satisfaction by 7% can be a competitive advantage.

Finally, as future work, new implementations and experiments are being carried out in order to find evidence that the use of the multimedia approach can decrease the time of

implementation and improve comprehension in different contexts, as well as corroborating evidence of the reduction of errors in coding and increased customer satisfaction.

Declarations

Availability of data and materials

The questionnaires mentioned in this paper are available at https://drive.google.com/drive/folders/1qcR8Fr2_7etoCZEqqzSQsMJVlrTV4Yg5?usp=sharing.

Competing interests

The authors declare that they have no competing interests.

Funding

The authors declare that there is no funding.

Authors' contributions

All these results presented here were evidenced and analyzed by the authors. In this paper, our main contribution was an experiment that evaluates the feasibility of the Multimedia Approach, automated by the CodeMedia plugin, to promote gains in efficiency and effectiveness in the process of comprehension and maintaining software. All authors read and approved the final manuscript.

Acknowledgements

Not applicable.

References

1. Calazans, A., Mariano, A.M.: A utilização da linguagem natural na especificação de requisitos : um estudo por meio das equações estruturais. WER 2016 - 19o Workshop em Engenharia de RequisitosAt: Quito, Ecuador. April, 2016.
2. Menten, A., Scheibmayr, S., Klimpke, L.: Using audio and collaboration technologies for distributed requirements elicitation and documentation. In: 2010 3rd International

- Workshop on Managing Requirements Knowledge, MaRK'10. pp. 51–59 (2010).
3. Bruni, E., Ferrari, A., Seyff, N., Tolomei, G.: Automatic analysis of multimodal requirements: A research preview. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*. 7195 LNCS, 218–224 (2012).
 4. Colaço Júnior, M., Menezes, M. de F., Corumba, D., Mendonça, M.: Do Software Engineers Have Preferred Representational Systems? *J. Res. Pract. Inf. Technol.* (2017).
 5. Chen, H., Yin, C., Li, R., Rong, W., Xiong, Z., David, B.: Enhanced Learning Resource Recommendation Based on Online Learning Style Model. in *Tsinghua Science and Technology*, vol. 25, no. 3, pp. 348–356, June 2020, doi: 10.26599/TST.2019.9010014
 6. Gartner, S., Schneider, K.: A method for prioritizing end-user feedback for requirements engineering. In: *2012 5th International Workshop on Co-operative and Human Aspects of Software Engineering, CHASE 2012 - Proceedings*. pp. 47–49 (2012).
 7. Creighton, O., Ott, M., Bruegge, B.: Software cinema - Video-based requirements engineering. In: *Proceedings of the IEEE International Conference on Requirements Engineering*. pp. 106–115 (2006).
 8. Karras, O., Kiesling, S., Schneider, K.: Supporting Requirements Elicitation by Tool-Supported Video Analysis. In: *Proceedings - 2016 IEEE 24th International Requirements Engineering Conference, RE 2016*. pp. 146–155. Institute of Electrical and Electronics Engineers Inc. (2016).
 9. Haumer, P., Jarke, M., Pohl, K., Weidenhaupt, K.: Improving reviews of conceptual models by extended traceability to captured system usage. *Interact. Comput.* 13, 77–95 (2000).
 10. Basili, V.R.: The Role of Experimentation in Software Engineering: Past, Current, and Future. In: *Proceedings of IEEE 18th International Conference on Software Engineering*. pp. 442–449. IEEE, Berlin/Heidelberg (1996).
 11. SANTOS, B. S. S; COLAÇO JÚNIOR, M; SOUZA, J.G.: A Initial Experimental Evaluation of the NeuroMessenger: A Collaborative Tool to Improve the Empathy of Text Interactions. In: *Proceedings of the 15th International Conference on Information Technology: New Generations*. pp. 411–419. Springer International Publishing (2018).
 12. Nogueira de Oliveira, Robert & Colaço Júnior, Methanias. (2018). Experimental Analysis of Stemming on Jurisprudential Documents Retrieval. *Information*. 9. 28 (2018).
 13. Pham, R., Meyer, S., Kitzmann, I., Schneider, K.: Interactive multimedia storyboard for

- facilitating stakeholder interaction: Supporting continuous improvement in IT-ecosystems. In: Proceedings - 2012 8th International Conference on the Quality of Information and Communications Technology, QUATIC 2012. pp. 120–124 (2012).
14. Karras, O., Schneider, K.: Software Professionals are Not Directors : What Constitutes a Good Video ? In: 1st International Workshop on Learning from other Disciplines for Requirements Engineering. IEEE, Hannover, Germany (2018).
 15. Bruegge, B., Stangl, H., Reiss, M.: An experiment in teaching innovation in software engineering : Video presentation. In: Proceedings of the Conference on Object-Oriented Programming Systems, Languages, and Applications, OOPSLA. pp. 807–809 (2008).
 16. Hollis, C., Bhowmik, T.: Automated support to capture verbal just-in-time requirements in agile development: A practitioner view. In: Proceedings - 2017 IEEE 25th International Requirements Engineering Conference Workshops, REW 2017. pp. 419–422. Institute of Electrical and Electronics Engineers Inc. (2017).
 17. BASS, Len; CLEMENTS, Paul; KAZMAN, R.: Software architecture in practice. Addison-Wesley Longman Publishing Co., Boston, MA, United states (2003).
 18. Shaw, M., Garlan, D.: Software Architecture. In: Prentice Hall (ed.) Perspectives on an Emerging Discipline (1996).
 19. Shaw, M., Garlan, D.: An introduction to software architecture. Tech. Rep. (1994).
 20. Vis, N.E.T.T.: Introdução à linguagem C# e ao .NET. Microsoft, Was (2020).
 21. Weiss, D., Basili, V.R.: A Methodology for Collecting Valid Software Engineering Data. IEEE Trans. Softw. Eng. 10, 728–738 (1984).
 22. Solingen, R. Van, Berghout, E.: The Goal / Question / Metric Method : A Practical Guide for Quality Improvement of Software Development. (1999).
 23. Costa, J., Santos, I., Nascimento, A., Colaço Júnior, M.: Experimentação na Indústria para Aumento da Efetividade da Construção de Procedimentos ETL em um Ambiente de Business Intelligence Alternative Title : Experimentation at Industrial Setting to Improve the Effectiveness of the ETL Procedures Implementation. In: Simpósio Brasileiro de Sistemas de Informação. pp. 459–466. , Porto Alegre (2015).
 24. Anderson, D.J.: Kanban: Successful Evolutionary Change for Your Technology Business. Blue Hole Press (2010).
 25. Fred, D., Davis, B.F.D.: Perceived Usefulness , Perceived Ease of Use , and User Acceptance of Information Technology. MIS Q. 13, 319–340 (1989).
 26. Travassos, G H; Gurov, D; Amaral, E.A.G.: Introdução à Engenharia de Software

Experimental. UFRJ, Rio de Janeiro (2002).

27. Wainer, J.: Métodos de pesquisa quantitativa e qualitativa para a ciência da computação. In: Jornada de Atualização em Informática. pp. 221–262 (2007).

Realizada a Avaliação Experimental, serão apresentadas, no próximo capítulo, as contribuições, conclusões e dificuldades obtidas, bem como os possíveis trabalhos futuros.

5. CONCLUSÃO

A compreensão de software é uma das atividades mais complexas do ciclo de desenvolvimento de sistemas, pois ela envolve diferentes pessoas, com diferentes pontos de vista e diferentes formas de expressão. Definir o escopo de um software com base nessas impressões se torna uma tarefa extremamente difícil e sujeita a falhas, dada a subjetividade envolvida.

Para mitigar esse risco e evitar a construção de softwares ineficientes, que não atendem às necessidades de seus clientes, a Engenharia de Software vem se utilizando de recursos textuais para registrar e enriquecer a documentação de software, a fim de dar um maior esclarecimento e visibilidade às características do sistema. Todavia, as documentações textuais têm se mostrado insuficientes, principalmente, dentro de um contexto no qual as descrições escritas, para algumas pessoas, podem não ser suficientes para o entendimento do que precisa ser desenvolvido, prejudicando a compreensão e a execução de demandas. Tal como explicitado anteriormente (vide capítulo 1), indivíduos, em contextos específicos, podem ter canais preferenciais para absorver e interpretar a informação.

Dado este cenário, este trabalho se propôs a responder as seguintes questões de pesquisa (vide Seção 1.1):

- **RQ1** - A utilização da abordagem multimídia, automatizada pelo uso do *plug-in CodeMedia*, pode reduzir o tempo de codificação dos programadores no processo de manutenção de software?
- **RQ2** - A utilização da abordagem multimídia, automatizada pelo uso do *plug-in CodeMedia*, pode reduzir erros na codificação para manutenção de software?
- **RQ3** - A utilização da abordagem multimídia, automatizada pelo uso do *plug-in CodeMedia*, pode aumentar o nível de compreensão de software pelos programadores no processo de manutenção de software?
- **RQ4** - A utilização da abordagem multimídia, automatizada pelo uso do *plug-in CodeMedia*, pode aumentar o nível de satisfação do cliente com a solução desenvolvida, do ponto de vista de usabilidade da tarefa entregue?

Para respondê-las, foi construída uma abordagem multimídia, automatizada pelo *plugin CodeMedia*, para apoiar o processo de compreensão e manutenção de software, bem

como foi realizada uma avaliação experimental para verificar a viabilidade da abordagem, do ponto de vista de eficiência e eficácia.

Para primeira questão de pesquisa, a partir do experimento feito (vide capítulo 4), constatou-se que quando a abordagem multimídia foi utilizada, o tempo de codificação foi um pouco maior, se comparado ao tempo sem o uso da abordagem. Alguns fatores podem explicar esse resultado, dentre eles: o estilo de programação apresentado por cada participante e o tempo necessário para reprodução e entendimento do conteúdo multimídia anexado ao código.

Para segunda questão de pesquisa, foi possível constatar que quando a abordagem multimídia foi utilizada, a média de acertos por tarefa foi maior, se comparada à média de acertos sem o uso da abordagem. Para a terceira questão de pesquisa, houve um coeficiente de correlação negativo baixo, segundo respostas do questionário qualitativo, de -0,16. Ou seja, quando a ferramenta foi usada, a compreensão, segundo os programadores, caiu numa proporção de apenas 16%. Dentre os fatores que podem ter influenciado esse resultado, é possível citar: a familiaridade dos desenvolvedores com o código; a qualidade e assertividade do conteúdo multimídia anexado às demandas; a falta de entendimento da pergunta de compreensão do código; e o fato da maior parte do material multimídia anexado às demandas ter sido persistido em formato de áudio. Já para quarta questão de pesquisa, evidenciou-se que quando a ferramenta foi usada, a satisfação do cliente, segundo ele próprio, subiu numa proporção de 7%.

Finalmente, é importante salientar que os principais desafios enfrentados neste trabalho foram: aquisição de voluntários para realização da avaliação experimental; aplicação do experimento, considerando que ele foi aplicado em um contexto real de desenvolvimento de software; e a adequação do experimento às restrições impostas pela pandemia ocasionada pela Covid-19. Apesar das evidências encontradas e das questões de pesquisas terem sido respondidas, estas não foram as únicas contribuições deste trabalho, sendo as demais apresentadas na próxima seção.

5.1. Resultados e Contribuições

Dando prosseguimento ao tópico anterior, a principal contribuição deste estudo consiste na condução de um processo experimental para avaliar o uso da abordagem multimídia para ganho de eficiência e eficácia no processo de compreensão e manutenção de software. O

processo experimental adotado neste estudo seguiu as diretrizes de (BASILI, 1996) (SANTOS, COLAÇO JÚNIOR, & SOUZA, 2018) e (OLIVEIRA & COLAÇO JÚNIOR, 2018), considerando as adaptações necessárias, impostas pela pandemia causada pela Covid 19 – coronavírus. Dada a sua natureza sistêmica, tais diretrizes facilitam a condução, replicação e empacotamento de experimentos, mitigando possíveis ameaças à validade da pesquisa, além de fornecer evidências para tomada de decisões mais claras e objetivas. Além disso, destacam-se as seguintes contribuições deste trabalho:

- Mapeamento sistemático das abordagens que promovem o uso de multimídia para apoiar o processo de compreensão e manutenção de software. Seus resultados foram submetidos e publicados pelo *Journal of Software: Evolution and Process*;
- Experimento que avalia a viabilidade da Abordagem Multimídia, automatizada pelo *plugin CodeMedia*, para promover ganho de eficiência e eficácia no processo de compreensão e manutenção de software. Ressalta-se que a Avaliação Experimental, assim como seus resultados, foi submetida ao *Journal of the Brazilian Computer Society (JBACS)*;
- *Plugin CodeMedia*. Foi desenvolvido um *plugin*, integrado ao *VisualStudio*, para automatizar a Abordagem Multimídia proposta neste trabalho. Esta ferramenta poderá eventualmente ser utilizada em contextos reais, para apoiar o desenvolvimento e manutenção de software;
- Geração de evidências experimentais sobre o uso de mídias dinâmicas (áudio e vídeo) na área de compreensão e manutenção de software. Os resultados encontrados no experimento foram promissores (vide capítulo 4). Foi constatado que quando a Abordagem Multimídia foi utilizada, houve aumento da satisfação do cliente com o produto final, bem como houve aumento do número de acertos na codificação de tarefas.

Apesar dos resultados promissores, ainda são necessárias mais averiguações acerca do fenômeno estudado e do uso da Abordagem Multimídia como facilitadora no processo de compreensão e manutenção de software. Na próxima seção, serão apresentados possíveis desdobramentos relacionados a esta pesquisa.

5.2. Trabalhos Futuros

Devido à necessidade de maiores evidências empíricas do uso da abordagem multimídia para ganho de eficiência e eficácia no processo de compreensão e manutenção de software, destacam-se os possíveis trabalhos futuros:

- Realização de novas avaliações experimentais da Abordagem Multimídia;
- Replicação dos experimentos realizados (vide Capítulo 4), em contextos diferentes, com programadores com menos experiência na linguagem utilizada no experimento;
- Melhoria do *plugin CodeMedia*, considerando aspectos de usabilidade, estrutura e armazenamento de arquivos.

REFERÊNCIAS

BASILI, V. R. **The Role of Experimentation in Software Engineering: Past, Current, and Future.** Proceedings of IEEE 18th International Conference on Software Engineering. *Anais...*Berlin/Heidelberg: IEEE, 1996.

BOULILA, N.; HOFFMANN, A.; HERRMANN, A. **Using Storytelling to record requirements: Elements for an effective requirements elicitation approach.** 2011 4th Int. Workshop on Multimedia and Enjoyable Requirements Eng. - Beyond Mere Descriptions and with More Fun and Games, MERE'11 - Co-located with the 19th IEEE Int. Requirements Eng. Conf., RE'11. *Anais...*2011. Disponível em: <<https://www.scopus.com/inward/record.uri?eid=2-s2.0-80555123774&doi=10.1109%2FMERE.2011.6043945&partnerID=40&md5=8e798e4fe0c73d881342b472c60feed>>.

BRILL, O.; SCHNEIDER, K.; KNAUSS, E. **Videos vs. use cases: Can videos capture more requirements under time pressure?** Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), v. 6182 LNCS, p. 30–44, 2010.

BRUNI, E. *et al.* **Automatic analysis of multimodal requirements: A research preview.** Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), v. 7195 LNCS, p. 218–224, 2012.

CALAZANS, A.; MARIANO, A. M. **A utilização da linguagem natural na especificação de requisitos : um estudo por meio das equações estruturais.** WER 2016 - 19o Workshop em Engenharia de RequisitosAt: Quito, Ecuador. April, 2016.

CHANGBAI, C. **Intellectual Property Strategy - Optimal allocation of Intellectual Property resources in economic growth** [M]. Beijing: Science Press, 1998 (68).

CHEN, H. *et al.* **Enhanced Learning Resource Recommendation Based on Online Learning Style Model.** TSINGHUA SCIENCE AND TECHNOLOGY. p. 348–356, 2020.

COLAÇO JÚNIOR, M. *et al.* **Do Software Engineers Have Preferred Representational Systems?** Journal of Research and Practice in Information Technology, n. June, 2017.

COLAÇO JÚNIOR, M., MENDONÇA, M.G., FARIAS, M.A. and HENRIQUE, P. (2010): **OSS Developers Context- Specific Preferred Representational Systems: An Initial Neurolinguistic Text Analysis of the Apache Mailing List.** In: 7th IEEE Working Conference on Mining Software Repositories, Cape Town, SA, 126–129.

CREIGHTON, O.; OTT, M.; BRUEGGE, B. **Software cinema - Video-based requirements engineering.** Proceedings of the IEEE International Conference on Requirements Engineering. *Anais...*2006. Disponível em: <<https://www.scopus.com/inward/record.uri?eid=2-s2.0-34748911363&doi=10.1109%2FIRE.2006.59&partnerID=40&md5=b28f3f785fab25d0e55d2bc56c109b85>>.

FLUCKIGER, F. **Development and application of multimedia Network** [M]. Beijing: Machinery Industry Press, 1997.

GARTNER, S.; SCHNEIDER, K. **A method for prioritizing end-user feedback for requirements engineering.** 2012 5th International Workshop on Co-operative and Human Aspects of Software Engineering, CHASE 2012 - Proceedings. *Anais...*2012. Disponível em: <<https://www.scopus.com/inward/record.uri?eid=2-s2.0-84864143377&doi=10.1109%2FCHASE.2012.6223020&partnerID=40&md5=e8e5ab581e4b3a73a3aaf1323825640c>>.

HAUMER, P. *et al.* **Improving reviews of conceptual models by extended traceability to captured system usage.** Interacting with Computers, v. 13, n. 1, p. 77–95, 2000.

KARRAS, O. *et al.* **Video as a by-product of digital prototyping: Capturing the dynamic aspect of interaction.** Proceedings - 2017 IEEE 25th International Requirements Engineering Conference Workshops, REW 2017. *Anais...*Institute of Electrical and Electronics Engineers Inc., 2017. Disponível em: <<https://www.scopus.com/inward/record.uri?eid=2-s2.0-85034644587&doi=10.1109%2FREW.2017.16&partnerID=40&md5=a99356b0de35de0fc5b>>

bbd665fffc24e>.

KARRAS, O.; KIESLING, S.; SCHNEIDER, K. **Supporting Requirements Elicitation by Tool-Supported Video Analysis**. Proceedings - 2016 IEEE 24th International Requirements Engineering Conference, RE 2016. **Anais...**Institute of Electrical and Electronics Engineers Inc., 2016. Disponível em: <<https://www.scopus.com/inward/record.uri?eid=2-s2.0-85007155242&doi=10.1109%2FRE.2016.10&partnerID=40&md5=697578e1b5ee505639aa8fbfaed6671f>>.

MENTEN, A.; SCHEIBMAYR, S.; KLIMPKE, L. **Using audio and collaboration technologies for distributed requirements elicitation and documentation**. 2010 3rd International Workshop on Managing Requirements Knowledge, MaRK'10. **Anais...**2010. Disponível em: <<https://www.scopus.com/inward/record.uri?eid=2-s2.0-78650485207&doi=10.1109%2FMARK.2010.5623808&partnerID=40&md5=c359718fa9df5fdc2cb85abba4e328ad>>.

OLIVEIRA, R. A. DE.; COLAÇO JÚNIOR, M. **Experimental Analysis of Stemming on Jurisprudential Documents Retrieval**. Information, v. 9, p. 28, 2018.

SANTOS, B. S. S; COLAÇO JÚNIOR, M; SOUZA, J. G. **A Initial Experimental Evaluation of the NeuroMessenger: A Collaborative Tool to Improve the Empathy of Text Interactions**. Proceedings of the 15th International Conference on Information Technology: New Generations. **Anais...**Springer International Publishing, 2018.

SANTOS, A. C. M.; COLAÇO JÚNIOR, M.; ANDRADE, E. C. **Multimedia Resources as a support for requirements engineering: A systematic mapping**. Journal of Software: Evolution and Process. DOI:10.1002/smr.2327. 2020.

APÊNDICES

APÊNDICE A - Formulário de Caracterização da Amostra



Universidade Federal de Sergipe

Cidade Universitária Prof. José Aloísio Campos

Programa de Pós-Graduação em Ciência da Computação (PROCC)

Formulário de Caracterização da Amostra

Formulário de caracterização de participantes aplicado na realização do experimento.

*** Obrigatório**

Dados Pessoais

1. Nome completo *

2. E-mail *

3. Idade (Marcar apenas uma opção) *

(a) Até 18 anos

(b) Entre 19 e 30 anos

(c) Entre 31 e 40 anos

(d) Entre 41 e 50 anos

(e) Acima de 50 anos

4. Gênero (Marcar apenas uma opção) *

- (a) Feminino
- (b) Masculino
- (c) Outro: _____

5. Formação (Marcar apenas uma opção) *

- (a) Técnico
- (b) Graduado
- (c) Especializado
- (d) Mestrado
- (e) Doutorado

6. Identifique qual é o seu modo de aprendizagem preferencial, para absorver e interpretar informação. Indique a sua ordem de preferência. *

1 = primeiro em ordem de preferência

2 = segundo em ordem de preferência

3 = último em ordem de preferência

[] Auditivo (vale mais escutar)

[] Visual (imagem é tudo)

[] Cinestésico (mão na massa)

7. Você faz uso de alguma técnica para documentação dos requisitos de usuário? (Marcar apenas uma opção) *

- (a) Sim
- (b) Não

8. Caso sim, especifique quais são as técnicas que você utiliza para descrever as características e requisitos do usuário.

Dados Profissionais

9. Profissão/Cargo *

10. Área de Atuação (Marcar apenas uma opção) *

- (a) Desenvolvimento
- (b) Suporte e Desenvolvimento
- (c) Desenvolvimento e Infraestrutura
- (d) Desenvolvimento, Infraestrutura e Suporte

11. Anos de experiência na linguagem utilizada no experimento (Marcar apenas uma opção) *

- (a) Até 3 anos
- (b) 3 a 5 anos
- (c) 5 a 10 anos
- (d) Mais de 10 anos

12. Experiência (número de sistemas já mantidos) *

- (a) 1 a 5
- (b) 6 a 10
- (c) 11 a 20
- (d) Mais de 20

13. Linguagem conhecida (é possível selecionar mais de uma opção). *

- (a) C#

- (b) Java
- (c) PHP
- (d) Python
- (e) Ruby
- (f) Typescript
- (g) Swift
- (h) C++
- (i) JavaScript
- (j) Outro: _____

APÊNDICE B – Pergunta de Avaliação do Nível de Compreensão do Código



Universidade Federal de Sergipe
Cidade Universitária Prof. José Aloísio Campos
Programa de Pós-Graduação em Ciência da Computação (PROCC)

Pergunta para Avaliação do Nível de Compreensão do Código

Pergunta de dimensão qualitativa/quantitativa, para avaliar o nível de compreensão do código pelos programadores, para cada tarefa executada.

*** Obrigatório**

1. De 1 a 5, com relação à compressão do código desta tarefa, que nota você daria?*

1 2 3 4 5

Não compreendi ☐ ☐ ☐ ☐ ☐ Compreendi Totalmente

APÊNDICE C - Questionário de Avaliação de Nível de Satisfação do Cliente



Universidade Federal de Sergipe

Cidade Universitária Prof. José Aloísio Campos

Programa de Pós-Graduação em Ciência da Computação (PROCC)

Questionário de Avaliação de Nível de Satisfação do Cliente

Questionário de dimensão qualitativa/quantitativa, para avaliar o nível de satisfação do cliente com as tarefas executadas pelos programadores, com e sem o auxílio da extensão *CodeMedia*.

* Obrigatório

1 - Esta tarefa contém todas as funções e recursos que eu espero que ela tenha. *

1 2 3 4 5

Discordo Totalmente ☐ ☐ ☐ ☐ ☐ Concordo Totalmente

2 - Posso efetivamente concluir o meu trabalho com a conclusão desta tarefa. *

| | 1 | 2 | 3 | 4 | 5 | |
|---------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|---------------------|
| Discordo Totalmente | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | Concordo Totalmente |

3 - Eu sou capaz de completar ou concluir o meu trabalho, da forma mais eficiente (rápida) possível, após a disponibilização deste ajuste. *

| | 1 | 2 | 3 | 4 | 5 | |
|---------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|---------------------|
| Discordo Totalmente | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | Concordo Totalmente |

4 - Com a execução desta tarefa, o sistema ficou mais fácil de usar. *

| | 1 | 2 | 3 | 4 | 5 | |
|---------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|---------------------|
| Discordo Totalmente | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | Concordo Totalmente |

5 - Com a execução desta tarefa, a interface do sistema ficou mais agradável. *

| | 1 | 2 | 3 | 4 | 5 | |
|---------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|---------------------|
| Discordo Totalmente | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | Concordo Totalmente |

6 - No geral, estou satisfeito com esta tarefa. *

| | 1 | 2 | 3 | 4 | 5 | |
|---------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|---------------------|
| Discordo Totalmente | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | Concordo Totalmente |

APÊNDICE D – Questionário de Avaliação de Nível de Compreensão de Software



Universidade Federal de Sergipe

Cidade Universitária Prof. José Aloísio Campos

Programa de Pós-Graduação em Ciência da Computação (PROCC)

Questionário de Avaliação de Nível de Compreensão de Software

Questionário de dimensão qualitativa, para avaliar o nível de compreensão de software pelos programadores, com e sem a utilização da abordagem multimídia, considerando todas as demandas executadas.

*** Obrigatório**

1. O uso de documentação multimídia REDUZIU o tempo e esforço necessários para compreensão do código. *

1 2 3 4 5

Discordo Totalmente ☐ ☐ ☐ ☐ ☐ Concordo Totalmente

2. Os requisitos multimídia estavam confusos, vagos e ambíguos. *

1 2 3 4 5

Discordo Totalmente ☐ ☐ ☐ ☐ ☐ Concordo Totalmente

3. A documentação multimídia NÃO forneceu meios mais eficazes para codificar as funcionalidades do sistema. *

1 2 3 4 5

Discordo Totalmente ☐ ☐ ☐ ☐ ☐ Concordo Totalmente

4. A compreensão do código, SEM o auxílio da documentação multimídia, foi dificultada. *

1 2 3 4 5

Discordo Totalmente ☐ ☐ ☐ ☐ ☐ Concordo Totalmente

5. O uso da extensão *CodeMedia* DIFICULTOU o entendimento do código. *

1 2 3 4 5

Discordo Totalmente ☐ ☐ ☐ ☐ ☐ Concordo Totalmente

6. Compreender o software, considerando apenas códigos, comentários e documentação tradicional, é mais rápido e prático do que com a extensão *CodeMedia*. *

1 2 3 4 5

Discordo Totalmente ☐ ☐ ☐ ☐ ☐ Concordo Totalmente

7. A documentação multimídia se mostrou mais compreensível, portanto, pode substituir as técnicas de documentação textual. *

1 2 3 4 5

Discordo Totalmente ☐ ☐ ☐ ☐ ☐ Concordo Totalmente

8. Documentação multimídia deve ser utilizada no contexto de manutenção de software, porém, a extensão *CodeMedia* não é a melhor opção para promover essa abordagem.

*

1 2 3 4 5

Discordo Totalmente ☐ ☐ ☐ ☐ ☐ Concordo Totalmente

9. Com o uso da extensão *CodeMedia* não será mais necessário entrar em contato com o desenvolvedor para tirar dúvidas sobre o código original.

*

1 2 3 4 5

Discordo Totalmente ☐ ☐ ☐ ☐ ☐ Concordo Totalmente

10. A extensão *CodeMedia*, com seu acoplamento ao código, é simples e fácil de usar, o que facilita a visualização da documentação durante o processo de manutenção de software.

*

1 2 3 4 5

Discordo Totalmente ☐ ☐ ☐ ☐ ☐ Concordo Totalmente

11. Abordagem proposta pela extensão *CodeMedia* é escalável e pode ser facilmente aplicada no dia-a-dia das empresas de desenvolvimento de software.

*

1 2 3 4 5

Discordo Totalmente ☐ ☐ ☐ ☐ ☐ Concordo Totalmente

12. A documentação multimídia pode ser utilizada de forma complementar às documentações textuais (comentários, documento de requisitos, casos de uso).

*

1 2 3 4 5

Discordo Totalmente ☐ ☐ ☐ ☐ ☐ Concordo Totalmente