SUPPORTING THE GENERATION OF AUTOMATED ACCEPTANCE TESTS OF PROCESS-AWARE INFORMATION SYSTEMS

Tales Mello Paiva

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientadores: Toacy Cavalcante de Oliveira
Raquel Mainardi Pillat Basso

Rio de Janeiro
Agosto de 2024

SUPPORTING THE GENERATION OF AUTOMATED ACCEPTANCE TESTS OF PROCESS-AWARE INFORMATION SYSTEMS

Tales Mello Paiva

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Orientadores: Toacy Cavalcante de Oliveira
Raquel Mainardi Pillat Basso

Aprovada por: Prof. Toacy Cavalcante de Oliveira
Prof. Raquel Mainardi Pillat Basso
Prof. Cláudia Maria Lima Werner
Prof. Paulo Sérgio Conceição de Alencar

RIO DE JANEIRO, RJ – BRASIL
AGOSTO DE 2024

Mello Paiva, Tales

Supporting the Generation of Automated Acceptance Tests of Process-Aware Information Systems/Tales Mello Paiva. – Rio de Janeiro: UFRJ/COPPE, 2024.

XVII, 103 p.: il.; 29, 7cm.

Orientadores: Toacy Cavalcante de Oliveira
Raquel Mainardi Pillat Basso

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2024.

Referências Bibliográficas: p. 80 – 84.

1. Test Automation. 2. BPMN. 3. PAIS. I. Cavalcante de Oliveira, Toacy *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

À família, minha maior riqueza.

Agradecimentos

Primeiramente, agradeço à minha esposa, Franciane, que esteve ao meu lado e me apoiou incondicionalmente durante toda essa jornada, seja em Macaé, Rio de Janeiro ou Waterloo.

À minha família, minha maior fonte de inspiração: Tetê, Edu, Pedro, Cleide, Paulo e Jonas, meu mais sincero agradecimento pela torcida constante.

Aos meus orientadores, Toacy e Raquel, e ao meu tio Walter, sou grato por todo o incentivo e ensinamentos que me proporcionaram.

Agradeço ao Ulisses, à Gláucia, ao Prof. Paulo Alencar, ao Prof. Don Cowan, ao Prof. Dan Berry, à University of Waterloo e ao ÉduCanada, por expandirem meus horizontes e contribuírem para meu crescimento acadêmico e pessoal.

Aos meus mais-que-colegas de trabalho, pela paciência, compreensão e *insights*: Carlos, Tati, Índio, Andressa, Mari, e tantos outros, meu muito obrigado.

Às comunidades de usuários do AgileKIP, Camunda e Robot Framework, em especial ao Asko, ao Markus e ao Thomas, agradeço por me encorajarem a explorar novas possibilidades.

Por fim, mas não menos importante, agradeço à UFRJ, à Coppe e a todos do PESC, em especial aos companheiros de jornada Mariano, Clinton, Marcus e Carlos, pela amizade e companheirismo.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

SUPORTE À GERAÇÃO DE TESTES DE ACEITAÇÃO AUTOMATIZADOS DE SISTEMAS DE INFORMAÇÃO CIENTES DE PROCESSO

Tales Mello Paiva

Agosto/2024

Orientadores: Toacy Cavalcante de Oliveira
Raquel Mainardi Pillat Basso

Programa: Engenharia de Sistemas e Computação

Esta dissertação apresenta o AATPAIS (Automated Acceptance Testing of Process-Aware Information Systems), uma solução projetada para abordar as complexidades de testar PAIS. O AATPAIS automatiza a geração e execução de casos de teste de aceitação derivados de modelos de processos BPMN e especificações de sistema, utilizando Robot Framework por suas robustas capacidades de RPA para automação de tela. O sistema extrai pontos de interação humana dentro de modelos de processos, como Eventos de Início e Tarefas do Usuário, e gera scripts RPA correspondentes para realizar testes de aceitação.

A avaliação do AATPAIS envolveu avaliar sua eficácia em uma variedade de modelos de processos. Os resultados demonstraram uma cobertura de caminho de 100% em 15 dos 21 modelos de processos sem customização adicional, demonstrando a eficiência da ferramenta no tratamento de ambientes de processos complexos e dinâmicos. No entanto, certos modelos de processos revelaram limitações devido a condições de *gateway* e acionadores de eventos, indicando áreas para melhoria futura.

O feedback da pesquisa *survey* destacou o potencial significativo da AATPAIS para melhorar a produtividade e simplificar os processos de teste. Embora os respondentes tenham apreciado a sua utilidade, algumas preocupações sobre a facilidade de uso e a curva de aprendizado foram observadas, sugerindo a necessidade de uma melhor instrução e de refinamento da interface. No geral, o AATPAIS apresenta uma abordagem promissora para automatizar testes de aceitação em PAIS, com potencial para desenvolvimento adicional visando melhorar suas capacidades e experiência do usuário.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

SUPPORTING THE GENERATION OF AUTOMATED ACCEPTANCE TESTS OF PROCESS-AWARE INFORMATION SYSTEMS

Tales Mello Paiva

August/2024

Advisors: Toacy Cavalcante de Oliveira
Raquel Mainardi Pillat Basso

Department: Systems Engineering and Computer Science

This dissertation introduces AATPAIS (Automated Acceptance Testing of Process-Aware Information Systems), a solution designed to address the complexities of testing PAIS. AATPAIS automates the generation and execution of test cases derived from BPMN process models and system specifications, utilizing Robot Framework for its robust RPA capabilities for screen automation. The system extracts human interaction points within process models, such as Start Events and User Tasks, and generates corresponding RPA scripts to perform acceptance tests.

The evaluation of AATPAIS involved assessing its effectiveness across a variety of process models. The results demonstrated a 100% path coverage in 15 out of 21 process models without further customization, showcasing the tool's efficiency in handling complex and dynamic process environments. However, certain process models revealed limitations due to gateway conditions and event triggers, indicating areas for further improvement.

Survey feedback highlighted AATPAIS's significant potential for improving productivity and streamlining testing processes. While users appreciated its usefulness, some concerns about ease-of-use and the learning curve were noted, suggesting the need for enhanced user training and interface refinement. Overall, AATPAIS presents a promising approach to automating acceptance testing in PAIS, with the potential for further development to enhance its capabilities and user experience.

Contents

List of Figures **xi**

List of Tables **xv**

List of Abbreviations **xvi**

1 Introduction **1**
- 1.1 Context and Motivation . . . 3
- 1.2 Objectives . . . 6
- 1.3 Methodology . . . 6
- 1.4 Organization . . . 10

2 Theoretical Foundation **11**
- 2.1 Introduction . . . 11
- 2.2 Software Development and Testing Life Cycles . . . 12
  - 2.2.1 Acceptance Testing . . . 13
  - 2.2.2 Model-Based Testing . . . 13
  - 2.2.3 Test Automation . . . 15
- 2.3 Business Process Model and Notation . . . 16
- 2.4 Robotic Process Automation . . . 18
- 2.5 Process-Aware Information Systems . . . 21
- 2.6 AKIP Process Automation Platform . . . 22

3 Related Work **25**
- 3.1 Introduction . . . 25
- 3.2 Automated Testing of a PAIS based on Process Models . . . 28
- 3.3 Threats to Validity . . . 29
- 3.4 Conclusion . . . 30

4 AATPAIS: Automated Acceptance Testing of PAIS **31**
- 4.1 Introduction . . . 31
- 4.2 Solution Overview . . . 31

4.3 Generated Test Suite . . . 34
4.4 Random Execution . . . 36
4.5 Pre-Planned Execution . . . 38
4.6 Final Considerations . . . 43

5 Evaluation **44**
5.1 Introduction . . . 44
5.2 Simulation . . . 45
5.2.1 Goals . . . 45
5.2.2 Metrics and Procedure . . . 46
5.2.3 Subjects . . . 47
5.2.4 Test Suite Generation Results . . . 48
5.2.5 Path Coverage Results . . . 50
5.2.6 Human Interaction Coverage Results . . . 51
5.3 Survey . . . 51
5.3.1 Planning . . . 51
5.3.2 Preparation . . . 52
5.3.3 Execution . . . 55
5.3.4 Results . . . 60
5.4 Discussion . . . 70
5.5 Threats to Validity . . . 72
5.5.1 Sampling Bias . . . 72
5.5.2 Limited Sample Size . . . 72
5.5.3 Self-Selection Bias . . . 72
5.5.4 Demonstration Bias . . . 73
5.5.5 Experience and Expertise Variability . . . 73
5.5.6 Recency of Technology Adoption . . . 73
5.5.7 Subjectivity in Responses . . . 73

6 Conclusion **75**
6.1 Introduction . . . 75
6.2 Contributions . . . 75
6.3 Limitations . . . 77
6.4 Future Work . . . 78

References **80**

A Evaluated Process Models **85**
A.1 Customer Feedback Repository . . . 86
A.2 AgileKIP's Travel Plan Tutorial Repository . . . 88

B Survey's Questionnaire **92**

List of Figures

1.1 The V-Model (adapted from [1]) . . . 3
1.2 An example of a BPMN process model. . . . 4
1.3 The Design Science Research Methodology (adapted from [2]) 7

2.1 An example of the visual representation a simple process with a parallel gateway. . . . 17
2.2 An example of the XML of a simple process with a parallel gateway. . 18
2.3 An example of a Test Suite using the Robot Framework syntax. . . . 19
2.4 An example of the keywords implementation using the Robot Framework syntax. . . . 20
2.5 The main components of a PAIS . . . 21
2.6 The AKIP Process Automation Platform overview [3] . . . 22
2.7 An example of a KIPApp Start-form entity file and its generated user interface. . . . 24

4.1 The AATPAIS solution schema . . . 32
4.2 Process model of the Travel Plan Process with Exclusive Gateways (`TP-XOR` in the Assessment). . . . 35
4.3 An example of the Robot Framework test file containing the higher-level keyword implementation. . . . 35
4.4 An example of the Robot Framework resources file containing the lower-level keyword implementation. . . . 36
4.5 An example of the Start Form of the Travel Plan Process with the fields filled by the RPA. . . . 37
4.6 Highlighting of the four possible paths in the Travel Plan Process with Exclusive Gateways. . . . 37
4.7 Process model of the Travel Plan Process with a Inclusive Gateway (`TP-OR` in the Assessment). . . . 38
4.8 The `TC_Planned` test case default structure. . . . 39
4.9 The planned execution of `TC_01` of the Travel Plan Process. . . . 39
4.10 A screenshot from the execution of `TC_01` of the Travel Plan Process. 40

4.11 The planned execution of `TC_02` of the Travel Plan Process. 40
4.12 The planned execution of `TC_03` of the Travel Plan Process. 41
4.13 The planned execution of `TC_04` of the Travel Plan Process. 41
4.14 A screenshot from the execution of `TC_04` of the Travel Plan Process. 42
4.15 The standard Robot Framework test execution report. 42

5.1 Process model of the Customer Feedback Process with Parallel and Exclusive Gateways (`CF-parallel-all-types-with-scalation` in the Assessment). 47
5.2 Process model of the Travel Plan Process with Parallel Gateways (`TP-AND` in the Assessment). 47
5.3 Survey Demographics - Question 2: Age Distribution 56
5.4 Survey Demographics - Question 3: Gender Distribution 57
5.5 Survey Demographics - Question 4: Highest Academic Degree 57
5.6 Survey Demographics - Question 5: Working Sector 58
5.7 Survey Demographics - Question 6: Primary Role 58
5.8 Survey Demographics - Question 7: IT Experience 59
5.9 Survey Demographics - Question 8: Experience with PAIS 59
5.10 Survey Demographics - Question 9: Experience with RPA Tools . . . 60
5.11 Perceived Usefulness and Ease-of-Use Questionnaire - Question 1 . . . 60
5.12 Perceived Usefulness and Ease-of-Use Questionnaire - Question 2 . . . 61
5.13 Perceived Usefulness and Ease-of-Use Questionnaire - Question 3 . . . 62
5.14 Perceived Usefulness and Ease-of-Use Questionnaire - Question 4 . . . 63
5.15 Perceived Usefulness and Ease-of-Use Questionnaire - Question 5 . . . 64
5.16 Perceived Usefulness and Ease-of-Use Questionnaire - Question 6 . . . 64
5.17 Perceived Usefulness and Ease-of-Use Questionnaire - Question 7 . . . 65
5.18 Perceived Usefulness and Ease-of-Use Questionnaire - Question 8 . . . 66
5.19 Perceived Usefulness and Ease-of-Use Questionnaire - Question 9 . . . 67
5.20 Perceived Usefulness and Ease-of-Use Questionnaire - Question 10 . . 67
5.21 Perceived Usefulness and Ease-of-Use Questionnaire - Question 11 . . 68
5.22 Perceived Usefulness and Ease-of-Use Questionnaire - Question 12 . . 69
5.23 Perceived Usefulness and Ease-of-Use Questionnaire - Question 13 . . 70

6.1 Model of the Customer Feedback Process with only a single User Task. 75

A.1 Process model of the Customer Feedback Process with all 3 types of feedback and exclusive gateway without escalation (`CF-exclusive-all-types-no-scalation` in the Evaluation). 86

A.2 Process model of the Customer Feedback Process with all 3 types of feedback and exclusive gateway with escalation (`CF-exclusive-all-types-with-scalation` in the Evaluation). . . . 86
A.3 Process model of the Customer Feedback Process with all 3 types of feedback and inclusive gateway without escalation (in the Evaluation). 86
A.4 Process model of the Customer Feedback Process with all 3 types of feedback and inclusive gateway with escalation (`CF-inclusive-all-types-with-scalation` in the Evaluation). . . . 87
A.5 Process model of the Customer Feedback Process with all 3 types of feedback and parallel gateway without escalation (`CF-parallel-all-types-no-scalation` in the Evaluation). 87
A.6 Process model of the Customer Feedback Process with all 3 types of feedback and parallel gateway with escalation (`CF-parallel-all-types-with-scalation` in the Evaluation). . . . 87
A.7 Process model of the Customer Feedback Process with only complaint as feedback and without escalation (`CF-only-complaint-no-scalation` in the Evaluation). 88
A.8 Process model of the Customer Feedback Process with only complaint as feedback and with escalation (`CF-only-complaint-with-scalation` in the Evaluation). 88
A.9 Process model of the Customer Feedback Process with only suggestion or compliment as feedback and without escalation (`CF-only-suggestion-compliment` in the Evaluation). 88
A.10 Process model of the Travel Plan Process with parallel gateway (`TP-AND` in the Evaluation). 88
A.11 Process model of the Travel Plan Process with a boundary event message (`TP-EMSG` in the Evaluation). 89
A.12 Process model of the Travel Plan Process with version 1 of entities (`TP-ENTITIES` in the Evaluation). 89
A.13 Process model of the Travel Plan Process with version 2 of entities (`TP-ENTITIES2` in the Evaluation). 89
A.14 Process model of the Travel Plan Process with version 3 of entities (`TP-ENTITIES3` in the Evaluation). 89
A.15 Process model of the Travel Plan Process with a loop (`TP-LOOP` in the Evaluation). 89
A.16 Process model of the Travel Plan Process with an inclusive gateway (`TP-OR` in the Evaluation). 90
A.17 Process model of the Travel Plan Process without gateways and a simple domain (`TP-SIMPLE` in the Evaluation). 90

A.18 Process model of the Travel Plan Process with a service task (`TP-SRV` in the Evaluation). 90
A.19 Process model of the Travel Plan Process with a timer boundary event (`TP-TIMER` in the Evaluation). 90
A.20 Process model of the Travel Plan Process with a simple validation in the domain (`TP-VAL` in the Evaluation). 91
A.21 Process model of the Travel Plan Process with exclusive gateways (`TP-XOR` in the Evaluation). 91

List of Tables

5.1 Test Suite Generation Average Time 48
5.2 Size and Complexity Measures and Path Coverage of Process Models in 30 Automated Random Executions 49
5.3 Questionnaire's Demographic Questions 54
5.4 Questionnaire's Perceived Usefulness and Ease-of-Use Questions . . . 55

6.1 Comparison of Manual and Automated Test Suite Execution Times . 76

List of Abbreviations

AATPAIS Automated Acceptance Testing of PAIS. 10, 44, 45

ATDD Acceptance Test-Driven Development. 13

BPEL WS-Business Process Execution Language. 18, 26

BPM Business Process Management. 44

BPMN Business Process Model and Notation. 3–8, 10–12, 14–18, 22, 25–34, 44, 45

BPMS Business Process Management Systems. 3, 28

IEEE Institute of Electrical and Electronics Engineers. 2

ISO International Organization for Standardization. 4, 16

JSON JavaScript Object Notation. 23

KIPApp Knowledge Intensive Process Application. 23

MBT Model-Based Testing. 4–7, 10–15, 18, 25–27, 29, 30

OMG Object Management Group. 4, 16, 17

PAIS Process-Aware Information Systems. 3–5, 7, 10–12, 15, 17, 18, 21, 22, 25, 26, 29–31, 34, 44

PDA Process-Driven Application. 3, 28

QA Quality Assurance. 16

RF Robot Framework. 19, 20, 31, 32

RPA Robotic Process Automation. 5–11, 13, 16, 18–20, 26, 29–32, 34

SDLC Software Development Life Cycle. 2, 12

SLR Systematic Literature Review. 16

STLC Software Testing Life Cycle. 12, 13, 16

SUT System Under Test. 10, 14, 31

UAT User-Acceptance Test. 2, 3, 6–8, 10–13, 15, 18, 20

WfMS Workflow Management Systems. 3

XML Extensible Markup Language. 17

Chapter 1

Introduction

Software quality is a systematic process that tries software products meet specified requirements and quality standards. According to the standards groups, it is "*the degree to which a software product meets established requirements*"[4] and "*the degree to which the system satisfies the stated and implied needs of its various stakeholders*"[5]. As described in a more aspirational definition, software quality is "*an ideal toward which we strive*"[1].

According to [1], there are different perspectives on what constitutes quality in software. The *User View* measures external product characteristics like defect density or reliability to understand the overall product quality. Users assess software quality based on the number and types of "*bugs*", categorizing them as minor, major, or catastrophic.

The *Manufacturing View* focuses on quality during production and post-delivery, evaluating whether the product was built correctly from the outset to avoid costly rework. This process-oriented view emphasizes adherence to good practices and conformance to established processes.

Both the *Manufacturing View* and the *User View* assess the product from an external standpoint. The *Manufacturing View* focuses on the production process rather than the final product itself, while the *User View* is concerned with the final product's usability and reliability after its delivery, without regard to the production process.

In contrast, the *Product View* peers inside and evaluates a product's inherent characteristics. This perspective, often advocated by software metrics experts, assumes that good internal quality indicators reflect conformance to sound production processes and lead to desirable external qualities such as usability, reliability, and maintainability. Developers, who scrutinize internal characteristics before delivery, often use the number and types of "*bugs*" as quality indicators, tracking **faults** identified during inspections of requirements, design, code, and testing to predict the final product's quality and the likelihood of **failures**.

The number and type of "*bugs*" detected serve as proxies for assessing software quality [1]. In software terminology, the term "*bug*" encompasses various meanings depending on the context. It can refer to a mistake in interpreting a requirement, a syntax error in code, or the cause of a system crash. The Institute of Electrical and Electronics Engineers (IEEE) has established a standard terminology for describing "*bugs*" in software products.

According to IEEE Standard 729-1983, a **fault** is "*an accidental condition that causes a functional unit to fail to perform its required function*", leading to **failures**. A **failure** is "*an event in which a system or system component does not perform a required function within specified limits*"[6]. **Failures** occur during the execution of software due to **faults** introduced during requirements gathering, design, or coding phases.

Faults originate from human *errors* during software activities. For example, a requirements analyst might misunderstand a user's request, or a designer might misinterpret a requirement, resulting in a flawed design. This design **fault** can lead to additional **faults** in the code and documentation. Consequently, a single *error* can generate multiple **faults**, which can exist in any development or maintenance product [1].

A **failure** is a departure from the system's required behavior. It can be discovered before or after system delivery. Given that the requirements documentation can contain **faults** committed by the analyst, a **failure** may indicate that the system is not performing as required, even though it may be performing as specified [1].

Therefore, a robust testing strategy with high test coverage is crucial for certifying the overall quality of developed software. When a **failure** is discovered during testing, prior to product delivery, it serves as an internal direct measure of the quality of the software development process, as it is assessed by analysts, developers, and testers.

Conversely, if a **failure** is found after product delivery, it becomes an external indirect measure of the software's quality, as experienced by the user. Specifically, a testing strategy that evaluates the final product by simulating the end-user experience before release is particularly effective in safeguarding and enhancing the perceived quality of the software.

The V-model[1] (Fig. 1.1) suggests that each Development Activity within the Software Development Life Cycle (SDLC) aligns with a corresponding Test Activity. This model situates Acceptance Testing, or User-Acceptance Test (UAT) as the final phase in the SDLC and illustrates the direct correlation between UAT and Requirements Analysis.

As defined by [7], UAT is "*testing conducted to determine whether a system satisfies its acceptance criteria and to enable the customer to determine whether to*

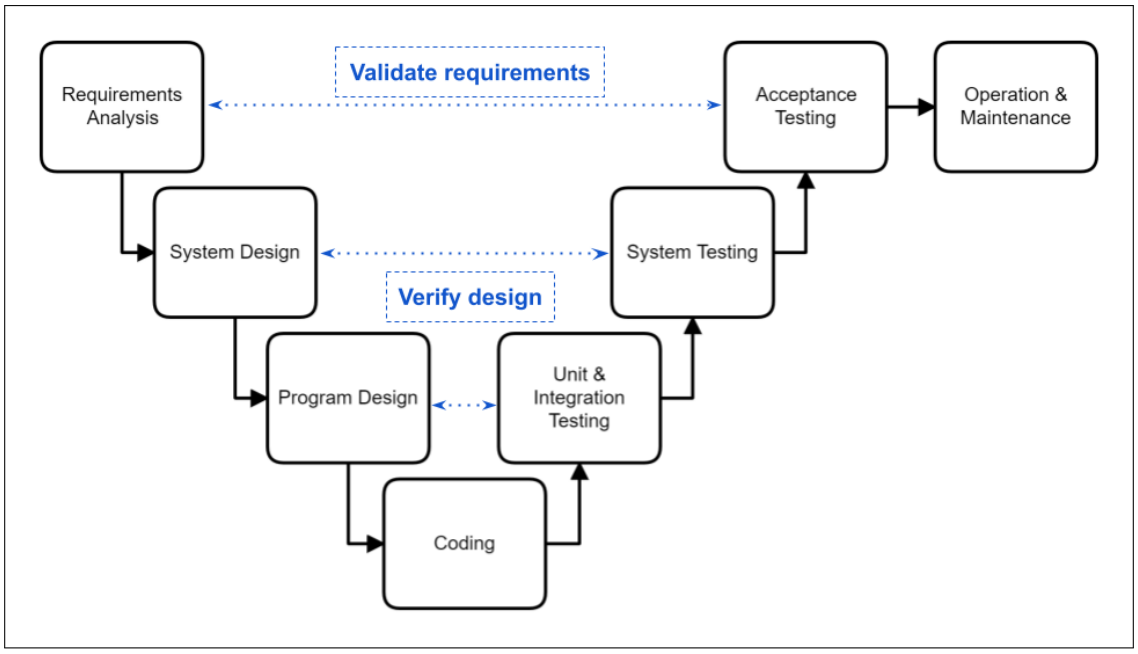


Figure 1.1: The V-Model (adapted from [1])

accept the system". Following the V-model approach involves designing and executing UATs with careful consideration of the artifacts generated during Requirements Gathering.

Therefore, UAT is a vital step for guaranteeing software quality, as it focuses on validating that the system satisfies the expectations of end-users or stakeholders and, according to the V-model, meets the previously specified requirements.

1.1 Context and Motivation

Process-Aware Information Systems (PAIS) are software systems designed to automate and support business processes by integrating process modeling, execution, monitoring, and analysis [8, 9]. PAISs are also referred to as Workflow Management Systems (WfMS), Business Process Management Systems (BPMS) or Process-Driven Application (PDA). A more detailed explanation about this type of software systems can be seen in Section 2.5.

The *AgileKIP Process Automation Platform* [10] is an example of a data-driven tool for generating a low-code PAIS that performs process orchestration. It is an open-source project originated from the academic research conducted by the AgileKIP Group[1], devoted to facilitate Process/Workflow Automation initiatives based on code generation techniques. An in-depth explanation about this platform can be found in Section 2.6.

PAIS are based on business process models such as the Business Process Model

[1]https://agilekip.github.io/pap-documentation/about

and Notation (BPMN), a standardized meta-model and notation maintained by the Object Management Group (OMG) [11] and by the International Organization for Standardization (ISO) [12]. A process model typically features multiple execution paths, options, and dependencies among activities, including decision points, parallel execution, and synchronization, as can be found in Figure 1.2. A comprehensive explanation about this notation can be found in Section 2.3.

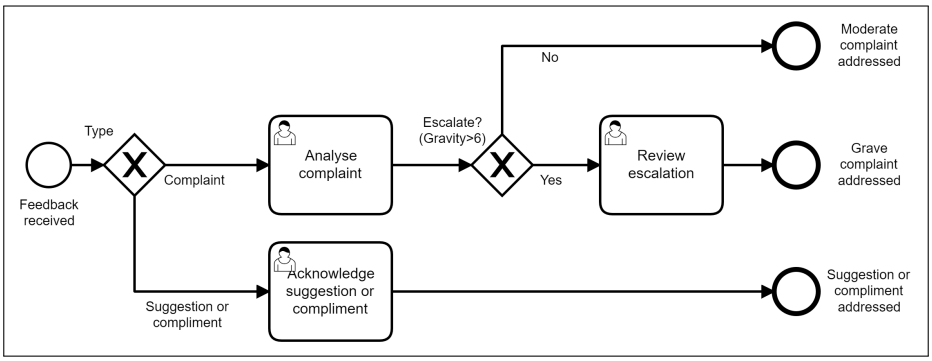


Figure 1.2: An example of a BPMN process model.

Given that a process model can have numerous execution paths and decision points, ensuring software quality in PAIS is challenging due to their inherent complexity and dynamism. This complexity makes it difficult to cover all scenarios manually and enforce a thorough test coverage. Identifying all possible paths within a process model can be difficult, even for domain experts.

For example, in [13] it is illustrated that domain experts struggle to manually determine all conceivable paths for processes with fewer than 20 possible flows, achieving an average precision and recall of 0.76 (75.75%) and 0.78 (77.75%), respectively. Furthermore, a PAIS often interfaces with external systems and human interactions, introducing additional variability. Maintaining high test coverage becomes increasingly difficult as the system evolves, requiring significant time and resources to keep test suites up-to-date and effective. In [14], a BPM expert was surprised when a tool identified more test paths than initially anticipated.

A key aspect of a BPMN-based PAIS is its capability to facilitate system evolution through changes in the process model. However, updates to process models complicate thorough test coverage, as changes can affect interconnected elements, requiring extensive revalidation to ensure quality and reliability. Experts in the field describe the current practice of keeping test suites updated for BPMN-based applications as "error-prone and time-consuming", attributing this to the frequent alterations in process models [14].

Given that a PAIS is fundamentally built upon process models [8], particularly BPMN [11], it is possible to employ a model-based strategy for test case generation [15, 16]. This systematic technique, known as Model-Based Testing (MBT), aims to

ensure comprehensive test coverage by meticulously analyzing all conceivable paths and dependencies among activities and entities within the model. Consequently, MBT effectively addresses the challenge of achieving comprehensive test coverage in complex and dynamic environments. MBT is further explained in Section 2.2.2.

A key advantage of MBT is its ability to automate test case generation [17]. Automatically generated test suites improve the quality of tests by reducing the possibility of human errors, thereby enhancing reliability and consistency. Considering that the BPMN standard explicitly delineates human interaction points within a process, it is feasible to compile a comprehensive set of these interactions in the generated test cases from the process models. As a result, the MBT approach can enhance the quality of tests by automating the generation of acceptance test cases [18] of a BPMN-based PAIS.

In addition to the challenges associated with the discovery and planning of acceptance test cases for PAIS, executing them poses its own set of difficulties. The sheer volume of potential combinations makes it extremely challenging for a human to exhaustively execute all test cases.

Robotic Process Automation (RPA) involves using software robots ("bots") to interact with the system's user interface, simulate user actions, and compare expected outcomes with actual results. By strategically implementing RPA, it becomes feasible to automate the execution of these user interactions within an acceptance test suite. This deployment of RPA further amplifies the efficiency and consistency of the overall testing procedure, offering significant benefits to the testing process as a whole [19]. RPA is further explained in Section 2.4.

According to [20], numerous companies are embracing automated testing within their pipelines, since it is seen as a strategic capability that offers a competitive advantage. Adopting acceptance criteria-driven tests is seen as a progressive step to further improve software quality [21]. MBT supports the automated generation of acceptance test cases, and RPA enables their automated execution.

Automated test generation and execution can significantly reduce the time and effort required to create and conduct test cases manually, thereby freeing up testers' time to concentrate on more intricate and high-value tasks, while facilitating the implementation of a regression testing strategy. Test automation is further discussed in Section 2.2.3.

Automating software tests reduces manual effort, allows for test repeatability, and facilitates faster feedback cycles [17]. Leveraging test automation in BPMN-based PAIS ensures consistent and repeatable testing outcomes, enhancing efficiency and reliability, ultimately enhancing software quality.

1.2 Objectives

The primary objective of this work is to propose and evaluate a MBT strategy that leverages BPMN process models and system requirements specifications to generate keyword-driven RPA scripts. These scripts automate a comprehensive UAT procedure, aspiring to cover all conceivable paths within the BPMN of the PAIS under evaluation. By leveraging RPA, this proposal aims to enhance the efficiency and consistency of the acceptance testing procedure, thereby delivering substantial benefits to the overall software testing process.

To further elaborate on this primary objective, the following specific goals have been identified:

1. Automated Generation of a Model-Based Acceptance Test Suite
    - 1.1 Formulate a systematic approach for generating test cases from BPMN models and system requirements specifications.
    - 1.2 Enhance the comprehensiveness of test coverage by effectively managing the complexity and variability of process models.
2. Automated Execution of Acceptance Tests
    - 2.1 Implement a framework that uses a screen automation tool, i.e., RPA, to automate the execution of a comprehensive Acceptance Testing strategy.
    - 2.2 Enhance the testing process by automating repetitive and time-consuming tasks that are prone to errors.
    - 2.3 Formulate an approach that enables the tester to steer and plan the test cases execution.
3. Evaluate the Proposed Solution
    - 3.1 Conduct an assessment to simulate the practical application and effectiveness of the proposed solution across different process models.
    - 3.2 Conduct a survey to evaluate the perceived usefulness and ease-of-use of the proposed solution.

1.3 Methodology

The current work followed the Design Science Research Methodology for Information Systems Research, proposed by [2]. It is typically composed of the following 6 steps, with possible recurrent iterations between steps 2 to 6, as seen in Figure 1.3.

The first step, **Identify Problem and Motivate**, was defined as follows:

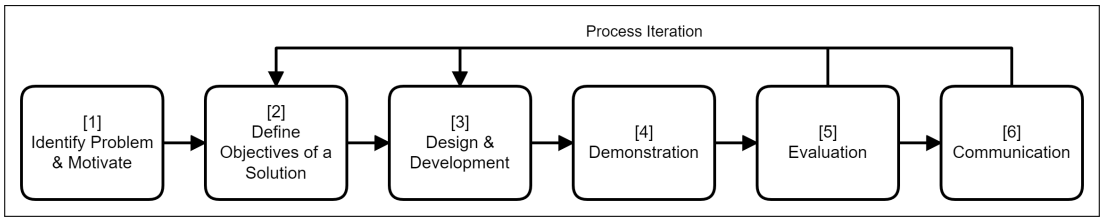


Figure 1.3: The Design Science Research Methodology (adapted from [2])

1. Given the challenges associated with achieving comprehensive test coverage in a PAIS, and that the process model is a core artifact in these systems, this research focuses on developing an innovative MBT strategy. This strategy is designed to automate the generation and execution of UAT using BPMN models and system requirements specifications, with the goal of improving the efficiency, effectiveness, and reliability of the testing process.

The **first iteration of steps 2 to 6** had an exploratory nature, and occurred as follows:

2. **Define Objectives of a Solution:** Develop a solution proposal to demonstrate the feasibility of the MBT strategy.

3. **Design and Development:** Conduct an *ad-hoc* literature review to identify existing techniques and methodologies related to MBT and to the Automated Testing of a PAIS. The review aimed to uncover gaps in current research and to build a foundation for the proposed MBT strategy. The findings from this review were instrumental in shaping the design and implementation of the proposed solution.

4. **Demonstration:** Develop a prototype that generates keyword-driven RPA scripts from BPMN models and system requirements specifications. At this stage, it was only able to execute the test cases randomly and there wasn't much consideration to variable naming and reusability. A set of process models were designed to give context.

5. **Evaluation:** The proof of concept was validated through a series of tests on sample process models.

6. **Communication:** The results were documented in an article that was accepted and presented as a Regular Full Paper at the *19th International Conference on Web Information Systems and Technologies (WEBIST 2023)*. The article was **awarded Best Student Paper** and, as a result, it was invited for publication at the *Springer Nature Lecture Notes on Computer Science*. These results were also presented at the industry conference RoboCon 2024[2].

[2]https://robocon.io/

The second iteration of steps 2 to 6 occurred as follows:

2. **Define Objectives of a Solution:** It was made clear at this point that the final solution had to be able to consider both random and pre-planned executions to ensure comprehensive test coverage and to enable a regression strategy. A systematic mapping study was initially planned to further explore and validate the research topic, in order to identify existing techniques and methodologies related to BPMN testing and formal verification and certify the innovative nature of the proposed solution. However, a recently published systematic literature review by [22] provided ample context and background on the topic, making an additional systematic mapping study redundant. This is further elaborated in Section 3.1.

3. **Design and Development:** Modifications to the initially developed solution were identified to accommodate both random and pre-planned executions. These modifications included:

    - Generalize variables throughout the test suite, in different levels of keywords;
    - Implement variables as 'Parameters' in the keywords, to enable the the steering of Test Cases, increasing the reusability of keywords;
    - Implement 'Tags' to enable executing multiple Test Cases at once in the pre-planned executions strategy;

4. **Demonstration:** The developed solution considered both random and pre-planned executions through the implementation of tags and reusable parameterized keywords. It was able to generate RPA scripts that mimic user interactions within the process models, thereby automating the UAT procedure.

5. **Evaluation:** The final solution was evaluated through a series of assessments using both the previously designed process models and a set from the AgileKIP open-source project repository. The effectiveness of the solution was measured by the coverage and accuracy of the generated test cases, as well as the reduction in manual effort required for test cases identification and execution. Additionally, a survey was conducted to gather feedback from developers, testers and stakeholders on the usefulness and ease-of-use of the proposed automated testing framework.

6. **Communication:** The present dissertation was written to document the entire research process, from the initial literature review and proposal to the implementation and evaluation of the final solution. It includes detailed descriptions of the methodologies used, the results obtained, and the conclusions

drawn from the study. This comprehensive documentation aims to provide a clear and thorough account of the research, its findings, and its contributions to the field of software testing in PAIS. Following the Best Student Paper award received at WEBIST 2023, the invitation to publish in the Springer Nature Lecture Notes on Computer Science was accepted, with the publication currently pending. This final implementation was also presented at the industry conference CamundaCon 2024 Europe[3].

The following summarizes the work conducted, published, and presented, as well as the awards received during the period of development of the Master's research:

- The article "*Supporting the Automated Generation of Acceptance Tests of Process-Aware Information Systems*" was **awarded Best Student Paper** at the *19th International Conference on Web Information Systems and Technologies (WEBIST 2023)* [23].
- The article "*AATPAIS: Automated Generation and RPA-backed Execution of User Acceptance Tests of a BPMN-based Process-Aware Information System*" submitted at the *Springer Nature Lecture Notes on Computer Science* has been approved, with its publication pending at this moment.
- Invited to work as a **Research Fellow at the University of Waterloo** for a period of six months under the Emerging Leaders in the Americas Program (ELAP) Scholarship, provided by the Government of Canada.
- The research findings were invited to be presented at two industry conferences:
  - *RoboCon 2024*, organized by the Robot Framework Foundation, the maintainer of the open-source RPA used in this implementation;
  - *CamundaCon 2024 Europe*, organized by Camunda, the maintainer of the open-source Process Engine used in this implementation;
- The article "*O Sistema Portuário Brasileiro: Um Panorama dos Processos Informacionais de Importação e Exportação de Cargas Conteinerizadas*" was published in "Revista Gestão & Tecnologia" [24];
- The article "*Análise de Semântica Latente (LSA) Aplicada a Projetos de Lei*" was published in the Technical Report PESC-782 "Explorações em Mineração de Texto".

[3]https://www.camundacon.com/

1.4 Organization

The current work proposes a MBT strategy that uses BPMN models and system requirements specifications to generate a keyword-driven RPA script, automating a comprehensive UAT procedure. It is organized into five further chapters.

The **Theoretical Foundation** chapter lays the groundwork for the research by exploring relevant theoretical concepts and frameworks. It begins with an exploration of Software Development and Testing Life Cycles, which includes a detailed discussion on Acceptance Testing, Model-Based Testing, and Test Automation. This chapter provides an overview of BPMN and RPA, highlighting its relevance and application within the context of the study, and also delves into PAIS. Finally, it introduces the AKIP Process Automation Platform, the chosen PAIS.

Existing literature and research related to the topic are reviewed at the **Related Work**. It aims to identify gaps in the current knowledge and establish the context for the proposed solution. By comparing various approaches and findings, this chapter positions the present work within the broader research landscape.

This solution, the **AATPAIS** framework, is then introduced as the core contribution of the research. It begins with an Introduction to the framework, followed by a detailed description of the System Under Test (SUT). The Solution Overview provides a high-level view of how AATPAIS functions. Subsequent sections cover the Generated Test Suite, the Output from Random Execution, and methods for Customizing Scripts for Pre-Planned Execution.

The **Evaluation** chapter assesses the effectiveness of the proposed solution. It starts with an Introduction to the evaluation process and outlines the Assessment Strategy. Detailed evaluations of Random Execution and Pre-Planned Execution are presented. A Survey section describes the metrics, methodology, and results of a survey conducted to gather feedback from testers and developers. The Discussion offers insights into the findings, and Threats to Validity are acknowledged and addressed.

The **Conclusion** provides a summary of the research findings and their implications. Contributions of the research are highlighted, along with its Limitations. Finally, Future Work outlines potential directions for further research and development in this area.

Chapter 2

Theoretical Foundation

2.1 Introduction

Software development and testing encompass extensive domains within software engineering, each with a multitude of frameworks and approaches. Given the focus of the current work on the automated generation and execution of acceptance tests for BPMN-based PAIS using RPA, it is crucial to clearly define these concepts within its scope.

Utilizing MBT techniques in a BPMN-based PAIS enables the derivation of a UAT suite directly from the process model. The process model encapsulates the expected interactions between end-users and the system, facilitating the generation of relevant test cases. These test cases then support UAT, allowing users to validate whether the system meets their expectations and requirements. Additionally, this approach presents an excellent opportunity to automate the testing of user interfaces using RPA [19].

Automated User-Acceptance Test of BPMN-based PAIS encompasses several crucial concepts: MBT, Acceptance Testing, and Test Automation.

MBT plays a pivotal role in this strategy, enabling the automatic derivation of test cases from an explicit abstract model of the system under test, in this case, the BPMN. By interpreting the model's behavior as the intended system behavior, MBT ensures comprehensive test coverage and aids in understanding requirements, documenting specifications, and generating test cases.

Acceptance Testing is a vital component of UAT for BPMN-based PAIS. It focuses on validating that the system meets specified requirements and satisfies end-user or stakeholder expectations. Through acceptance testing, organizations can ensure that the implemented system aligns with desired functionality, user experience, and overall business objectives. This process provides stakeholders with an opportunity to give feedback, identify issues, and confirm that the system meets

their acceptance criteria.

Test automation is key to achieving efficiency and reliability in UAT. By automating the execution of test cases, data input, and result verification, organizations can streamline the testing process. Test automation reduces manual effort, allows for test repeatability, and facilitates faster feedback cycles [25]. Leveraging test automation in BPMN-based PAIS ensures consistent and repeatable testing outcomes, enhancing overall efficiency and reliability.

In summary, a comprehensive strategy for the automated UAT of BPMN-based PAIS involves leveraging MBT for test case derivation, conducting Acceptance Testing to ensure system compliance with requirements, and embracing Test Automation to streamline the testing process. By incorporating these concepts, organizations can enhance the quality, efficiency, and reliability of UAT in BPMN-based PAIS.

2.2 Software Development and Testing Life Cycles

Software Development Life Cycle (SDLC) and Software Testing Life Cycle (STLC) are essential processes in software development. SDLC is a methodological framework in software engineering that encompasses the systematic stages of developing software, including requirements gathering, design, coding, testing, deployment, and maintenance.

The primary objectives of SDLC framework are to maximize software quality, manage project timelines and costs effectively, and minimize potential risks associated with software development. By employing SDLC, organizations aim to deliver software products that meet specified requirements, ensure functional precision, and provide value to end-users.

Conversely, STLC focuses specifically on testing activities within SDLC, involving the planning, designing, executing, and evaluating of tests to verify software functionality and quality. Both SDLC and STLC work in tandem, with STLC ensuring that the software meets requirements through comprehensive testing. Together, they facilitate structured and efficient software development, resulting in high-quality products that meet stakeholder expectations.

According to [26], software testing consists of:

- Unit Test: testing new functionalities for correctness, typically against function and code coverage requirements;
- Integration Test: designed to test the unit's interfaces and the interactions among the units;
- System Test: checks for defects in system functionalities, typically using a black-box approach;

- Acceptance Test: tests the software against the user requirements, both functional and non-functional, to demonstrate the software readiness for operational use.

Although testing is a crucial aspect of STLC [25], most existing testing strategies focus more on the developer rather than the end-user [26]

2.2.1 Acceptance Testing

Acceptance Tests, also referred as User-Acceptance Test when conducted with the presence of or even by the customer, evaluate whether a set of features work from the customer's perspective, ensuring their satisfaction with the final product [27]. This process enhances the customer's confidence in accepting the system [25].

One key distinction among test modalities is that Unit and Integration Tests are typically designed and written by developers, System Tests by developers or testers, while Acceptance Tests, especially UATs, are usually designed by customers and possibly written by them with the assistance of a developer or tester [27]. Traditionally, end-users enumerate a set of acceptance test cases that cover the "*major functions, user interface, and capabilities in handling invalid input and exceptions in operation*," with the primary objective of evaluating the system's readiness for operational use [26].

Two literature reviews on Acceptance Testing were identified. The first, from 2008 [28], was very specific, focusing on the automation of Acceptance Testing. Although it concentrated on test automation, RPA was not mentioned, likely due to the limited availability of the technology at that time.

In 2016, [29] identified 26 relevant papers using the most significant indexes. An important finding of this review was the prevalence of inconsistent and incomplete acceptance tests, which, nonetheless, are still considered advantageous. This highlights the need to compare the outcomes of Acceptance Test-Driven Development (ATDD) research with those of traditional manual tests.

Additionally, in [29]'s literature review about the state of empirical research within the domain of Acceptance Testing, it was noted that the open-source standalone wiki and integrated acceptance testing framework FitNesse, based on the Framework for Integrated Testing (Fit), is the most prominent tool in the research papers. Furthermore, no empirical studies were reported to use the Robot Framework for Acceptance Testing.

2.2.2 Model-Based Testing

Model-Based Testing (MBT) is a specialized approach that relies on the use of explicit behavior models. These models encapsulate the anticipated behaviors of the

System Under Test (SUT) or, in some cases, its corresponding environment. Test cases are derived from these behavioral representations or a combination thereof, and are subsequently executed on the designated SUT [30].

MBT represents a testing paradigm in which test cases are wholly or partially generated from a model. It is a broad subject, with a wide array of applications developed around UML [15, 31, 32]. For successful implementation, it requires that the software's behavioral or structural characteristics be explicitly defined by models crafted with well-defined rules. These models can take various forms, such as formal models, finite state machines, or UML diagrams [15].

Despite some contexts conflating MBT with Test Case Generation, it is crucial to clarify the difference between these concepts to enhance the understanding of MBT. MBT involves using models developed during the software development process, which are adapted by the testing team to automatically generate test case sets. In contrast, Test Case Generation is merely one task within the broader testing process, and it may or may not involve the use of formal software models [31].

The benefits of MBT, according to [31]:

1. Lower cost and effort for testing planning/execution and shorter testing schedule;

2. Improvement of the final product quality, because the models are used as an oracle for testing;

3. Testing process can be automated;

4. Ease of communication between the development and testing teams;

5. Capacity of automatically generating and running large sets of useful and non-repetitive (non-redundant) tests;

6. Ease of updating the test cases set after the software artifacts used to build the software model changes;

7. Capacity of evaluating regression testing scenarios.

Two systematic reviews of the literature on MBT were independently conducted around the same period in 2007 [31] and 2009 [32]. The work by [31] identified 219 distinct MBT approaches, classifying them based on 29 different attributes, such as the use of UML models, the focus on functional or non-functional testing, the testing level (system/integration/unit/regression testing), the degree of automation, and various other attributes related to the model, test generation process, and software development environment. Although BPMN was not referenced in this review, the

study presented 12 risk factors that could influence the use of MBT strategies in software projects, which could be applied to a BPMN-oriented MBT strategy.

Similarly, [32] did not mention BPMN as a model for MBT approaches. This omission can be partly explained by the fact that BPMN gained significant traction only after the release of the BPMN 2.0 standard in 2010 [11], and these reviews were conducted prior to that period.

In [15]'s overview of recent advances in MBT, published in 2016, it is noted that BPMN began to emerge for modeling business applications and could be further utilized for describing test cases.

An automated regression testing framework designed for business process models conforming to the BPMN 2.0 standard is presented in [33]. This framework captures execution snapshots of these models in the production environment to achieve two main objectives: automatically generating regression test cases and enabling controlled and automated isolation of business process execution from external dependencies.

By leveraging these capabilities, the framework provides an efficient solution for conducting regression testing on BPMN-based PAIS, specifically jBPM[1], thereby enhancing the reliability and maintainability of the tested systems. Similarly, the work shown in [34] focuses on service-based processes, presenting strategies that improve the testing of such systems.

However, it is important to note that the strategies proposed by both [33] and [34] do not specifically address UAT in the context of BPMN-based PAIS. [33]'s approach primarily focuses on regression testing using mocking with jUnit, while [34] centers on service-based processes. Consequently, there appears to be a research gap in the domain of UAT specifically tailored to BPMN-based PAIS. Further investigation and research are needed to address this gap and develop effective UAT methodologies for BPMN-based PAIS.

2.2.3 Test Automation

The terms "automated software testing" and "software testing automation" are often used interchangeably, but they have subtle differences.

"Automated software testing" specifically refers to using automation tools and scripts to perform various types of testing. The goal is to reduce the time and effort required for repetitive and predictable tasks, thereby improving software quality. It involves executing specific sets of tests (e.g., regression tests) via automation rather than manually. However, test planning in this context is usually done manually [17].

"Software testing automation," on the other hand, is a broader concept. It

[1] https://www.jbpm.org/

includes not only automating the execution of test cases but also managing test data and test environments. The aim is to improve the efficiency and effectiveness of software testing by reducing time and effort while enhancing the accuracy and consistency of results [17].

In other words, software testing automation encompasses all aspects of automating the STLC, including test execution, management, and reporting. It involves automating the process of tracking and managing different tests, not just the execution itself. It also includes various test strategies within the overall Quality Assurance (QA) process.

According to [35], test automation was a popular research activity, addressed by 34.2% of the literature in that Systematic Literature Review (SLR). In the same article, 62.0% of the papers related to automated testing provided full automation for the test approaches (tools or scripts) presented, while 25.3% were semi-automated, involving both manual and automated aspects. Thus, 87.3% of the web application testing solutions listed involved some level of automation. However, as noted by [19], automated user interface testing using RPA is still a topic sparsely covered in the scientific literature.

Automating acceptance tests offers significant potential for enhancing development efficiency. It is estimated that, in many projects, 50% of the total cost of software development is allocated to software testing [25], and automation can reduce these costs by up to 70% [36]. However, as observed by [28], it is crucial to acknowledge the costs associated with writing and maintaining automated tests despite the promised cost savings. Therefore, a thorough evaluation of the potential benefits versus the associated costs is essential before implementation.

2.3 Business Process Model and Notation

Business Process Model and Notation (BPMN)[2] is a standardized graphical notation for modeling business processes, developed by the Object Management Group (OMG) and first introduced in 2004. The main goal of BPMN was to "*standardize a business process model and notation in the face of many different modeling notations and viewpoints*" [11]. Since the release **BPMN 2.0** standard in 2010, it has been widely adopted in the industry due to its clarity, versatility, and ability to bridge the gap between stakeholders, eventually becoming an ISO Standard in 2013 [12].

BPMN offers a visual representation that enables organizations to document, analyze, and communicate their business processes effectively. It provides "*a straightforward way to convey process information to other business users, process implementers, customers, and suppliers*" [11].

[2]https://www.omg.org/bpmn/

BPMN incorporates various symbols and elements to represent different aspects of a process. In BPMN diagrams, such as the one shown in Figure 2.1, a *Start Event* is represented by a simple circle, indicating where a process begins, while an *End Event* is depicted as a circle with a thick border, marking the conclusion of a process. A *User Task* is illustrated as a rectangular box with rounded corners and a small "human" icon inside, signifying an activity performed by a human participant. Given the focus of the present research on automating human interactions within a business process, User Tasks represent a primary element of interest. The complete specification can be found in OMG's website[3].

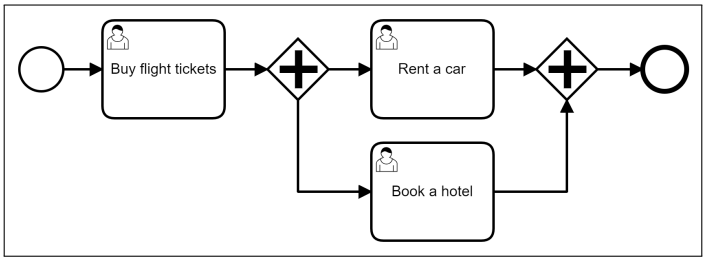


Figure 2.1: An example of the visual representation a simple process with a parallel gateway.

A BPMN is defined by an XML Schema underneath its visual representation. As an example, the XML definition of the process depicted in Figure 2.1 can is shown in Figure 2.2. Rows 3 to 39 (Fig. 2.2), within the tag `<bpmn:process>`,define the process flows and the interconnections between symbols and elements. For instance, rows 29 to 33 and 34 to 38 specify the parallel gateways, represented as diamond shapes with a "+" symbol inside, along with their respective incoming and outgoing flows. From row 40 onwards, within the tag `bpmndi:BPMNDiagram`, the visual representation of the process is defined.

BPMN plays a crucial role in PAIS by providing a standardized language for modeling and executing business processes. By utilizing BPMN, PAIS can visually and systematically represent complex process logic, control flow, and data dependencies. According to [34], the adoption of BPMN as a universal modeling language facilitates effective communication among all project participants and encourages the reuse of existing editors and repositories. Although their work focused on service-based processes, the same argument applies to human-oriented processes and their User Acceptance Tests.

Human interaction points within a business process are crucial for assessing software quality. These interactions occur when tasks require human input or decision-making. BPMN provides specific elements to model these interactions, such as *User Tasks*, which "need to be rendered on user interfaces like forms, clients, portlets,

[3]https://www.omg.org/spec/BPMN/2.0/PDF

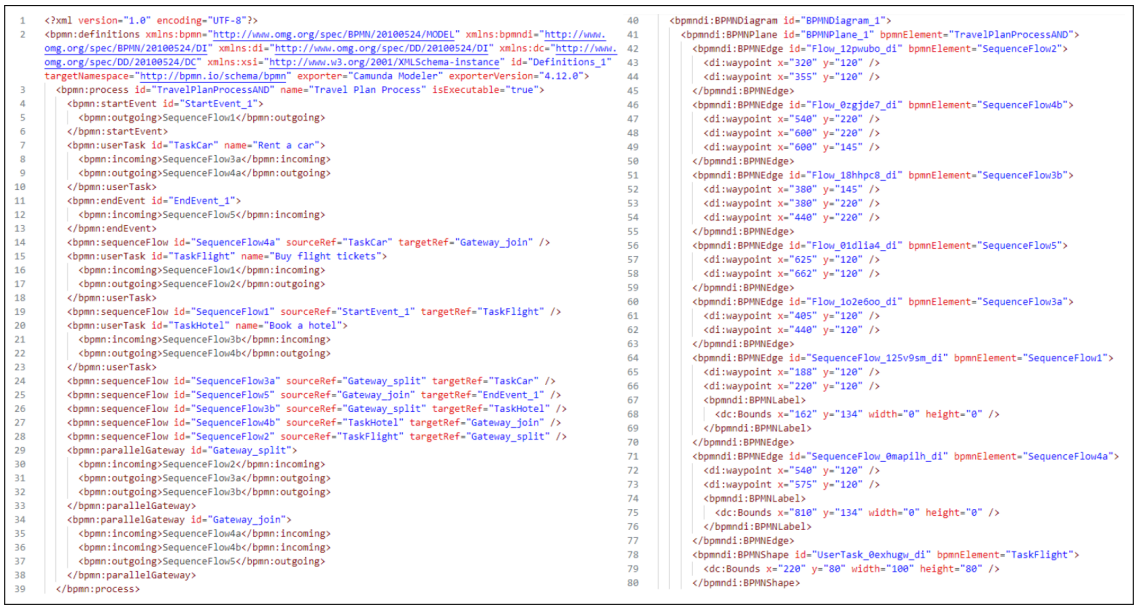


Figure 2.2: An example of the XML of a simple process with a parallel gateway.

etc" [11]. Specifically in KIPApps [3], both Start Events and User Tasks are human interaction points, each associated with forms.

UATs are essential in BPMN-based PAIS to ensure the quality and usability of implemented processes. UATs focus on validating the process models and their execution within the system. By aligning UATs with BPMN diagrams, organizations can assess whether the implemented processes adhere to the intended logic and achieve the desired outcomes. This validation helps identify and address discrepancies or issues early on, enhancing user satisfaction and system reliability.

Related to utilizing process models to derive test cases, BPEL (WS-Business Process Execution Language) and BPMN are the most common forms of workflow representation used for it [37]. Since BPMN serves as the underlying model for these systems, MBT leverages these process models to automate the testing process. This approach enables comprehensive coverage of user interactions and early detection of potential issues. By utilizing MBT, organizations can improve the effectiveness and efficiency of UATs, ensuring robust testing and higher software quality within BPMN-based PAIS.

2.4 Robotic Process Automation

Robotic Process Automation (RPA) involves using software robots or "bots" to automate repetitive, rule-based tasks within business processes. These bots mimic human interactions with various software systems to perform tasks such as data entry, form filling, and screen navigation. The concept originated from screen scraping tools and macros, which evolved into more advanced automation capabilities.

One notable open-source RPA solution is the Robot Framework (RF)[4]. RF is a generic keyword-driven test automation framework that supports RPA and other types of software testing. It features a simple and readable syntax, making it accessible for both technical and non-technical users. RF allows the creation of test cases using keywords, which can be extended through libraries or custom implementations.

For didactic purposes, a simple example is shown in the "*Get Started*" section of the website[5], containing a single test suite for a user login scenario, using a mocked backend API for user management. The test suite `TestSuite.robot` (Fig 2.3), contains two test cases: "**Login User with Password**" and "**Denied Login with Wrong Password**". These test cases calls keywords from the resource file `keywords.resource` (Fig. 2.4), which contains the keyword definitions.

```
1   *** Settings ***
2   Documentation     A test suite for valid login.
3   ...
4   ...               Keywords are imported from the resource file
5   Resource          keywords.resource
6   Default Tags      positive
7
8   *** Test Cases ***
9   Login User with Password
10      Connect to Server
11      Login User            ironman    1234567890
12      Verify Valid Login    Tony Stark
13      [Teardown]    Close Server Connection
14
15  Denied Login with Wrong Password
16      [Tags]    negative
17      Connect to Server
18      Run Keyword And Expect Error    *Invalid Password    Login User    ironman    123
19      Verify Unauthorised Access
20      [Teardown]    Close Server Connection
```

Figure 2.3: An example of a Test Suite using the Robot Framework syntax.

[4]https://robotframework.org/
[5]https://robotframework.org/

```
*** Settings ***
Documentation     This is a resource file, that can contain variables and keywords.
...               Keywords defined here can be used where this keywords.resource in loaded.
Library           CustomLibrary.py


*** Keywords ***
Connect to Server
    Connect    fe80::aede:48ff:fe00:1122

Close Server Connection
    Disconnect

Login User
    [Arguments]    ${login}    ${password}
    Set Login Name    ${login}
    Set Password    ${password}
    Execute Login

Verify Valid Login
    [Arguments]    ${exp_full_name}
    ${version}=    Get Server Version
    Should Not Be Empty    ${version}
    ${name}=    Get User Name
    Should Be Equal    ${name}    ${exp_full_name}

Verify Unauthorised Access
    Run Keyword And Expect Error    PermissionError*    Get Server Version

Login Admin
    [Documentation]    'Login Admin' is a Keyword.
    ...                It calls 'Login User' from 'CustomLibrary.py'
    Login User    admin    @RBTFRMWRK@
    Verify Valid Login    Administrator
```

Figure 2.4: An example of the keywords implementation using the Robot Framework syntax.

RF offers robust capabilities for creating, customizing, and reusing keywords, which are essential for building flexible and maintainable test suites. Users can define their own keywords, which can then be utilized across multiple test cases, promoting code reuse and reducing redundancy. Additionally, RF's ecosystem includes a rich collection of Standard and community-contributed libraries[6] that extend its functionality. These libraries provide a wide range of pre-built keywords and integrations, allowing testers to easily connect with various tools and technologies. Given its open-source nature, the community-driven libraries ensures continuous enhancement and adaptation to new testing challenges, making Robot Framework a versatile and powerful tool for automated testing.

RPA is beneficial in UAT by automating repetitive and manual testing activities [19]. Since UATs require end-users or stakeholders to validate a system's functionality, user experience, and compliance with requirements, RPA can streamline and accelerate this testing cycle. It automates the execution of test cases, data input, and result verification. Bots interact with the system's user interface, simulate user actions, and compare expected outcomes with actual results. This automation reduces manual effort, increases test coverage, and enhances the efficiency of the UAT process. However, it is essential to emphasize that the current solution is not de-

[6]https://docs.robotframework.org/docs/different_libraries/overview

signed to replace the end user or eliminate the acceptance testing phase. Instead, its goal is to enhance test coverage and facilitate a smoother acceptance testing process.

2.5 Process-Aware Information Systems

According to [8], a Process-Aware Information Systems (PAIS) is "*a software system that manages and executes operational processes involving people, applications, and/or information sources on the basis of process models*". It typically consists of 3 main components, as shown in Figure 2.5:

1. **Modeler**: where the process model is designed and a set of rules and configurations can be registered.
2. **Process Engine**: interprets the process model and controls the flows and outcomes of each interaction.
3. **Monitoring and Analysis Tool**: captures the human interactions with the Process Engine and displays the status and the data contained in the processes.

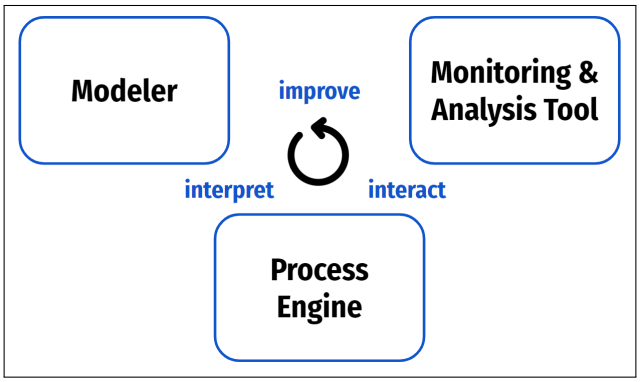


Figure 2.5: The main components of a PAIS

PAIS help organizations streamline workflows, track progress, manage exceptions, and enhance operations. They provide a framework that aligns information systems with business processes, thus improving efficiency and facilitating the effective management of complex workflows.

There are various providers of PAIS solutions, including both commercial vendors and open-source platforms [10]. Among these, Camunda[7] is notable for its comprehensive PAIS capabilities as an open-source provider. Camunda 7 offers a flexible and scalable platform for modeling, executing, and monitoring complex processes. It is renowned for its extensive feature set and robust integration capabilities, making it a popular choice for organizations seeking an open-source PAIS solution.

[7]https://camunda.com/platform-7/

2.6 AKIP Process Automation Platform

The *AgileKIP Process Automation Platform*[8] was built by developers and researchers for developers, professors and researchers willing to disseminate and build PAIS based on known technologies such as BPMN, Java and Javascript [3]. The platform's technology stack is based on two main components: the **AKIP Generator** and the **Reference Architecture**.

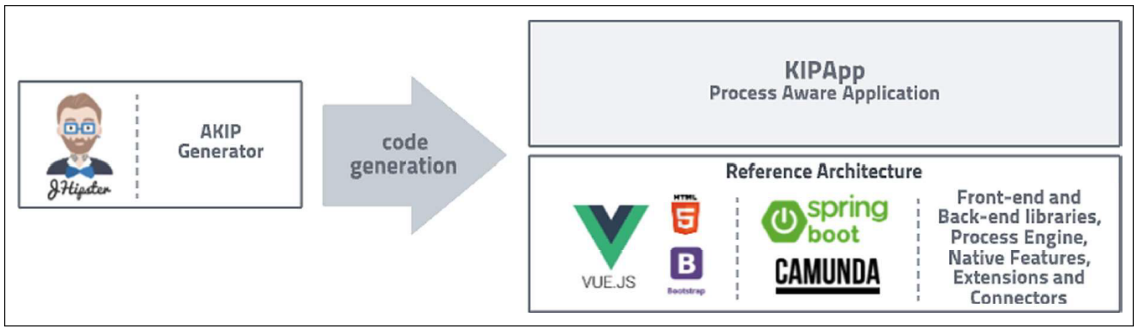


Figure 2.6: The AKIP Process Automation Platform overview [3]

The AKIP Generator is built on top of JHipster, a solid and well-known open-source application generator that makes it easy to create, configure, and deploy modern web applications and microservices. It is able to combine Spring Boot, a framework for Java development, with Angular, React or VueJS for building the frontend, providing a robust and popular technology stack.

The AKIP Generator generates code for the base Reference Architecture. The Reference Architecture seamlessly integrates features from SpringBoot and VueJS with an open-source process engine to support creating mission-critical and customizable PAISs. The Camunda 7 Process Engine[9] is the tool used in our solution to orchestrate the process workflow from an executable BPMN model.

The remaining three elements of the reference architecture are:

- **Native Features** are common features already provided by the reference architecture, meaning there is no need to repeat the code of these features in each KIPApp. Examples of native features include *Advanced Tasks List, Start-Process, Process-Instances Dashboard, Management of Deployed Processes, and User Management.*

- **Extensions** are components that allow seamless integration of the KIPApp with the process engine.

- **Connectors** are components that allow the application to integrate quickly with the external world. The main idea behind these elements is to reuse

[8]https://agilekip.github.io/pap-documentation/about
[9]https://docs.camunda.org/manual/7.14/introduction/#process-engine-infrastructure

the most common integration components through a sophisticated and highly configurable set of connectors. Examples of connectors include:

  - *Email Connector* used to send email automatically throughout the process execution;
  - *RestAPI Connector* to integrate with other systems through Rest APIs;
  - *JMS Connector* to carry out communication through messaging;
  - *AWS Connectors* that allow integration with main AWS services;

These resulting modern web applications are called **KIPApp**s [3], which stand for *Knowledge Intensive Process Applications*. A KIPApp provides the following main features:

- **Web Application Management:** This includes the management of users, configuration properties, health checks, logs, and application metrics.
- **Process Management:** It includes the management of domain entities, deployed processes, tenants, and process instances.
- **Task List:** It provides an updated user tasks list showing tasks completed, assigned, and waiting to execute.

A key aspect of a KIPApp is that the user interface for a process is generated based on descriptions provided in its entity files[10]:

1. Domain entity;
2. Process-binding entity;
3. Start-form entity;
4. User-task entity

These entities are specified in a set of JSON reference architecture files. An example of a JSON entity file, specifically the Start Form of a process, and the corresponding generated KIPApp user interface can be seen in Figure 2.7, highlighted by the red icons.

A complete walk-through of the platform and further implementation details can be found in [10] and [3].

[10]https://agilekip.github.io/pap-documentation/tutorials/getting-started#entities

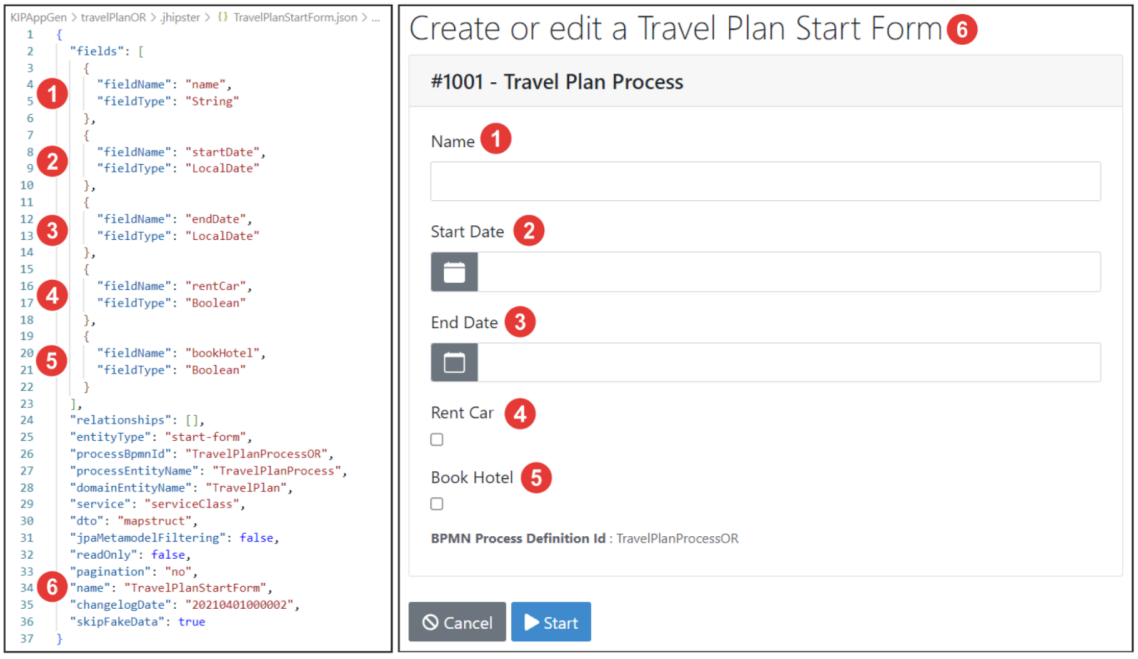


Figure 2.7: An example of a KIPApp Start-form entity file and its generated user interface.

Chapter 3

Related Work

3.1 Introduction

In the beginning of this research, an exploratory and unstructured *ad-hoc* search was made at the main scientific databases (Scopus, ScienceDirect, IEEE Xplore and ACM Digital Library) and Google Scholar, snowballing between articles that adhered to the subjects listed in the Theoretical Foundation. The objective was to find articles that were in the following intersections:

- Model-Based Testing for systems where a Process Model (e.g. XPDL, BPEL or BPMN) is a key artifact, which include PAISs;
- Model-Based Testing for the Automated Generation and/or Execution of (User-)Acceptance Tests;
- Automated (User-)Acceptance Testing strategies in general, not necessarily from Model-Based Testing, but which primarily focused on PAISs.

Given that the proposed solution involves the **Automated Testing** of **Acceptance Tests** of **BPMN**-based PAISs through a **Model-Based Testing** approach, the following 4 Core Concepts are to be considered in order to conduct a literature review:

- **CC1:** *Automated Testing* or *Test Automation*;
- **CC2:** *Acceptance Testing* or *User-Acceptance Testing*
- **CC3:** *BPMN ("Business Process Model and Notation")* or *BPM*, with its derivations *"Business Process Management"* and *"Business Process Model"*, or even only *"Business Process"* or *"Process Model"*, which are a key component of a PAIS;

- **CC4:** *Model-Based Testing* or *Test Case Generation*, specially given the common interchangeable usage of these expressions mentioned in [31];

Among these, *BPMN* **[CC3]** is the main concept, for it contextualizes the type of system and artifacts in which *MBT* **[CC4]** will be applied to achieve the *Automated Acceptance Testing* **[CC1, CC2]**. Therefore, these concepts need to always be taken in consideration **CC3**.

This becomes clear if the intersection of **CC1** and **CC2** was to be considered without **CC3**. In this case, the Automated Acceptance Testing of Games could be retrieved, for example, which is a completely different discipline and doesn't relate to a PAIS. This is also true for **CC4** without considering **CC3**, for it retrieves MBT strategies applied to artifacts and requirements that aren't necessarily relatable to the context of a PAIS.

In this initial *ad-hoc* search, a systematic literature review on process model testing [37] was identified. The key findings are:

- It considered publications from 2002 to 2013;
- Regarding the process models, the most frequently used was BPEL, with some model transformation and abstraction strategies to convert BPEL models to BPMN, a process model notation that was gaining traction at that time;
- Strategies related to testing in the process domain and the testing of processes and workflows are listed: *Unit testing*, *Test case generation*, *Test data generation*, *Integration testing*, *Quality of Service testing*, *Conformance testing*, *Regression testing* and *Formal testing*;
- There was no mention of strategies for Acceptance Testing or testing of the system's User Interface;

When [37] was published in 2015, BPMN 2.0 was still relatively new, and RPA was not yet widely adopted. This may explain the lack of publications related to BPMN and the automation of Acceptance Testing during that period. Given that it considered publications up to 2013, a revised and up-to-date systematic literature review was planned in order to identify more recent advancements and trends in BPMN-based process modeling and testing, explore the current state of RPA integration in automated testing, particularly for Acceptance Testing, and provide a comprehensive understanding of the contemporary challenges and solutions in the domain of process model testing.

When the time came to conduct a systematic literature review based on [38] procedures to address the Core Concepts of the present research and to bridge the gap left by earlier studies, a thorough literature review on techniques for BPMN

testing and formal verification had just been published at [22]. This review covers research related to the verification, validation, or testing of business process models, specifically addressing BPMN and searching for publications that apply MBT to it.

Query String **S1** is broader, aiming at everything related to verification, validation or testing of business processes, process models or BPMN. It retrieves articles which have the stem **test** (along with its variations) and **business process** or **process model** in its Title, while mentioning **BPMN** in the Abstract. It also covers articles which have the stem **test** (along with its variations) and **BPMN** in its Title or Abstract.

Query String **S2** searches specifically for documents which apply MBT to business process models. It retrieves articles which have **Model-Based Testing** (along with its variations and abbreviation) and **BPMN** in its Title or Abstract. It also retrieves articles which have the words "**test case**", which cover, for example "*Automated Test Case Generation*" or "*Automated Test Case Execution*", along with **BPMN** in its Title or Abstract.

Both queries search for documents with key terms in the title (field tag TI) and/or the abstract (field tag AB). The strings can be seen below:

S1	*(**TI** = (("business process" **OR** "process model") **AND** (verif* **OR** valid* **OR** test*)) **AND** **AB**=BPMN) **OR** **TI** = BPMN **AND** (verif* **OR** valid* **OR** test*))) **OR** (**AB** = (BPMN **NEAR** (verif* **OR** valid* **OR** test*)))*
S2	***TI** = ("business process* test*" **OR** "test* business process*" **OR** (("model-based testing" **OR** "model based testing" **OR** MBT **OR** ("test case*")) (**NEAR/5** (BPM **OR** BPMN **OR** "business process"))) **OR** **AB** = ("business process* test*" **OR** "test* business process*" **OR** (("model-based testing" **OR** "model based testing" **OR** MBT **OR** ("test case*")) **NEAR/5** (BPM **OR** BPMN **OR** "business process"))))*

In both Query Strings **S1** and **S2**, the word **BPMN** is necessarily present in any given combination, be it in the Title or in the Abstract, which aligns with the strategy laid for the 4 Core Concepts previously mentioned.

- **CC1** (*Automated Testing* or *Test Automation*) and **CC2** (*Acceptance Testing* or *User-Acceptance Testing*)**:** retrieved by the stem *test** in **S1**, combined with *BPMN*, as highlighted in yellow;
- **CC3** (*BPMN* and its variations, such as *"process model"*)**:** retrieved in both **S1** and **S2**, as shown by the words highlighted in blue, yellow, red and green;

- **CC4** (*Model-Based Testing* and its variations, such as *"test case generation"*)**:** retrieved by **S2**, as shown by the words highlighted in red and green;

Therefore, it is safe to assume that the search strings used in [22] retrieve an up-to-date superset of recent and publications required to support our work. Among all the publications identified in these two systematic literature reviews [22, 37], along with previously executed *ad-hoc* review, the following works are directly related to the present solution.

3.2 Automated Testing of a PAIS based on Process Models

A novel approach for automated regression testing of BPMN-based Process-Driven Applications (PDA) is presented in [14]. This approach allows analysts to generate executable tests for a PDA without manual coding. It includes a sophisticated model analysis, a wizard-based specification of test cases, and subsequent code generation. The resulting tests can be easily integrated into CI pipelines, but are limited to unit testing through JUnit, with no strategy for acceptance testing or verification of user interface behavior.

Another model-based approach to automatically generate test cases from business process models is demonstrated in [13]. This method models business processes and converts them to state graphs. These graphs are then traversed and transformed into the input format of the "Spec Explorer" tool, which generates the test cases. Additionally, a study evaluates the impact of process characterizations on the performance of the proposed method. However, this approach does not address user interface or acceptance testing, and the generated test cases lack associated simulated data.

In [39], an end-to-end test automation platform for business processes called ETAP-Pro is introduced, showcasing the conversion of keywords and test cases to Gherkin. However, it does not specifically reference user interface or acceptance testing.

Lastly, an approach to generate test cases from BPMN models for the automated testing of Web Applications implemented with the support of BPMSs is proposed in [16]. It aims to identify execution paths from flow analysis in the BPMN model and generate the initial code of test scripts to be run on a given Web application testing tool. This work focuses primarily on functional testing and does not mention user interface or acceptance testing.

While less directly related to our proposed approach, the work presented in [33, 34, 40, 41] is still relevant. In [33] it is showcased a Capture & Replay framework

as a foundation for a practical regression testing tool for BPMN 2.0, primarily focused on performance testing with no reference to user interface or acceptance testing. On another approach, [34] implements a generator that creates BPELUnit test suites from BPMN for testing SOAP-based business processes, again without mentioning user interface or acceptance testing.

A framework in which the test cases are automatically generated from the BPMN before any development has been conducted is proposed in [41], emphasizing the principles of test-driven approaches for software development. These tests are intended to guide and improve software development, but there is no mention of the automated execution of these test cases. Similarly, [40] designs a tool for generating test cases from BPMN, but does not reference user interface or acceptance testing.

3.3 Threats to Validity

Several factors could potentially threaten the validity of the findings and conclusions presented in this work. These threats need to be considered and addressed to ensure a comprehensive understanding of the limitations and scope of the research.

- **Limited Scientific Publications:** There is a noticeable scarcity of scientific publications that integrate BPMN, RPA, MBT and software quality/testing into a cohesive research framework. This limited body of literature means that there are fewer validated methodologies, case studies, and empirical data to draw upon. Consequently, the generalizability of the findings may be constrained, and the conclusions drawn in this work may not fully represent the broader landscape of BPMN-based PAIS and automated testing.

- **Recent Adoption of BPMN:** The widespread adoption of BPMN as the primary process modeling notation has only occurred within the last decade. As a result, there is a scarcity of use cases and empirical evidence specifically supporting the effectiveness and efficiency of automated testing in BPMN-based systems. This lack of extensive historical data and well-established methodologies poses challenges in validating and benchmarking automated testing strategies for BPMN-based systems.

- **Novelty of RPA Technology**: RPA is a relatively new technology, and its integration into software acceptance testing and quality assurance processes is still in the early stages. The novelty of RPA implies that there may be a lack of comprehensive studies and empirical data on its long-term impact and effectiveness. Additionally, as with any emerging technology, there are likely to be ongoing developments and improvements, which could render some of the current findings obsolete or less relevant in the near future.

Given these threats, it is crucial to approach the findings and conclusions with a degree of caution. Future research should aim to address these limitations by conducting longitudinal studies, expanding the empirical evidence base, and refining methodologies to adapt to the evolving nature of BPMN and RPA technologies. By acknowledging and addressing these threats to validity, the research community can work towards developing more robust and reliable frameworks for integrating BPMN, RPA, and software quality/testing.

3.4 Conclusion

All of the previously mentioned solutions presented a MBT strategy to automate the generation of test cases based on process models, primarily BPMN. While some have proposed automated execution, none has addressed acceptance testing or specifically dealt with verifying user interface behavior. Therefore, the proposed method is innovative in implementing automation that addresses acceptance testing and employs RPA to automate the user interface of a PAIS.

Chapter 4

AATPAIS: Automated Acceptance Testing of PAIS

4.1 Introduction

Given the challenges associated with enumerating all the test cases for a given PAIS [13, 14], especially considering the frequent changes in process models, a sensible strategy is to automate the generation of test cases. The ISO/IEC/IEEE 24765:2017 standard defines a test-case generator as "*software tool that accepts as input source code, test criteria, specifications, or data structure definitions; uses these inputs to generate test input data; and, sometimes, determines expected results*" [42].

The proposed solution is named **AATPAIS** (**A**utomated **A**cceptance **Te**sting of **PAIS**), and builds upon the work presented in [23]. It accepts a BPMN process model and a set of specification files for a given PAIS as input to generate RPA scripts. These scripts will interact with the user interface to interpret and input the required data.

The selected System Under Test (SUT) is a KIPApp [3], which stands for "**K**nowledge-**I**ntensive **P**rocess **App**lication", and was introduced in Section 2.6. A KIPApp is a highly customizable PAIS, and is an open-source project maintained by AgileKip. These two facts were determinants for the selection of the SUT. The selected RPA tool is the Robot Framework, a versatile open-source automation framework, which was introduced in Section 2.4.

4.2 Solution Overview

The AATPAIS is schematically represented in Figure 4.1.

The procedure, indicated by the blue icons, performed by the AATPAIS, as shown in Figure 4.1, is explained in detail below:

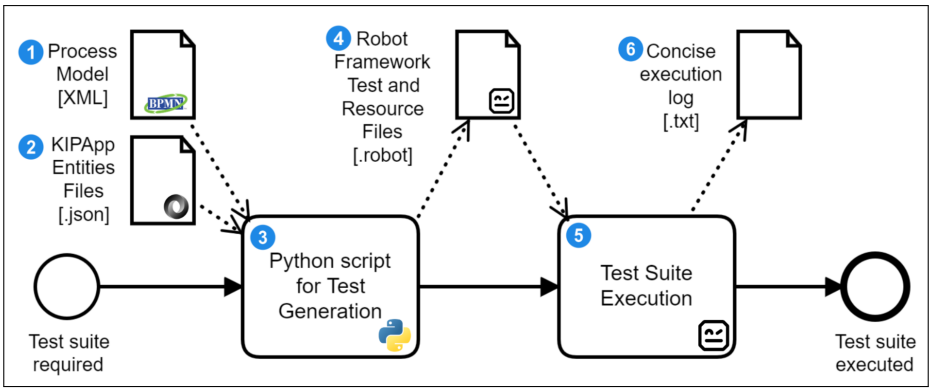


Figure 4.1: The AATPAIS solution schema

1. Extract all points of human interaction within the process's BPMN/XML. These points include:
   - The Start Event, which is always associated with a Start Form by default;
   - The User Tasks defined in the process, each linked to a corresponding user interface form.
2. Gather the specifications of fields and data for each human interaction point from the KIPApp's JSON entity files that scaffold the forms in the process. These include:
   - The JSON element corresponding to the form in the Start Event;
   - The JSON elements representing the forms for each User Task within the BPMN process.
3. Use a Python script to manipulate the aforementioned files.
4. Generate keyword-driven scripts using Robot Framework's syntax for the automated execution of an Acceptance Test Suite:
   - If the tester opts for executing tests randomly by selecting the test case `TC_BlindBatch`, the process follows the steps outlined in **Algorithm 1**, with a detailed explanation provided in Section 4.4;
   - If the tester opts for pre-planned test execution by selecting `Tags` or specified test cases, the process follows the steps outlined in **Algorithm 2**, with a detailed explanation provided in Section 4.5;
5. Execute the automated Acceptance Test Suite for the specified process defined in BPMN, using the RPA tool, i.e., Robot Framework.
6. If the execution was at random, generate a concise report of the execution.

Algorithm 1 Random Automated Testing Procedure

Require: Specify the number n of randomly conducted tests to be executed in the `FOR ${i} IN RANGE` statement of the `TC_BlindBatch` test case.

Open the web browser
Navigate to the platform's URL
Log in to the platform
for $i = 0;\ i < n;\ i = i + 1$ **do**
    Log the initiation of a test execution
    Generate simulated data
    Execute the Start Form
    while There is an available User Task **do**
        Generate simulated data
        Execute the identified User Task
        Log the execution of the User Task
    end while
    Log the completion of a test execution
end for
Generate a concise report detailing the execution of all n tests and identifying their process patterns.

Algorithm 2 Pre-Planned Automated Testing Procedure

Require: Duplicate and customize the test case `TC_Planned` and specify the `Tags` or the names of test cases to be executed.

Open the web browser
Navigate to the platform's URL
Log in to the platform
for Test Case in Names/Tags **do**
    Generate simulated data
    Execute the Start Form
    while There is a specified User Task **do**
        Generate simulated data
        Execute the specified User Task
    end while
end for

The BPMN and JSON entity files are parsed to identify the interaction points (Start Form and User Tasks), taking into account the data fields and desired behaviors within each interaction point, based on the currently supported data fields:

- String

- Integer
- Local Date
- Checkbox/Boolean
- Drop-down Menu (Many-to-One entity)

The retrieved information is then utilized to generate scripts for the keyword-driven RPA tool, Robot Framework. It generates a standard test case named `TC_BlindBatch`, which performs randomized executions of a specified number n of process instances, as determined by the tester. By default, n is set to 30. It also enables a pre-planned execution strategy, by laying out the interaction points and directing the tester. The pre-planned execution strategy will be further explained in Section 4.5.

The UI test automation is carried out using the RPA.Browser.Selenium library[1]. Each execution generates a unique set of simulated data using the Faker library[2] as follows:

- String: input a string with around 8 words of length at random;
- Integer: input a integer between 1 and 10;
- Local Date: input a date in the format `AAAA-MM-DD` at random;
- Checkbox/Boolean: select `true` or `false` at random;
- Drop-down Menu/Many-to-One: select one of the values present at random.

4.3 Generated Test Suite

This section presents the core output of the AATPAIS approach, transforming BPMN models and system specifications into executable test cases. Following the systematic procedures detailed in the Solution Overview section, the generation of these test suites aims to automate acceptance testing for PAIS effectively. By leveraging a keyword-driven approach, the test suite encapsulates user interactions, business logic, and system behavior, ensuring comprehensive test coverage and reusability.

To illustrate the practical application of the generated test suite, consider a sample BPMN model representing a `Travel Plan Process`. This process, depicted

[1]https://pypi.org/project/rpaframework/
[2]https://pypi.org/project/robotframework-faker/

in Figure 4.2, outlines the steps involved in planning a trip, including buying flight tickets, booking a hotel, and renting a car.

It is a straightforward example with key decision points, modeled to show how test cases can be generated to cover different paths within the process flow. It includes four human interaction points: `Start Form`, `Buy flight tickets`, `Book a hotel`, and `Rent a car`.

Decisions made at the `Start Form` (Figure 2.7) by selecting the checkboxes `Book Hotel` and `Rent Car` determine whether the process skips or sequentially executes the user tasks `Book a hotel` and `Rent a car`.

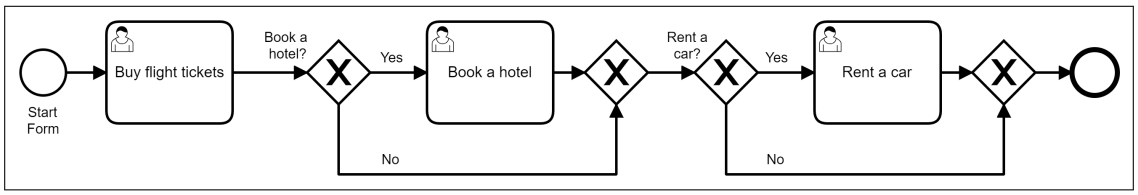


Figure 4.2: Process model of the Travel Plan Process with Exclusive Gateways (`TP-XOR` in the Assessment).

Two `.robot` files are generated as part of the implementation. The primary file, `test.robot`, serves as the main testing script where test cases and high-level keywords are defined, as illustrated in Figure 4.3. In this example, line 8 imports the process-related resources file, enabling access to lower-level keyword implementations. The complementary `resources.robot` file, shown in Figure 4.4, contains the implementations of the lower-level keywords and certain variables such as URLs and web element locators.

```
1  # Prompt Command to Execute the BlindBatch test case:
2  # robot -i TC_BlindBatch TravelPlanProcessXOR_test.robot
3  *** Settings ***
4  Library    FakerLibrary    #locale=pt_BR
5  Library    OperatingSystem
6  Library    Collections
7  Library    DateTime
8  Resource    TravelPlanProcessXOR_resources.robot
9  Test Setup    kwFakerDataSetup
10
11 *** Variables ***
12 @{TaskNames}    TaskCar  TaskFlight  TaskHotel
13
14 *** Test Cases ***
15 TC_BlindBatch
16     ${kw_executed}=    Create List
17     kwFakerDataSetup
18     kwLogin
19     ${start_time}=    Get Time    epoch
20     FOR    ${i}    IN RANGE    30
21         ${inner_list}=    Create List
22         kwFakerDataSetup
23         kwStartEvent
24         Append To List    ${inner_list}    Start of Execution #${i}
25         Append to List    ${inner_list}    StartEvent_1
26         WHILE    $processRunning == True
27             kwFindFirstAvailableTask
28             IF    $found_task == "TaskCar"
29                 kwFakerDataSetup
30                 kwTaskCar
31                 Append To List    ${inner_list}    TaskCar
32             ELSE IF    $found_task == "TaskFlight"
33                 kwFakerDataSetup
34                 kwTaskFlight
35                 Append To List    ${inner_list}    TaskFlight
36             ELSE IF    $found_task == "TaskHotel"
37                 kwFakerDataSetup
38                 kwTaskHotel
39                 Append To List    ${inner_list}    TaskHotel
40             ELSE IF    $found_task == "No task available."
41                 ${processRunning}=    Set Variable    ${False}
42                 Set Test Variable    ${processRunning}
43                 BREAK
44             END
45         END
46         Append To List    ${inner_list}    End of Execution #${i}
47         Append To List    ${kw_executed}    ${inner_list}
48 END
49 ${end_time}=    Get Time    epoch
50 ${time_spent}=    Subtract Time From Time    ${end_time}    ${start_time}    number
51 ${json_string}=    Evaluate    json.dumps(${kw_executed}, indent=4)
52 Create File    C:/Users/tales/LocalDocuments/Development/aat4pais/AssessmentProcessModels/travelPlan-XOR/
   executedKeywords-travelPlan-XOR.json    ${json_string}
53 ${data}=    Evaluate    json.loads(open("C:/Users/tales/LocalDocuments/Development/aat4pais/
   AssessmentProcessModels/travelPlan-XOR/executedKeywords-travelPlan-XOR.json").read())
54 ${execution_paths}=    Evaluate    [' => '.join(execution[1:-1]) for execution in $data]
55 ${execution_counts}=    Evaluate    dict(collections.Counter($execution_paths))    modules=collections
56 ${output}=  Evaluate  "{} executions of TravelPlanProcessXOR in {} seconds\\n".format(len($data), ${time_spent})
   + '\\n'.join(["{} executions: {}".format(count, path) for path, count in $execution_counts.items()])
57 Create File    C:/Users/tales/LocalDocuments/Development/aat4pais/AssessmentProcessModels/travelPlan-XOR/
   executionsCounter-travelPlan-XOR.txt    ${output}
```

Figure 4.3: An example of the Robot Framework test file containing the higher-level keyword implementation.

The `kwStartEvent` keyword, located in the `test` file (line 23 in Fig. 4.3), addresses the `Start Form` shown in Figure 2.7. This higher-level keyword is im-

```
1  *** Settings ***
2  Library    FakerLibrary    #locale=pt_BR
3  Library    RPA.Browser.Selenium
4  Library    String
5  Library    RPA.Desktop
6
7  *** Variables ***
8  ${url_home}    http://localhost:8080/
9  ${url_my_tasks}    ${url_home}my-candidate-tasks
10 ${locator-task-book-a-hotel-hotelBookingNumber}    task-book-a-hotel-hotelBookingNumber
11 ${locator-task-book-a-hotel-hotelName}    task-book-a-hotel-hotelName
12 ${locator-task-buy-flight-tickets-airlineCompanyName}    task-buy-flight-tickets-airlineCompanyName
13 ${locator-task-buy-flight-tickets-airlineTicketNumber}    task-buy-flight-tickets-airlineTicketNumber
14 ${locator-task-rent-a-car-carBookingNumber}    task-rent-a-car-carBookingNumber
15 ${locator-task-rent-a-car-carCompanyName}    task-rent-a-car-carCompanyName
16 ${locator-travel-plan-start-form-bookHotel}    travel-plan-start-form-bookHotel
17 ${locator-travel-plan-start-form-endDate}    travel-plan-start-form-endDate
18 ${locator-travel-plan-start-form-name}    travel-plan-start-form-name
19 ${locator-travel-plan-start-form-rentCar}    travel-plan-start-form-rentCar
20 ${locator-travel-plan-start-form-startDate}    travel-plan-start-form-startDate
21
22 *** Keywords ***
23 The user logs in
24     [Arguments]
25     [Documentation]
26     Open Available Browser    maximized=true
27     Go To    ${url_home}
28     Click Element When Visible    account-menu__BV_toggle_
29     Click Element When Visible    login
30     Wait Until Element Is Visible    login-page___BV_modal_header_
31     Input Text When Element Is Visible    username    admin
32     Input Text When Element Is Visible    password    admin
33     Click Element When Visible    //span[contains(.,'Remember me')]
34     Click Button When Visible    //button[@data-cy='submit'][contains(.,'Sign in')]
35     Sleep    500ms
```

```
[...]
44 The user is in StartEvent
45     [Arguments]
46     [Documentation]
47     Sleep    500ms
48     Go To    http://localhost:8080/process-definition/TravelPlanProcessXOR/init
49
50 The user fills StartEvent
51     [Arguments]
52     [Documentation]
53     Wait Until Page Contains    Create or edit a
54     Input Text When Element Is Visible    travel-plan-start-form-name    ${faker-name}
55     Input Text When Element Is Visible    travel-plan-start-form-startDate    ${faker-startDate}
56     Input Text When Element Is Visible    travel-plan-start-form-endDate    ${faker-endDate}
57     IF    ${faker-rentCar} is True
58         Wait Until Element Is Visible    travel-plan-start-form-rentCar
59         Select Checkbox    travel-plan-start-form-rentCar
60     ELSE IF    ${faker-rentCar} is False
61         Wait Until Element Is Visible    travel-plan-start-form-rentCar
62         Unselect Checkbox    travel-plan-start-form-rentCar
63     END
64     IF    ${faker-bookHotel} is True
65         Wait Until Element Is Visible    travel-plan-start-form-bookHotel
66         Select Checkbox    travel-plan-start-form-bookHotel
67     ELSE IF    ${faker-bookHotel} is False
68         Wait Until Element Is Visible    travel-plan-start-form-bookHotel
69         Unselect Checkbox    travel-plan-start-form-bookHotel
70     END
71
72 The user submits StartEvent
73     [Arguments]
74     [Documentation]
75     Sleep    500ms
76     Capture Page Screenshot
77     Click Button    save-entity
```

Figure 4.4: An example of the Robot Framework resources file containing the lower-level keyword implementation.

plemented through lower-level keywords detailed between lines 50 and 77 of the `resources` file (Fig. 4.4).

This form consists of five data fields, and the specific interaction implementation for each of these fields in the `resources file` (Fig. 4.4) is outlined as follows, using mock data as input in each field's `${faker}` variable:

1. `Name`: Addressed in Line 54;
2. `Start Date`: Addressed in Line 55;
3. `End Date`: Addressed in Line 56;
4. `Rent Car`: Covered in Lines 57 through 63;
5. `Book Hotel`: Covered in Lines 64 through 70.

A screenshot of the Start Form as it is being filled during the execution of an RPA test suite is shown in Figure 4.5. The mock data generated by the Faker Library is random and does not follow specific validation rules by default.

4.4 Random Execution

Considering the previously shown `Travel Plan Process` (Figure 4.2), there are four different paths given the arrangement of user interactions within the process. These four paths are highlighted with different colors in Figure 4.6.

The generated RF script performs tests at random when executing the default test case `TC_BlindBatch` (Figure 4.3, line 15), and records the results from each execution to verify test coverage afterward, supplemented by RF's comprehensive

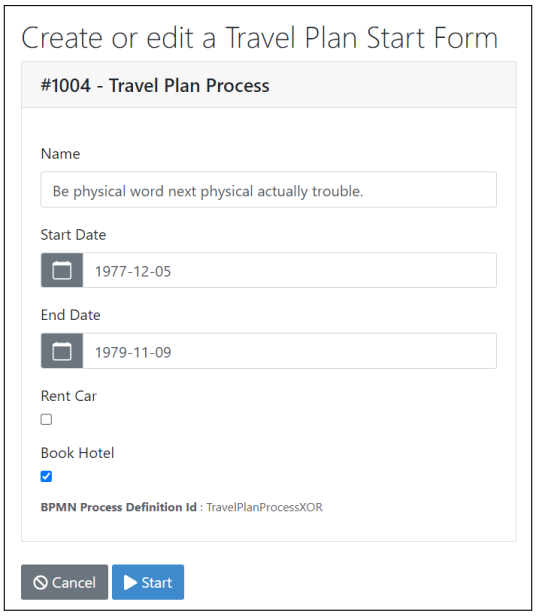


Figure 4.5: An example of the Start Form of the Travel Plan Process with the fields filled by the RPA.

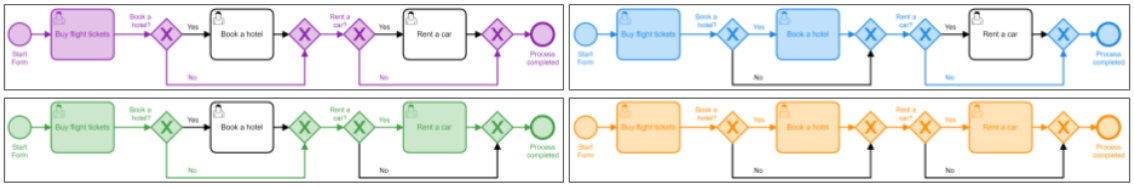


Figure 4.6: Highlighting of the four possible paths in the Travel Plan Process with Exclusive Gateways.

standard report. The simple testing paths log in a `.txt` file, with the content shown, as an example, in Listing 4.1.

To evaluate a different behavior, a slightly different version of the `Travel Plan Process` with an inclusive gateway is presented in Figure 4.7.

In this version, the same decisions made at the `Start Form` cause the process to execute none, one, or both user tasks `Book a hotel` and `Rent a car` The testing paths log for this version is shown in Listing 4.2.

It is noteworthy that in this process model, when the Start Form specifies that

Listing 4.1: Paths of 30 executions of the Travel Plan process with Exclusive Gateways

```
30 executions of TravelPlanProcessXOR in 240.0 seconds
12 executions: StartEvent=>TaskFlight
8 executions: StartEvent=>TaskFlight=>TaskHotel
5 executions: StartEvent=>TaskFlight=>TaskCar
5 executions: StartEvent=>TaskFlight=>TaskHotel=>TaskCar
```

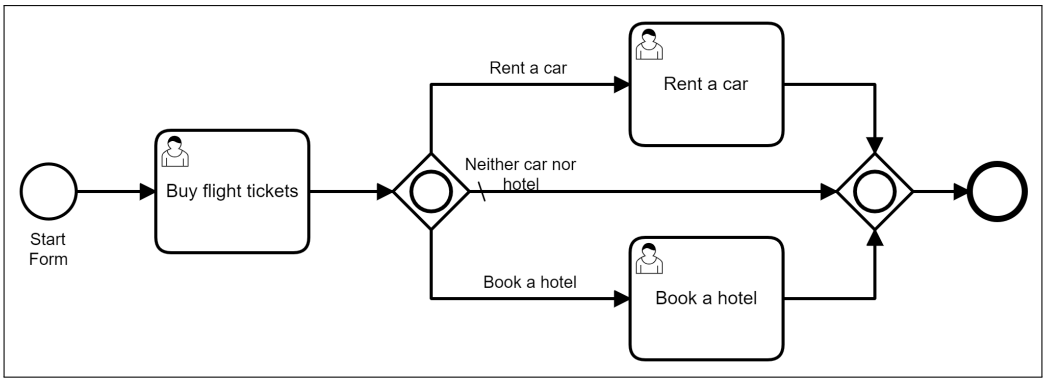


Figure 4.7: Process model of the Travel Plan Process with a Inclusive Gateway (`TP-OR` in the Assessment).

both tasks `Book a hotel` and `Rent a car` should be executed, the randomness of the executions results in four instances `Rent a car` is executed before `Book a hotel`, and two executions where the opposite occurs (see Listing 4.2). Although in these two cases the Inclusive Gateway functions as a Parallel Gateway, the tasks are executed in different orders across separate executions, which is a desired behavior.

Listing 4.2: Paths of 30 executions of the Travel Plan with a Inclusive Gateway

```
30 executions of TravelPlanProcessOR in 249.0 seconds
12 executions: StartEvent=>TaskFlight
5 executions: StartEvent=>TaskFlight=>TaskCar
7 executions: StartEvent=>TaskFlight=>TaskHotel
4 executions: StartEvent=>TaskFlight=>TaskCar=>TaskHotel
2 executions: StartEvent=>TaskFlight=>TaskHotel=>TaskCar
```

4.5 Pre-Planned Execution

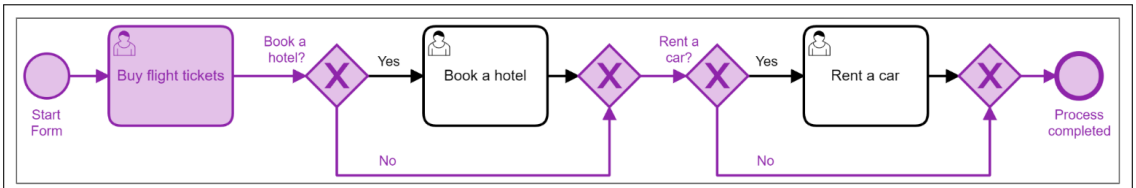
By default, the generated script has a test case named `TC_Planned`, where all the keywords related to the interaction points are listed and some guidance is given to the tester, as can be seen in Figure 4.8.

As seen in Figure 4.6, the Travel Plan Process with Exclusive Gateways has 4 possible execution paths. By manipulating the parameters of the variables in each keyword, it is possible to influence the flow of the process, fulfilling the goal set in Objective 2.3.

In `TC_01`, both Boolean variables `bookHotel` and `rentCar`, relatively related to diverting the flow towards the user tasks `Book a hotel` and `Rent a car`, are `False`, as can be seen in line 80 of the script in Figure 4.9.

```
61  TC_Planned
62      # [Tags]  planned
63      [Documentation]  Arrange the following Keywords below according to the desired test path,
        always interpolating with the kwMyTasks keyword:
64      kwLogin
65      kwFakerDataSetup
66      kwStartEvent_1
67      # kwMyTasks
68      # kwTaskCar
69      # kwMyTasks
70      # kwTaskFlight
71      # kwMyTasks
72      # kwTaskHotel
```

Figure 4.8: The `TC_Planned` test case default structure.




```
76  TC_01
77      [Tags]  planned
78      kwLogin
79      kwFakerDataSetup
80      kwStartEvent_1    argTest-bookHotel=False    argTest-rentCar=False
81      kwMyTasks
82      kwTaskFlight
```

Figure 4.9: The planned execution of `TC_01` of the Travel Plan Process.

In the screenshot captured during the execution of `TC_01` (Fig. 4.10), it can be seen that both checkboxes `Rent Car` and `Book Hotel` were left unmarked, and the rest of the fields were filled with random data.

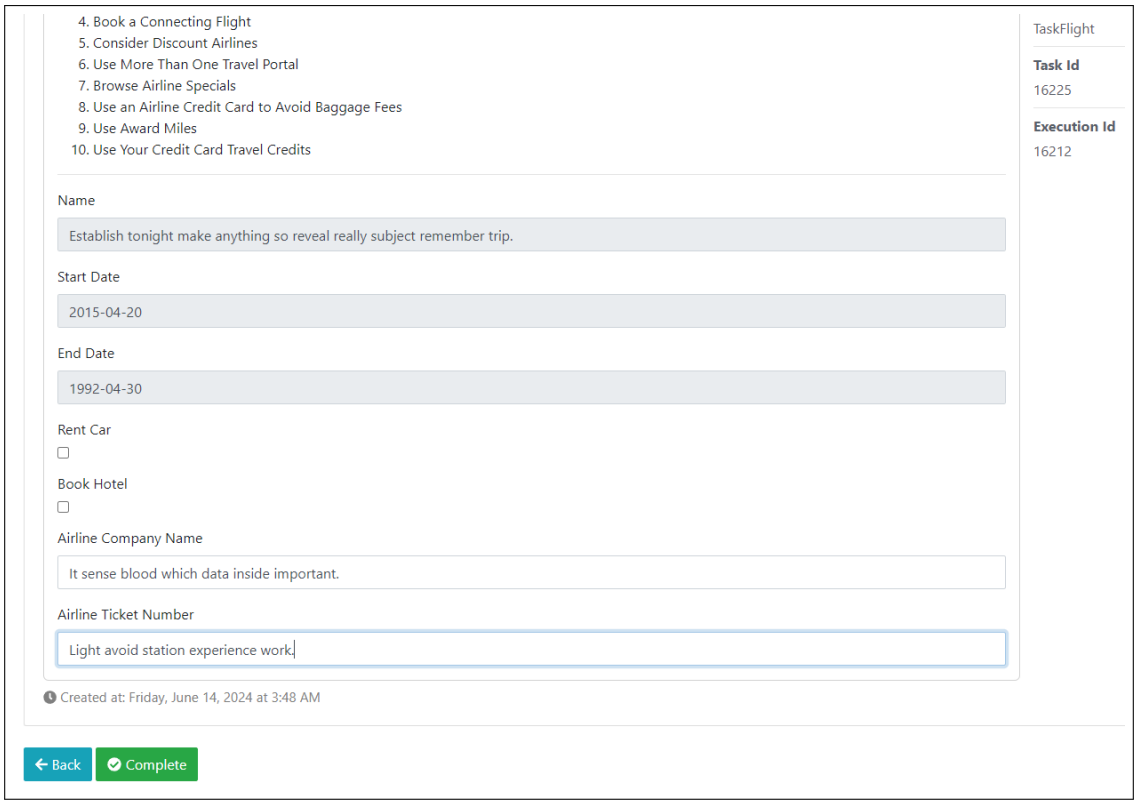


Figure 4.10: A screenshot from the execution of `TC_01` of the Travel Plan Process.

In `TC_02`, the Boolean variable `bookHotel` was assigned `True` and `rentCar` was assigned `False`, as can be seen in line 88 of the script in Figure 4.11.

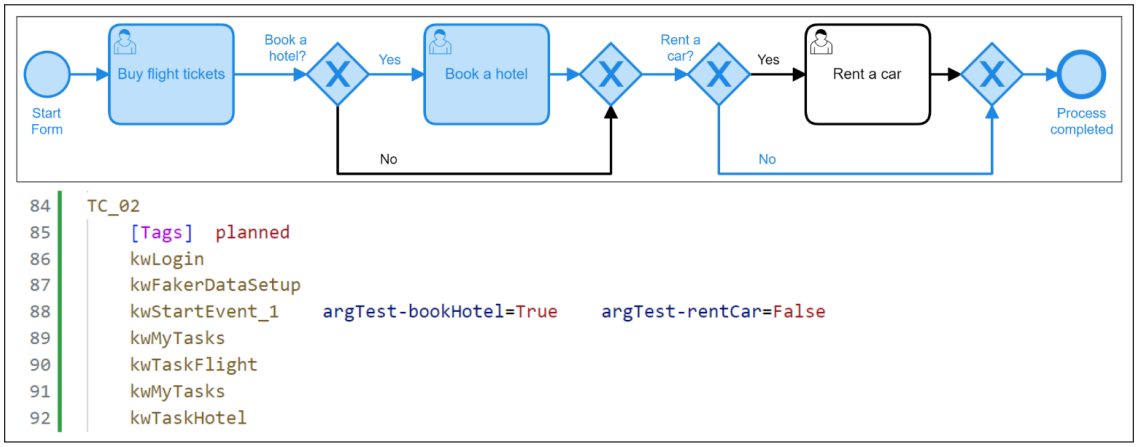
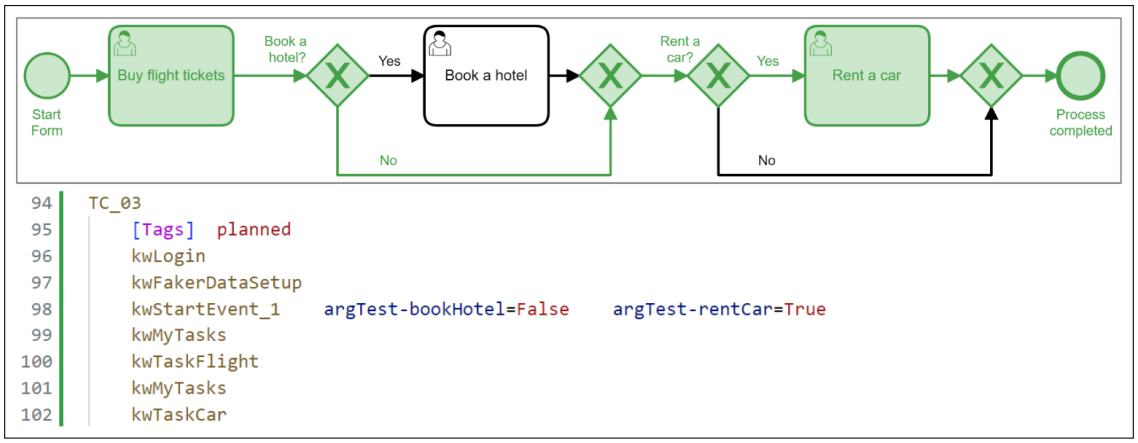


Figure 4.11: The planned execution of `TC_02` of the Travel Plan Process.

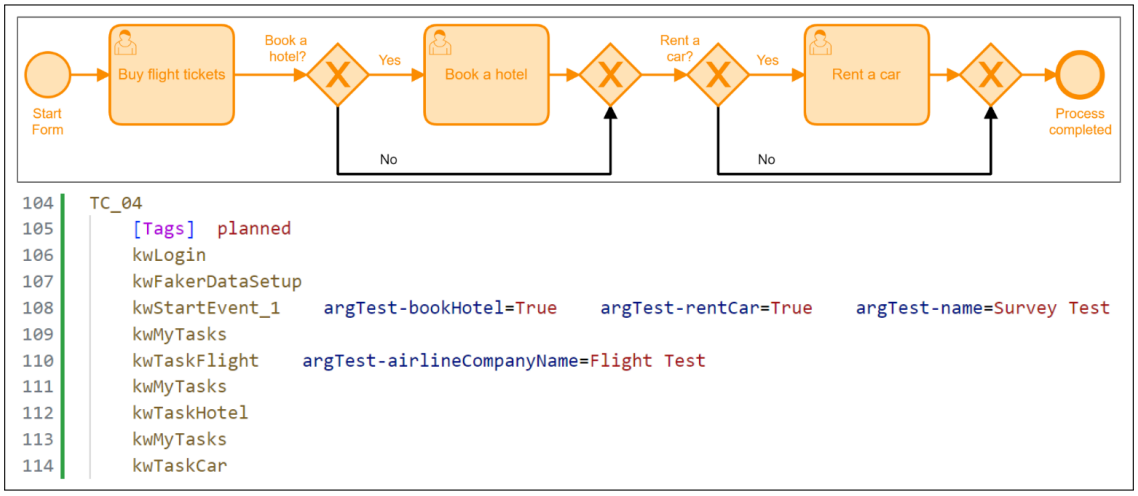
In `TC_03`, the opposite takes place, with the Boolean variable `rentCar` assigned `True` and `bookHotel` was assigned `False`, as can be seen in line 98 of the script in Figure 4.12.




```
94  TC_03
95      [Tags]  planned
96      kwLogin
97      kwFakerDataSetup
98      kwStartEvent_1    argTest-bookHotel=False    argTest-rentCar=True
99      kwMyTasks
100     kwTaskFlight
101     kwMyTasks
102     kwTaskCar
```

Figure 4.12: The planned execution of `TC_03` of the Travel Plan Process.

In `TC_04`, both Boolean variables `bookHotel` and `rentCar` are assigned `True`, as can be seen in line 108 of the script in Figure 4.13. In addition to the variables that are associated with the flow of the process, other variables can also receive arbitrary values. This can be verified as seen with the variable `name`, that receives the string "*Survey Test*" in line 108, and with the variable `airlineCompanyName`, which receives "*Flight Test*" in line 110.




```
104 TC_04
105     [Tags]  planned
106     kwLogin
107     kwFakerDataSetup
108     kwStartEvent_1    argTest-bookHotel=True    argTest-rentCar=True    argTest-name=Survey Test
109     kwMyTasks
110     kwTaskFlight    argTest-airlineCompanyName=Flight Test
111     kwMyTasks
112     kwTaskHotel
113     kwMyTasks
114     kwTaskCar
```

Figure 4.13: The planned execution of `TC_04` of the Travel Plan Process.

In the screenshot captured during the execution of `TC_04` (Fig. 4.14), it can be seen that field **Name** was filled with the string "*Survey Test*", as well as the field **Airline Company Name**, with the string "*Flight Test*". The difference is clear if compared to the previously shown screenshot captured during the execution of `TC_01` (Fig. 4.10), which used only random data.

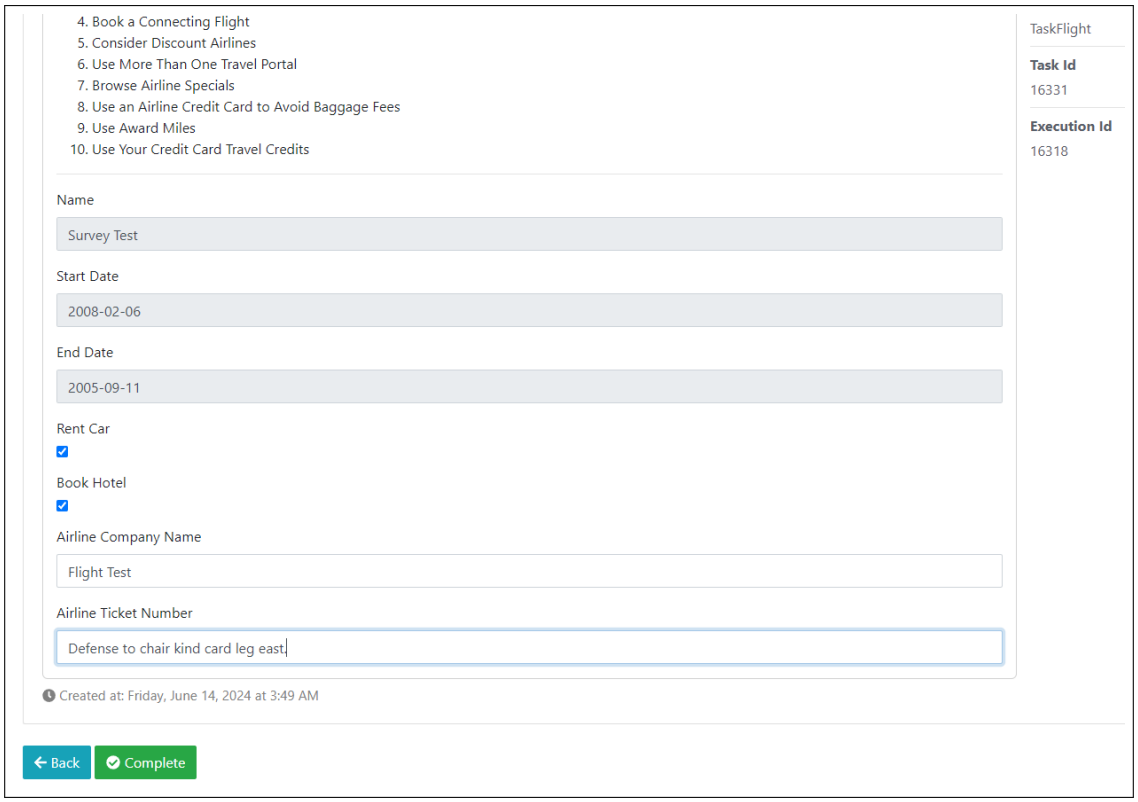


Figure 4.14: A screenshot from the execution of `TC_04` of the Travel Plan Process.

The four test cases were executed using the command `robot -i planned`, which includes all test cases with the tag `planned`, as seen in lines 77, 85, 95 and 105 of the previous figures. The execution of these four test cases took less than a minute, as seen in Figure 4.15, which shows Robot Framework's standard test execution report.

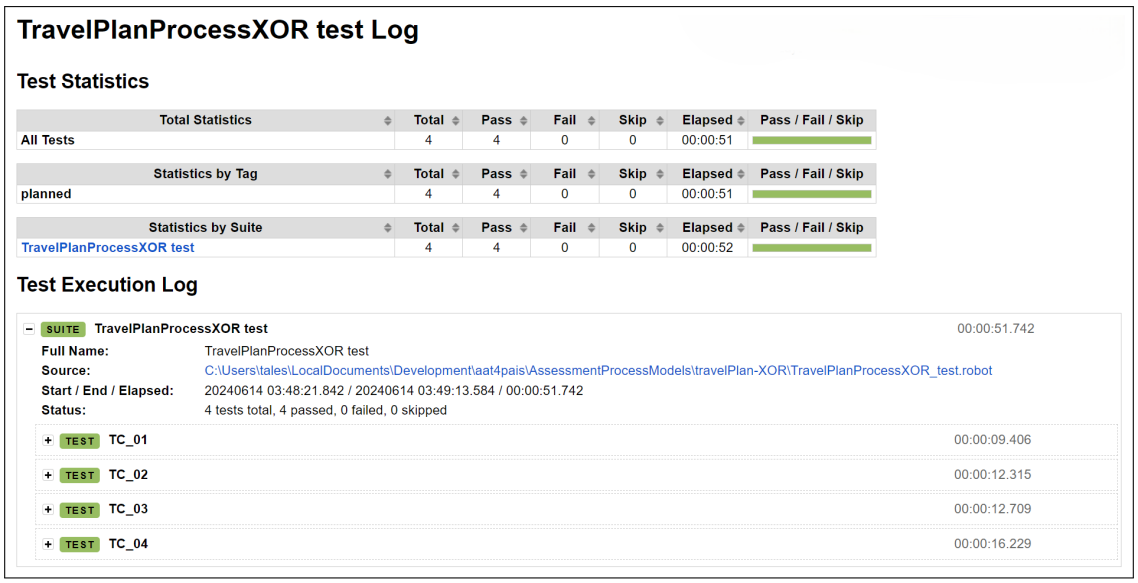


Figure 4.15: The standard Robot Framework test execution report.

4.6 Final Considerations

Our research has shown that AATPAIS can effectively generate RPA scripts that interact with user interfaces, interpreting and inputting required data for testing purposes. The integration with Robot Framework, a versatile open-source automation tool, further solidifies the robustness and adaptability of our solution. The use of KIPApps as the System Under Test (SUT) provided a practical and relevant context for evaluating the efficacy of AATPAIS.

Through the detailed procedures outlined for both random and pre-planned test executions, we have illustrated the flexibility and effectiveness of AATPAIS in handling various testing scenarios. The ability to generate keyword-driven scripts for automated execution suggests that testers can achieve high levels of test coverage without manually scripting each test case, thereby saving time and reducing the likelihood of human error.

In conclusion, AATPAIS represents a significant step forward in the automation of acceptance testing for PAISs. Its ability to generate and execute test cases automatically would make it a valuable addition to the toolkit of software testers and developers. The ongoing development and enhancement of AATPAIS will continue to contribute to the advancement of automated testing practices, ultimately leading to higher quality software and more efficient development processes.

Chapter 5

Evaluation

5.1 Introduction

The evaluation of AATPAIS is inspired by methodologies used in comparable studies that address automated testing and process automation in Business Process Management (BPM). These studies include works such as [3, 13, 14, 16, 39], which informed various aspects of our assessment strategy, from test coverage analysis to the application of acceptance testing in complex, dynamic environments.

In the context of test coverage evaluation, different kinds of coverage metrics are defined in [13]: Statement Coverage, Branch Coverage, Path Coverage, and Requirements Coverage. Statement and Branch Coverage focus on verifying whether individual statements and control structures within a program are executed, while Requirements Coverage ensures that all specified requirements are adequately tested. Our evaluation centers on Path Coverage as a primary metric, given that it measures the extent to which all possible execution paths within a business process model are covered by the automated tests.

However, the focus here is not on merely covering all paths in the process flow but rather on ensuring that paths requiring human interaction are thoroughly tested. In the context of BPMN-based PAIS, this entails validating the User Interface (UI) during an Acceptance Testing routine. Therefore, the goal of our evaluation is to examine the breadth of path coverage achieved by AATPAIS, specifically those that involve User Tasks, Start Forms, and other elements requiring direct human input.

By assessing the path coverage of human interaction points, we can gauge how well AATPAIS supports end-to-end UI testing, ensures the correctness of process flows, and maintains the integrity of user interactions across various scenarios. This evaluation helps identify the strengths of AATPAIS in automating acceptance testing and highlights any areas where coverage might be enhanced to meet real-world demands for software quality assurance in BPMN-based applications.

5.2 Simulation

The simulation phase of this evaluation aims to put AATPAIS to the test in real-world scenarios by applying it to a diverse set of process models. The goal of the simulation is to measure how effectively AATPAIS can automate acceptance testing across varying BPMN process models, emphasizing path coverage of human interaction points. By doing so, we can demonstrate the tool's potential in reducing the time, effort, and resources needed for thorough acceptance testing in a PAIS environment.

The simulation process involves selecting representative process models that vary in size, complexity, and structure, ensuring that the evaluation encompasses a broad spectrum of scenarios. These models are derived both from the previously designed process models and from existing use cases in the AgileKIP open-source project repository. By applying AATPAIS to these models, we aim to achieve a comprehensive understanding of its ability to generate test cases, execute them automatically, and assess path coverage in a realistic context.

By conducting this simulation, we seek to address critical questions about the efficiency, accuracy, and completeness of the test cases generated by AATPAIS. Specifically, the evaluation will explore whether AATPAIS can deliver considerable Path Coverage in a range of process models without additional tester customization, as well as how effectively it handles scenarios with complex decision points, multiple execution paths, and diverse human interaction requirements. The results and insights gained from this simulation will not only demonstrate the strengths of AATPAIS but also provide guidance for future improvements and enhancements to its automated acceptance testing approach.

5.2.1 Goals

The primary goal of this evaluation is to determine whether the test cases generated by AATPAIS can ensure comprehensive test coverage across different process models, as outlined in Objectives 1.1 and 1.2. By formulating a systematic approach for generating test cases from BPMN models and system requirements specifications, and by managing the complexity and variability of process models, we aim to enhance the comprehensiveness of test coverage.

Additionally, the second goal is to verify if the automatically executed test cases can cover most or all of the paths within the process models. This assessment will evaluate the effectiveness of automating repetitive and time-consuming tasks that are prone to errors, contributing to easing the workload of the tester. This goal aligns with Objectives 2.1 and 2.2, which focus on implementing a framework that uses a screen automation tool, such as RPA, to automate the execution of a

comprehensive Acceptance Testing strategy and enhancing the testing process by automating error-prone tasks. Objective 2.3 is evaluated mainly in Section 5.3, as the usefulness and ease-of-use of the solution are assessed through a survey, taking in consideration both Random and Pre-Planned executions.

5.2.2 Metrics and Procedure

The two guiding principles in the assessment of AATPAIS will be borrowed from [3, 13]. The proposed assessment methodology involves the following steps:

1. Enumerate N business processes for evaluation of the impact of process size and process complexity on the path coverage of a Random Execution;

2. Measure the *size* and *complexity* of the N business processes:

   - **Number of Human Activities (NOHA)**: This metric is similar to NOA [3], but focusing on human interaction, and it encompasses the total number of User Tasks and the form associated with the Start Event, disregarding Tasks that don't require human interaction, such as Service Tasks, and are not, by design, in the scope of the present solution;
   - **Number of Nodes (NON)**: This metric takes into account Gateways, Events and the other types of Tasks, in addition to the elements considered in NOHA, totalling the same as described by [3];
   - **Coefficient of Network Connectivity (CNC)**: This metric is obtained by dividing the number of flows in a process model by the NON.
   - **Control Flow Complexity (CFC)**: This metric is often utilized to evaluate the complexity introduced by various types of split gateways. CFC is an additive metric, where each split gateway in the model contributes a certain complexity value based on the subsequent states. Specifically:
     - Exclusive (XOR) or Event-based Split: a complexity value of n is assigned, corresponding to the number of outgoing flows from the gateway;
     - Inclusive (OR) Split: a complexity value of $2^n - 1$ is assigned;
     - Parallel (AND) Split: a complexity value of 1 is assigned.

3. Evaluate the *Path* and *Human Interaction* coverages of the generated test suite through n instances of a Random Execution of the N business processes. By default, $n = 30$.

Given that by default, in a KIPApp, the Start Event is linked to a form, and requires human interaction, the Start Event will be treated as a Task for NOHA calculation.

5.2.3 Subjects

In the assessment, a total of N=21 business processes models from 2 sources were selected for evaluation, as shown in Appendix A.

In the beginning of the current research, a Customer Feedback (**CF**) process was designed, in order to simulate a process in which a customer can convey a compliment, a suggestion or a complaint. Depending on the gravity of the complaint, the process might be escalated to a supervisor. It is a simple process with a simple domain, modeled by the authors, in order to try different paths and combinations while AATPAIS was being designed and developed. In total, 9 variations of the CF process were selected[1], and one of its variations depicted in Fig. 5.1.

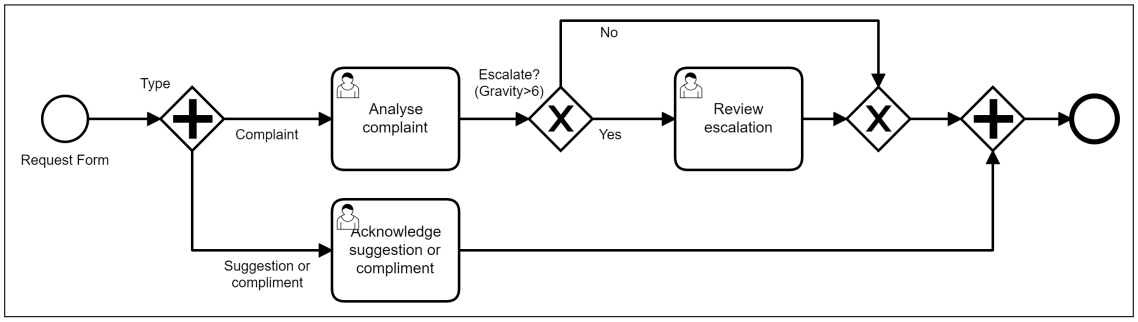


Figure 5.1: Process model of the Customer Feedback Process with Parallel and Exclusive Gateways (`CF-parallel-all-types-with-scalation` in the Assessment).

From AgileKIP's publicly available Tutorial[2], 12 variations of the Travel Plan (**TP**) process were selected from the repository[3]. The TP process encompasses the steps users execute when they are planning a trip, and involve choosing a destination and dates, booking a flight, booking a hotel, renting a car, etc. One of its variations depicted in Fig. 5.2.

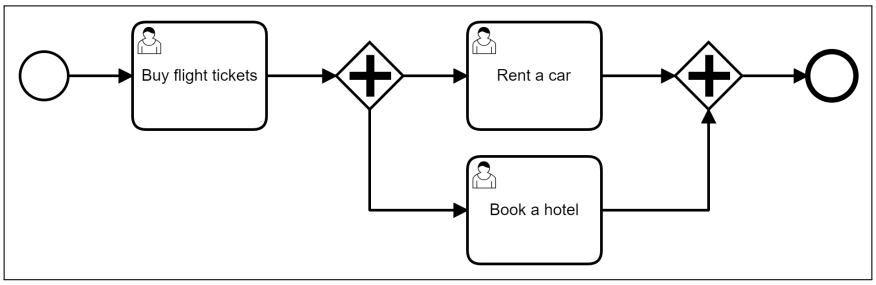


Figure 5.2: Process model of the Travel Plan Process with Parallel Gateways (`TP-AND` in the Assessment).

[1]https://github.com/talesmp/AATPAIS-SpringerNatureLNCS
[2]https://agilekip.github.io/pap-documentation/tutorials
[3]https://github.com/AgileKip/akip-travel-example

5.2.4 Test Suite Generation Results

The performance of the tool during the execution of the script for test suite creation was measured[4] using various process models. The measurements were performed threefold, and the arithmetic means were calculated based on the measured values. The test suite generation of each process model was completed well within less than 10ms, as seen in Table 5.1. Thus, the time needed for the test suite generation is more than acceptable for practical use.

Table 5.1: Test Suite Generation Average Time

Process Model	**Time (ms)**
CF-exclusive-all-types-no-scalation	3.57
CF-exclusive-all-types-with-scalation	2.45
CF-inclusive-all-types-no-scalation	2.33
CF-inclusive-all-types-with-scalation	5.94
CF-parallel-all-types-no-scalation	3.05
CF-parallel-all-types-with-scalation	2.98
CF-only-complaint-no-scalation	2.45
CF-only-complaint-with-scalation	3.40
CF-only-suggestion-compliment	2.00
TP-AND	2.99
TP-EMSG	3.21
TP-ENTITIES	3.46
TP-ENTITIES2	3.46
TP-ENTITIES3	3.14
TP-LOOP	2.64
TP-OR	2.89
TP-SIMPLE	3.06
TP-SRV	2.66
TP-TIMER	3.02
TP-VAL	2.32
TP-XOR	2.41

The *size* and *complexity* of the 21 business processes were evaluated and presented in Table 5.2. The *Path* and *Human Interaction* coverages of the automated test suite generation and execution of the 21 business processes are also shown in Table 5.2, and will be further discussed in the following sections.

[4]Measured with the following system: CPU: AMD Ryzen 9 (4.0 GHz); RAM: 16GB DDR5 (4800 MHz); SSD: 512GB

Table 5.2: Size and Complexity Measures and Path Coverage of Process Models in 30 Automated Random Executions

	Size Measures		Complexity Measures			Automattion Metrics			
Process Model (TP and CF)	NOHA[1]	NON[1]	Paths	CNC[1]	CFC[1]	EN0HA[2]	NOHAC[2]	EP[2]	PC[2]
CF-exclusive-all-types-no-scalation	3	6	2	0.33	2	3	100%	2	100%
CF-exclusive-all-types-with-scalation	4	9	3	0.33	4	4	100%	3	100%
CF-inclusive-all-types-no-scalation	3	6	3	0.50	3	3	100%	2	67%
CF-inclusive-all-types-with-scalation	4	9	5	0.56	5	4	100%	3	60%
CF-parallel-all-types-no-scalation	3	6	1	0.17	1	3	100%	1	100%
CF-parallel-all-types-with-scalation	4	9	2	0.22	3	4	100%	2	100%
CF-only-complaint-no-scalation	2	3	1	0.22	3	4	100%	1	100%
CF-only-complaint-with-scalation	3	6	2	0.33	2	3	100%	2	100%
CF-only-suggestion-compliment	2	3	1	0.33	0	2	100%	1	100%
TP-AND	4	7	1	0.14	1	4	100%	1	100%
TP-EMSG	4	10	3	0.30	2	3	75%	1	33%
TP-ENTITIES	4	5	1	0.20	0	4	100%	1	100%
TP-ENTITIES2	4	6	1	0.17	0	4	100%	1	100%
TP-ENTITIES3	4	5	1	0.20	0	4	100%	1	100%
TP-LOOP	4	6	2	0.33	2	4	100%	1	50%
TP-OR	4	7	4	0.57	7	4	100%	4	100%
TP-SIMPLE	4	5	1	0.20	0	4	100%	1	100%
TP-SRV	4	8	2	0.25	2	3	75%	1	50%
TP-TIMER	4	10	3	0.30	2	3	75%	2	67%
TP-VAL	4	5	1	0.20	0	4	100%	1	100%
TP-XOR	4	9	4	0.44	4	4	100%	4	100%

[a]NOHA: Number of Human Activities; NON: Number of Nodes; CFC: Coefficient of Network Connectivity; CFC: Control Flow Complexity.
[b]ENOHA: Executed NOHA; NOHAC: NOHA Coverage; EP: Executed Paths; PC: Path Coverage.

5.2.5 Path Coverage Results

It was observed a path coverage (PC) of 100% in 15 out of the 21 selected process models with the direct execution of the automation scripts, as shown in Table 5.2, requiring no further time spent by the tester/developer with customization. Although the proposed solution still requires time from the tester to interpret result logs according to the process model, it frees the tester from doing tedious and repetitive activities such as manually clicking and typing.

Furthermore, it means that these 15 process models can be constantly tested following a regression testing strategy. This facilitates the identification of issues and bugs stemming from changes in the code or interface that may not be directly related to changes in the process model itself.

Regarding the process models that did not achieve total path coverage (PC):

1. **TP-EMSG:**
   (a) Inability to trigger the Message Boundary Event associated with the `Book a hotel` user task, therefore never activating this specific path;
   (b) The exclusive gateway condition is never satisfied, because it requires a specific field to have exactly the substring "`byCar`" in it, and the default random generation of text by the Faker Library never generated such substring, hindering the `Rent a car` user task, and the flow associated to it, unused;
2. **TP-LOOP:** The exclusive gateway condition is never satisfied, since it requires that a specific field is null, and the automation script fills every field in each form that is not marked as `readOnly`, therefore never activating this specific path;
3. **TP-SRV:** Same reason as item **1(b)**;
4. **TP-TIMER:**
   (a) Inability to wait the time necessary to trigger the Timer Boundary Event associated with the `Book a hotel` user task, therefore never activating this specific path;
   (b) Same reason as item **1(b)**;
5. **CF-inclusive-all-types-no-scalation:** The conditions present in the inclusive gateway are incompatible with how the field `Type` works, being a single selection item in a drop-down menu, hence not being able to satisfy more than one condition at a time in the inclusive gateway (behaving like a exclusive gateway) and not activating more than one flow at a time;

6. **CF-inclusive-all-types-with-scalation:** Same reason as item **5**.

5.2.6 Human Interaction Coverage Results

Regarding the human interaction points coverage (NOHAC), i.e. Start Form, User Tasks and navigating through the platform, the execution was able to achieve 100% of NOHAC in 18 out of the 21 processes, as shown in Table 5.2.

All 3 cases in which complete coverage was not possible (`TP-EMSG`, `TP-SRV` and `TP-TIMER`), it was related to an exclusive gateway condition never being satisfied. This gateway condition requires a specific field to have exactly the substring "`byCar`" in it, and the default random generation of text by the Faker library never generated such substring, hindering the `Rent a car` user task and the flow associated to it, unused. In these cases, a simple customization of the RF automation scripts, as shown in Section 4.5, in order to foresee and conform to the conditions in the gateways, would be enough to achieve complete coverage of the interfaces.

5.3 Survey

To analyze the suitability of the proposed extension, we adopted an evaluation strategy through a survey, comprised of three stages: Planning, Preparation, and Execution. The survey aimed to gather comprehensive feedback from IT professionals and potential users regarding their experience with AATPAIS, focusing on their perceptions of its usefulness and ease-of-use, aligning with Objective 3.2 described in Section 1.2. These stages were designed to provide relevant information for analyzing the evaluation results. The following sections will detail each stage of the research evaluation protocol, ensuring a thorough understanding of the methodology employed.

5.3.1 Planning

The planning of the survey encompasses defining the evaluation objectives, outlining the scope for participant selection, and other essential elements. This foundational step ensures the necessary basis for preparing and conducting the evaluation, as well as for the subsequent extraction and analysis of the results.

Objective

The objective of the present survey was to evaluate the effectiveness and the user experience of the AATPAIS solution using a questionnaire designed to measure its

perceived usefulness and ease-of-use. This evaluation was inspired by the Technology Acceptance Model (TAM) proposed by [43] and the evaluation strategy used in [14].

Desired Participant Profile

The target audience for this research will consist preferably of professionals and researchers with experience in the selected tools and frameworks, namely AgileKIP, Camunda and Robot Framework. The choice of this target audience is based on the need to gain insights from professionals with practical knowledge in IT environments, as they possess a relevant perspective for analyzing the effectiveness of the proposed solution.

Invitations to participate in the study were sent by message to the users of the AgileKIP Process Automation Platform, and via Camunda and Robot Framework communities' Slack workspace. The AgileKIP user group consists of approximately 20 known users. Invitations to participate in the survey were sent directly to each member via message. The Camunda Community Slack workspace has around 200 users. The administrators recommended posting the invitation in the open `fun-random` channel.

The Robot Framework Slack workspace boasts approximately 30,000 users. The administrators granted permission for the invitation to be posted in two channels: `random`, an open and highly active channel accessible to all 30k users, and `robocon2024-online`, a closed channel with 34 members, including the organizers and presenters of RoboCon2024, one of the conferences where AATPAIS was showcased, who were already familiar with the tool.

Survey Tools

For the preparation, execution, and data analysis of this evaluation, several tools were selected. For the survey preparation and data collection, *Google Forms* was selected due to its ease of creating online questionnaires and collecting participant responses. Regarding the tabulation and analysis of the collected data, *Google Sheets* was chosen for its integration with *Google Forms* and advanced features that facilitate efficient data organization and analysis. These choices were fundamental for effectively conducting the evaluation and subsequently analyzing the results, as the combination of these tools provided an integrated workflow throughout the entire research process.

5.3.2 Preparation

The preparation phase encompasses all the elements and procedures necessary to support the execution, data extraction, and analysis phases. This stage must be

aligned with the characteristics of the participants involved and the predefined evaluation objectives.

Presentation

The participants were provided with a brief summary in the introduction of the questionnaire, outlining the purpose of the research and defining the evaluation objectives, as well as requesting their Informed Consent, with question `ID 1` being their *Consent Record*.

This is followed by questions about the respondent's Demographic Information. A tutorial video uploaded to YouTube[5] was provided to give a more detailed explanation about AATPAIS. In the tutorial video, the following topics were covered:

- Theoretical Fundamentals:
  - Acceptance Testing in SDLC;
  - Model-Based Testing;
  - Test-Case Generators;
- Our proposal: AATPAIS;
- Introducing a KIPApp;
- Introducing Robot Framework's syntax;
- AATPAIS in Action:
  - Automated Generation of Test Cases;
  - Random Execution of Test Cases;
  - Customization of Scripts and Planned Execution of Test Cases;

Finally, the questions related to the Perceived Usefulness and Ease-of-Use are presented. The complete questionnaire is presented in Appendix B.

Demographic Information

In the first section, participants were invited to complete the *Demographic Information Questionnaire* (Table 5.3).

[5]https://youtu.be/uWQ8DSWheXc

Table 5.3: Questionnaire's Demographic Questions

ID	Question	Format	Required
2	What is your preferred email address?	Open	Yes
3	What is your age?	Closed	Yes
4	What is your gender?	Closed	Yes
5	What is your highest academic degree?	Closed	Yes
6	Which sector do you primarily work in?	Closed	Yes
7	What is your primary role in your organization?	Closed	Yes
8	Do you work with IT? How many years of experience with IT do you have?	Closed	Yes
9	Have you used a Process-Aware Information System (PAIS) before?	Closed	Yes
10	Have you used a Robotic Process Automation (RPA) tool before?	Closed	Yes

This questionnaire collects demographic data mostly related to academic and professional experience, allowing us to profile the participants and identify potential differences in perceptions based on their professional characteristics. It also helps identifying participants that are minimally acquainted with Process-Aware Information Systems and Robotic Process Automation tools.

Perceived Usefulness and Ease-of-Use Questions

As previously mentioned, evaluation about the Perceived Usefulness and Ease-of-Use of AATPAIS was inspired by the Technology Acceptance Model (TAM) proposed by [43], as well as the evaluation strategy used in [14], which is the closest related work identified.

A pilot survey was applied to three people, and, based on their feedback, the final version of the Perceived Usefulness and Ease-of-Use Questions was designed. It is composed of 13 questions using the Likert scale, as seen in Table 5.4. Questions `ID 11 to 17` are related to the Perceived Usefulness of the solution, and questions `ID 18 to 23` are related to the Perceived Ease-of-Use.

Table 5.4: Questionnaire's Perceived Usefulness and Ease-of-Use Questions

ID	Question	Format	Required
11	I found AATPAIS helpful for the creation and specification of executable test cases.	Likert	Yes
12	Using AATPAIS would enable testing to be accomplished more quickly.	Likert	Yes
13	Using AATPAIS would increase the productivity.	Likert	Yes
14	AATPAIS increases my motivation and willingness to create test cases for process models.	Likert	Yes
15	Using AATPAIS would improve testing performance.	Likert	Yes
16	Using AATPAIS would make it easier to do my job.	Likert	Yes
17	I would find AATPAIS useful in my job.	Likert	Yes
18	I would find it easy to get AATPAIS to do what I want it to do.	Likert	Yes
19	Learning to operate AATPAIS would be easy for me.	Likert	Yes
20	The test cases seemed to be clearly organized to me.	Likert	Yes
21	My interaction with AATPAIS would be clear and understandable.	Likert	Yes
22	I would find AATPAIS easy to use.	Likert	Yes
23	It would be easy for me to become skillful at using AATPAIS.	Likert	Yes
24	You may already have ideas and suggestions for improving AATPAIS. Please indicate them below.	Open	Yes

5.3.3 Execution

This section provides an analysis of the Demographic Information collected from the survey respondents. The survey questionnaire was made available in the main communication channels of AgileKIP, Robot Framework Foundation, and Camunda user groups. These user groups were chosen to reach a broad audience of professionals familiar with BPMN, the chosen RPA tool, and chosen PAIS technologies, ensuring a relevant respondent pool.

The demographic data help to understand the background and diversity of the participants, which is crucial for interpreting the overall feedback and insights on AATPAIS. The survey included a total of 21 respondents, and the demographic section consisted of eight questions. It is important to note that Question 1 requested the respondents' email addresses for follow-up purposes, but this information will remain confidential and will not be disclosed in this analysis.

The demographic questions covered a range of aspects including age, gender, highest academic degree, primary work sector, primary role in the organization, IT experience, and prior use of Process-Aware Information Systems (PAIS) and Robotic Process Automation (RPA) tools. By examining these demographic factors, we can better understand the context in which the respondents are evaluating AATPAIS and ensure that the feedback is representative of a diverse group of users. The following paragraphs provide a detailed analysis of each demographic question, highlighting the implications and insights derived from the collected data.

The age distribution of the respondents indicates a diverse range of participants, as depicted in Figure 5.3. The largest group, 7 out of 21 respondents (52.4%), are aged 18-24 years. This large representation from younger professionals suggests that the survey reached individuals who are likely early in their careers and possibly more open to new technologies and methods.

Additionally, 6 respondents (28.6%) are aged 25-34 years, 6 respondents (14.3%) are aged 35-44 years, and 2 respondents (4.8%) are aged 45-54 years. No participant over 55 years old responded to the survey. As the form of consent demanded, there's no participant under 18 years old.

The presence of older, more experienced professionals (33.4% aged 35 and above) adds valuable insights from those who might have a broader perspective on the evolution and adoption of technologies like BPMN and RPA over time.

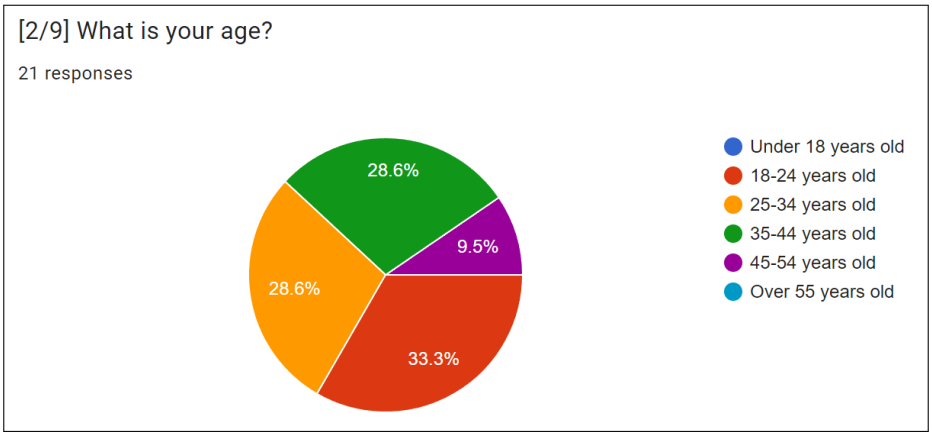


Figure 5.3: Survey Demographics - Question 2: Age Distribution

Regarding gender, the survey results show that 16 out of 21 respondents (76.2% are male, while 5 respondents (23.8%) are female, as seen in Figure 5.4. This gender distribution reflects a higher participation of males in the survey, which is consistent with industry trends where technology-related fields often have more male professionals. However, the presence of a good number of female respondents is encouraging, as it indicates growing diversity and the inclusion of different perspectives within the field.

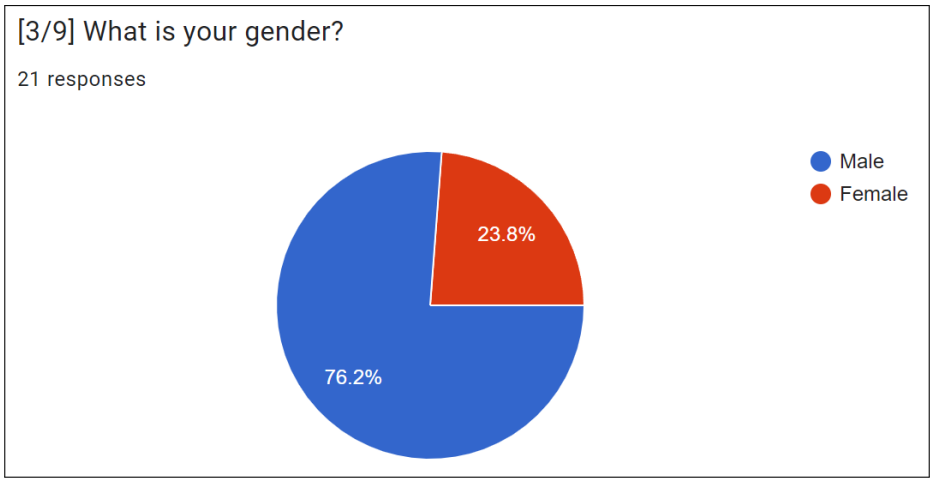


Figure 5.4: Survey Demographics - Question 3: Gender Distribution

The highest academic degrees of the respondents reveal a mix of educational backgrounds, as seen in Figure 5.5. The majority, 12 out of 21 respondents (57.1%), are currently pursuing a Bachelor's degree, indicating a strong presence of students or early-career professionals. This aligns with the results from Question 2, indicating a large group of young individuals (18-24 years old).

Meanwhile, 2 respondents (4.8%) have completed a Bachelor's degree, and 7 respondents (33.3%) have completed a Master's degree. This suggests that the survey reached individuals who are both currently in academia and those who have already transitioned into professional roles. The high level of academic engagement among respondents could imply a better understanding of theoretical concepts and a higher likelihood of appreciating the technical aspects of AATPAIS.

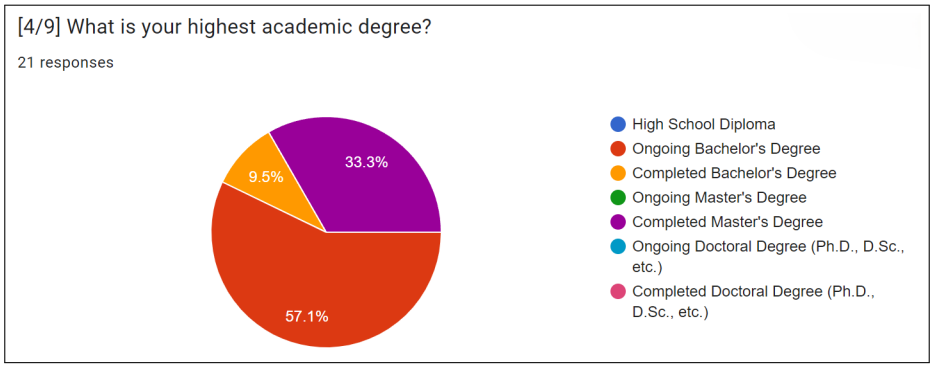


Figure 5.5: Survey Demographics - Question 4: Highest Academic Degree

The primary work sectors of the respondents are predominantly in the industry, with 17 out of 21 respondents (81.0%) working in the industry sector. This strong industry representation suggests that the feedback is grounded in practical, real-world experience, making it highly relevant for evaluating the applicability and effectiveness of AATPAIS in professional settings. Additionally, 3 respondents (14.3%) work

in both industry and academia, and 1 respondent (4.8%) works solely in academia. This mix (Fig. 5.6) ensures that the survey captures a range of perspectives, including those involved in scientific research and those applying such technologies in practice.

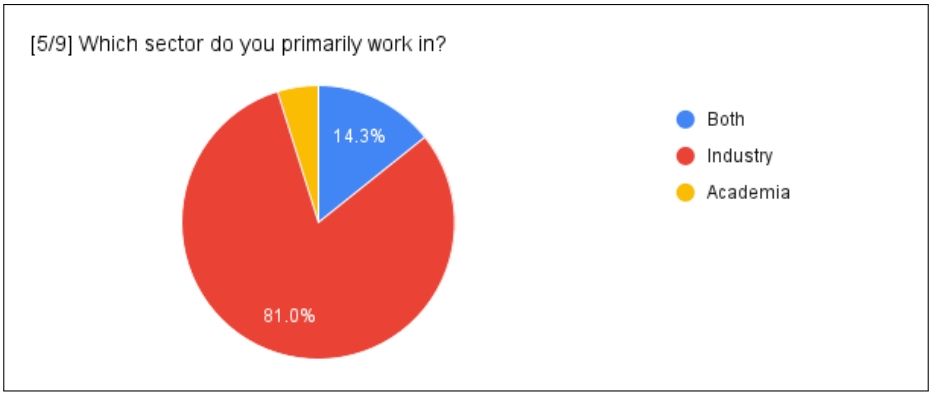


Figure 5.6: Survey Demographics - Question 5: Working Sector

The primary roles of the respondents within their organizations show a concentration in technical roles, particularly software development and IT consultancy, as seen in Figure 5.7. Specifically, 10 respondents (47.6%) identified as Software Developers, 5 respondents (23.8%) as IT Consultants, 3 respondents (14.3%) as QA Engineers/Test Engineers, and 2 respondents (9.5%) as Software Analysts. Additionally, 1 respondent (4.8%) is a Software Architect. The concentration in these roles suggests that the participants are well-positioned to provide informed feedback on the practical aspects of implementing and using AATPAIS, given their hands-on experience with software development and testing processes.

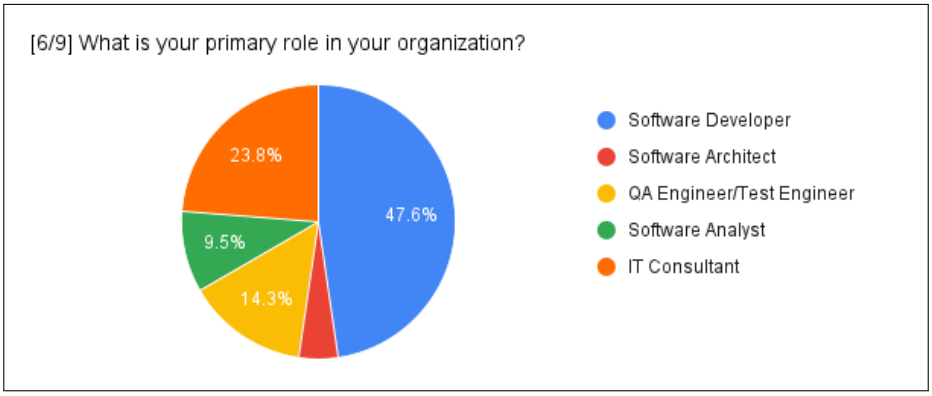


Figure 5.7: Survey Demographics - Question 6: Primary Role

In terms of IT experience, a significant proportion of respondents have substantial experience in the field (Fig. 5.8). Eight respondents (38.1%) have 1 to 3 years of experience, and five respondents (23.8%) have 3 to 5 years of experience, reflecting a mix of relatively new and moderately experienced professionals. This experience

distribution aligns with the Age and Highest Academic Degree seen in Questions 2 and 4.

Additionally, two respondents (9.5%) have 5 to 8 years of experience, and six respondents (28.6%) have more than 8 years of experience, providing insights from seasoned professionals who have likely witnessed the evolution of software testing methodologies. Notably, there are no responses from individuals with less than 1 year of IT experience, which suggests that all participants have a solid foundation in IT, enhancing the reliability of their feedback on AATPAIS.

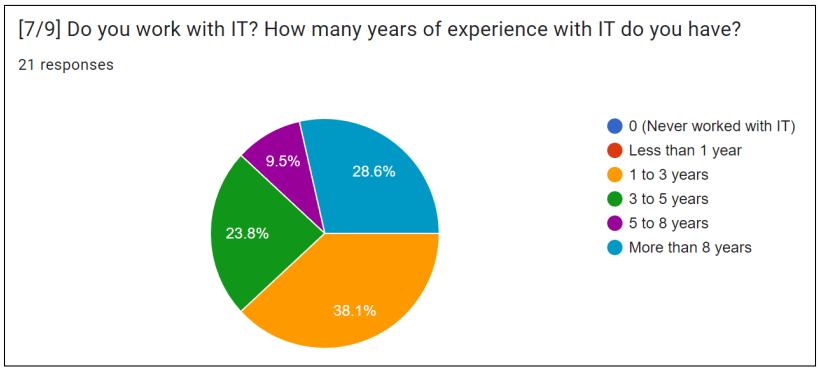


Figure 5.8: Survey Demographics - Question 7: IT Experience

Regarding prior use of Process-Aware Information Systems (PAIS), a majority, 13 out of 21 respondents (61.9%), have used PAIS before. This familiarity with PAIS is beneficial as it ensures that the respondents have a relevant background and can provide meaningful feedback on AATPAIS. Conversely, 6 respondents (28.6%) have not used PAIS, and 2 respondents (9.5%) are not sure if they have used it. This mix of experience levels (Fig. 5.9) helps gauge the accessibility and usability of AATPAIS for both experienced and novice users of PAIS.

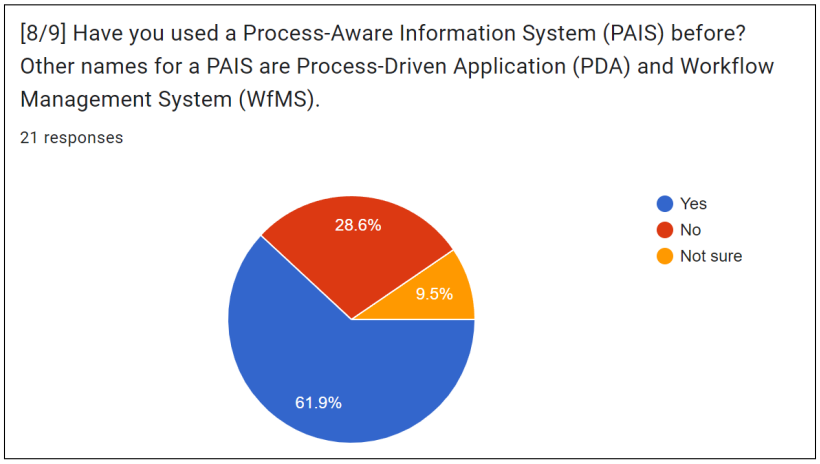


Figure 5.9: Survey Demographics - Question 8: Experience with PAIS

In terms of experience with Robotic Process Automation (RPA) tools, 10 out

of 21 respondents (47.6%) have used RPA tools before, indicating a fair level of familiarity with automation technologies among the participants. Eight respondents (38.1%) have not used RPA tools, and 3 respondents (14.3%) are not sure. This balanced distribution (Fig. 5.10) highlights the varying levels of exposure to RPA technology, which is valuable for assessing how AATPAIS is perceived by both experienced and inexperienced users in terms of automation capabilities.

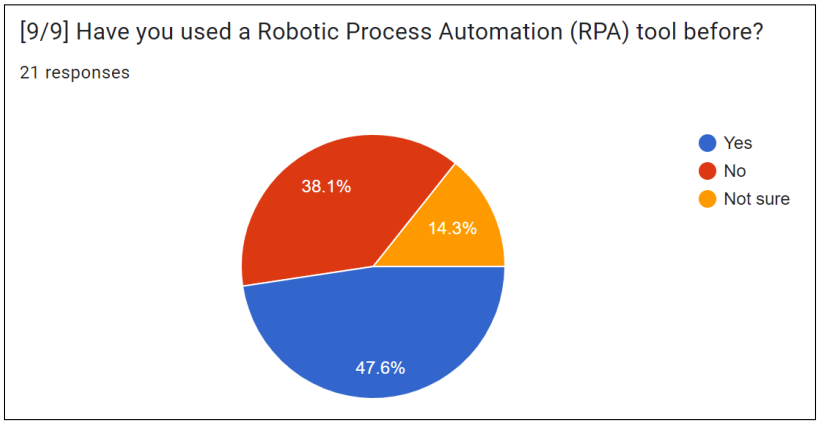


Figure 5.10: Survey Demographics - Question 9: Experience with RPA Tools

5.3.4 Results

Perceived Usefulness

Question 1: I found AATPAIS helpful for the creation and specification of executable test cases

The responses to this question (Fig. 5.11) indicate a generally positive perception of AATPAIS's usefulness in creating and specifying executable test cases. Out of 21 respondents, 7 respondents (33.3%) strongly agreed (rating of 5) that AATPAIS was helpful in this regard. An additional 12 respondents (57.1%) agreed (rating of 4), suggesting a high level of satisfaction with the tool's functionality.

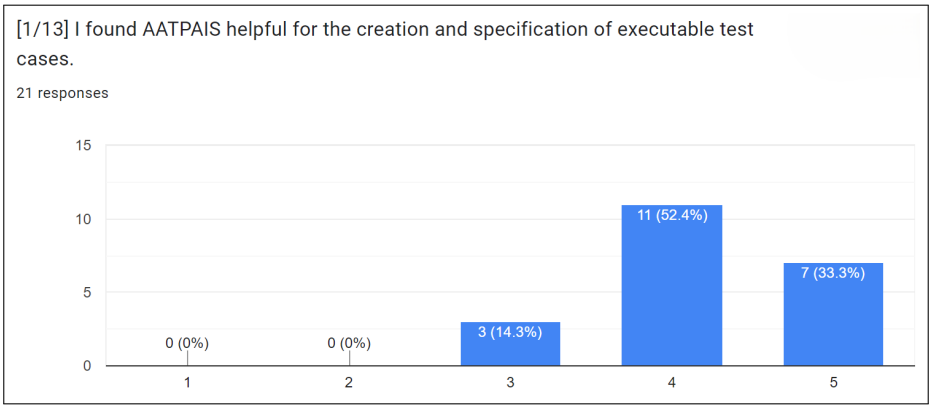


Figure 5.11: Perceived Usefulness and Ease-of-Use Questionnaire - Question 1

Meanwhile, 3 respondents (14.3%) gave a neutral rating (rating of 3), indicating some ambivalence about its helpfulness. No respondents rated the tool with a 2 or 1, which signifies that there was no strong disagreement or dissatisfaction reported.

Overall, 90.4% of the respondents rated AATPAIS as either 4 or 5, demonstrating a strong consensus that the tool is effective for the creation and specification of executable test cases. This high level of positive feedback underscores the potential of AATPAIS to improve the test case development process in BPMN-based PAIS.

Question 2: Using AATPAIS would enable testing to be accomplished more quickly

The responses to this question (Fig. 5.12) reflect a strong belief among respondents that AATPAIS can expedite the testing process. Out of 21 respondents, 12 respondents (57.1%) strongly agreed (rating of 5) that AATPAIS would enable testing to be accomplished more quickly. Additionally, 5 respondents (23.8%) agreed (rating of 4), indicating a substantial majority who view the tool as effective in speeding up testing activities.

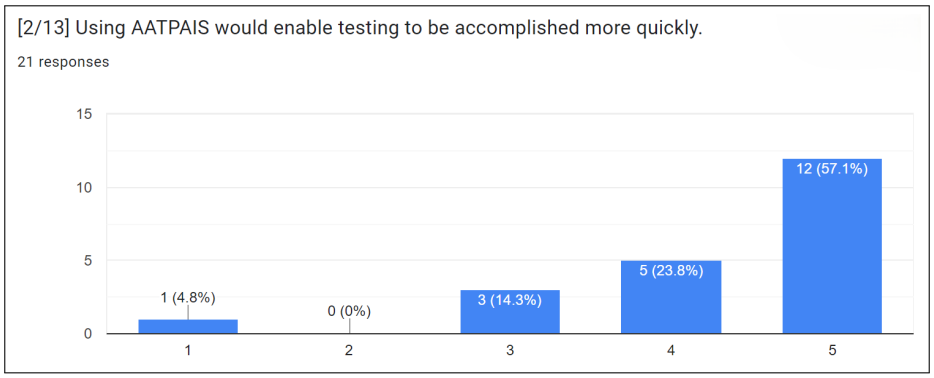


Figure 5.12: Perceived Usefulness and Ease-of-Use Questionnaire - Question 2

Three respondents (14.3%) provided a neutral rating (rating of 3), suggesting some uncertainty about the impact of AATPAIS on testing speed. Only 1 respondent (4.8%) strongly disagreed (rating of 1), indicating a minimal level of skepticism regarding the tool's efficiency.

Overall, 80.9% of the respondents rated AATPAIS as either 4 or 5, highlighting a strong consensus that the tool contributes to faster testing processes. This positive feedback underscores the perceived efficiency benefits of using AATPAIS in BPMN-based PAIS environments, affirming its potential to enhance testing productivity.

Question 3: Using AATPAIS would increase the productivity

The responses to this question (Fig. 5.13) show a strong agreement among respondents that AATPAIS would enhance productivity. Out of 21 respondents,

10 respondents (47.6%) strongly agreed (rating of 5) that using AATPAIS would increase productivity. An additional 9 respondents (42.9%) agreed (rating of 4), demonstrating a broad consensus on the productivity benefits of the tool.

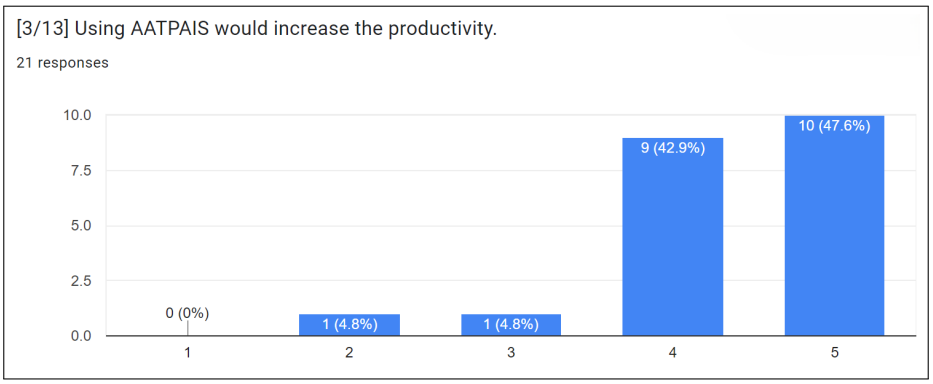


Figure 5.13: Perceived Usefulness and Ease-of-Use Questionnaire - Question 3

One respondent (4.8%) gave a neutral rating (rating of 3), indicating some uncertainty about the productivity impact of AATPAIS, while another respondent (4.8%) disagreed slightly (rating of 2), suggesting a minor level of skepticism.

Overall, 90.4% of the respondents rated AATPAIS as either 4 or 5, indicating a strong belief that the tool would significantly boost productivity. This overwhelming positive feedback highlights the effectiveness of AATPAIS in enhancing productivity in BPMN-based PAIS, reinforcing its value proposition for potential users.

Question 4: AATPAIS increases my motivation and willingness to create test cases for process models

The responses to this question (Fig. 5.14) indicate a generally positive impact of AATPAIS on users' motivation and willingness to create test cases for process models. Out of 21 respondents, 4 respondents (19.0%) strongly agreed (rating of 5) that AATPAIS increased their motivation and willingness. A significant majority, 14 respondents (66.7%), agreed (rating of 4), showing widespread acceptance of the motivational benefits of the tool.

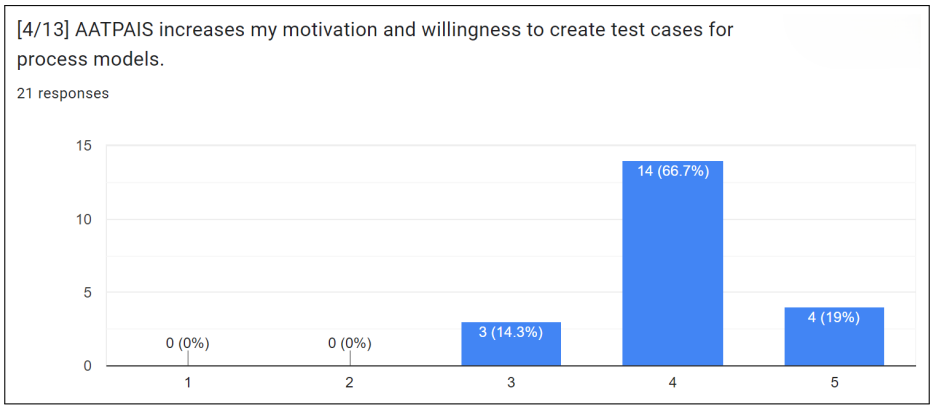


Figure 5.14: Perceived Usefulness and Ease-of-Use Questionnaire - Question 4

Three respondents (14.3%) provided a neutral rating (rating of 3), indicating some ambivalence regarding the motivational impact of AATPAIS. Notably, no respondents rated the tool with a 2 or 1, suggesting that there was no strong disagreement or negative sentiment about its ability to enhance motivation.

Overall, 85.7% of the respondents rated AATPAIS as either 4 or 5, highlighting a strong consensus that the tool positively influences their motivation and willingness to create test cases. This positive feedback underscores AATPAIS's potential to encourage more active engagement in the testing process within BPMN-based PAIS.

Question 5: Using AATPAIS would improve testing performance

The responses to this question (Fig. 5.15) reflect a strong belief that AATPAIS can enhance testing performance. Out of 21 respondents, 9 respondents (42.9%) strongly agreed (rating of 5) that using AATPAIS would improve testing performance. An additional 7 respondents (33.3%) agreed (rating of 4), indicating a substantial majority who view the tool as beneficial for improving performance.

Five respondents (23.8%) provided a neutral rating (rating of 3), suggesting some uncertainty about the impact of AATPAIS on testing performance. Notably, no respondents rated the tool with a 2 or 1, indicating that there was no strong disagreement or negative sentiment regarding its effectiveness.

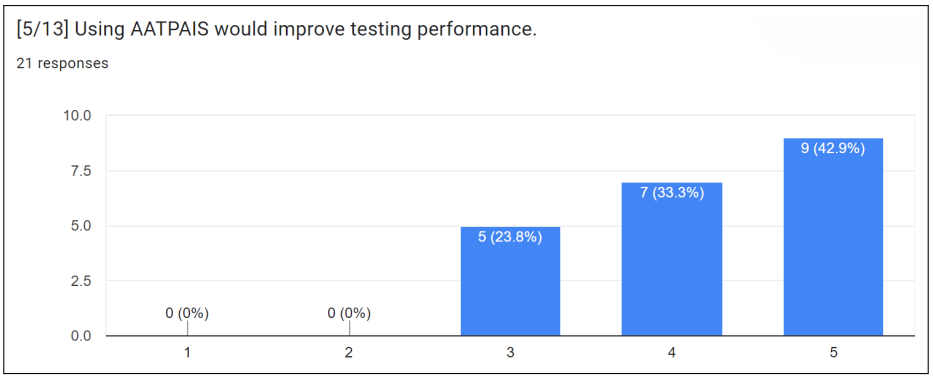


Figure 5.15: Perceived Usefulness and Ease-of-Use Questionnaire - Question 5

Overall, 76.2% of the respondents rated AATPAIS as either 4 or 5, highlighting a strong consensus that the tool contributes positively to testing performance. This positive feedback reinforces the potential of AATPAIS to enhance the efficiency and effectiveness of testing processes in BPMN-based PAIS environments.

Question 6: Using AATPAIS would make it easier to do my job

The responses to this question (Fig. 5.16) suggest that AATPAIS is generally perceived as a tool that can simplify users' job functions. Out of 21 respondents, 5 respondents (23.8%) strongly agreed (rating of 5) that using AATPAIS would make their jobs easier. An additional 8 respondents (38.1%) agreed (rating of 4), indicating that the majority of respondents found the tool helpful in easing their job responsibilities.

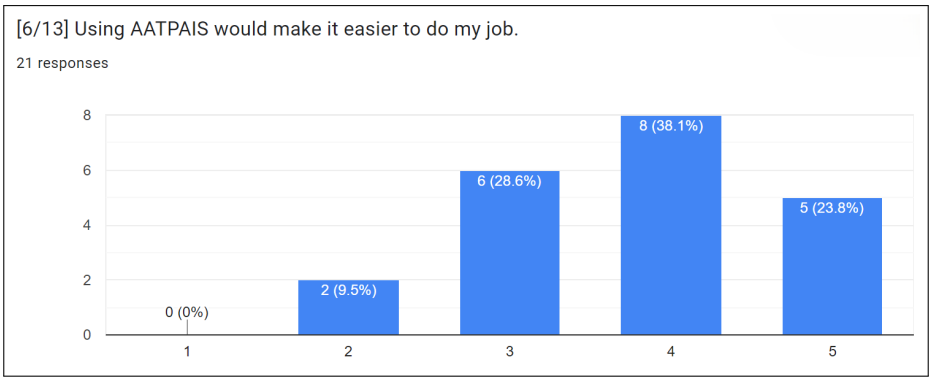


Figure 5.16: Perceived Usefulness and Ease-of-Use Questionnaire - Question 6

Six respondents (28.6%) provided a neutral rating (rating of 3), showing some level of uncertainty about AATPAIS's ability to make their job easier. Two respondents (9.5%) slightly disagreed (rating of 2), suggesting a small portion of users did not find the tool particularly helpful in this regard.

Overall, this is one of the questions within the category "*Perceived Usefulness*" with the strongest negative feedback. It is the first indicator that the respondents

might have considered AATPAIS somehow difficult to use. Although there was a majority of positive answers (ratings 4 and 5), this feedback underscores that the potential of AATPAIS to streamline job functions and enhance job performance for users working with BPMN-based PAIS is considered useful, but not easy to use.

Question 7: I would find AATPAIS useful in my job

The responses to this question (Fig. 5.17) indicate a slightly higher level of perceived usefulness of AATPAIS in respondents' professional roles. Out of 21 respondents, 6 respondents (28.6%) strongly agreed (rating of 5) that they would find AATPAIS useful in their job. An additional 12 respondents (57.1%) agreed (rating of 4), suggesting a broad consensus on the utility of the tool.

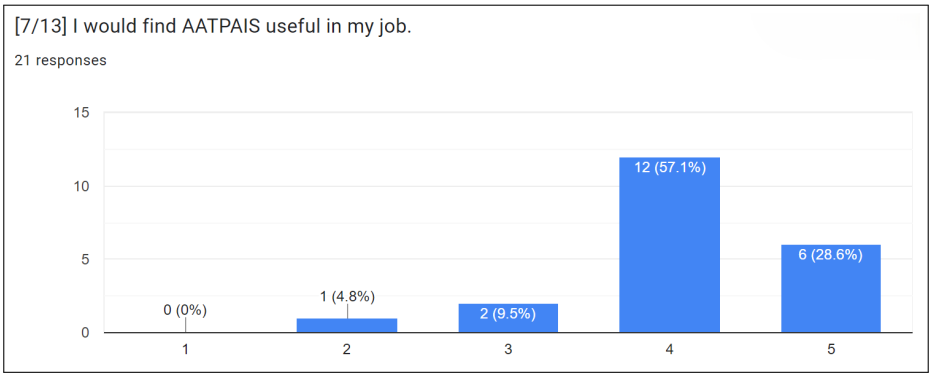


Figure 5.17: Perceived Usefulness and Ease-of-Use Questionnaire - Question 7

Two respondents (9.5%) provided a neutral rating (rating of 3), indicating some ambivalence about the usefulness of AATPAIS, while 1 respondent (4.8%) slightly disagreed (rating of 2), suggesting a minimal level of skepticism regarding the tool's utility.

Overall, most of the respondents rated AATPAIS as either 4 or 5, demonstrating a strong consensus that the tool would be useful in their job roles. This overwhelming positive feedback underscores the practicality and applicability of AATPAIS in professional environments, particularly for those involved in BPMN-based PAIS.

Perceived Ease-of-Use

Question 8: I would find it easy to get AATPAIS to do what I want it to do

The responses to this question (Fig. 5.18) reflect a mixed but generally positive perception of AATPAIS's ease of use in terms of achieving desired tasks. Out of 21 respondents, 6 respondents (28.6%) strongly agreed (rating of 5) that they would find it easy to get AATPAIS to do what they want it to do. An additional 5

respondents (23.8%) agreed (rating of 4), indicating that a substantial number of users found the tool user-friendly.

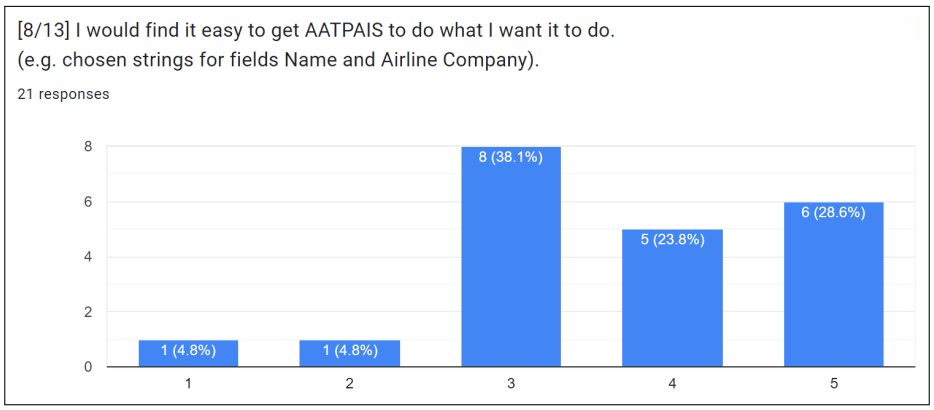


Figure 5.18: Perceived Usefulness and Ease-of-Use Questionnaire - Question 8

Eight respondents (38.1%) provided a neutral rating (rating of 3), suggesting a significant portion of users were unsure about the tool's ease of use. One respondent (4.8%) strongly disagreed (rating of 1), and one respondent (4.8%) slightly disagreed (rating of 2), indicating a small level of dissatisfaction or difficulty in using the tool.

Overall, 52.4% of the respondents rated AATPAIS as either 4 or 5, showing a majority who believe the tool is relatively easy to use. However, the notable percentage of neutral responses and the presence of some disagreement indicate areas where AATPAIS could be improved to enhance user-friendliness and better meet user expectations. This result aligns with what was indicated in the answers from Question 6.

Question 9: Learning to operate AATPAIS would be easy for me

The responses to this question (Fig. 5.19) indicate a generally neutral to slightly positive perception of the ease of learning to operate AATPAIS. Out of 21 respondents, only 4 respondents (19.0%) strongly agreed (rating of 5) that learning to operate AATPAIS would be easy for them. An additional 7 respondents (33.3%) agreed (rating of 4), suggesting that a majority found the learning process manageable.

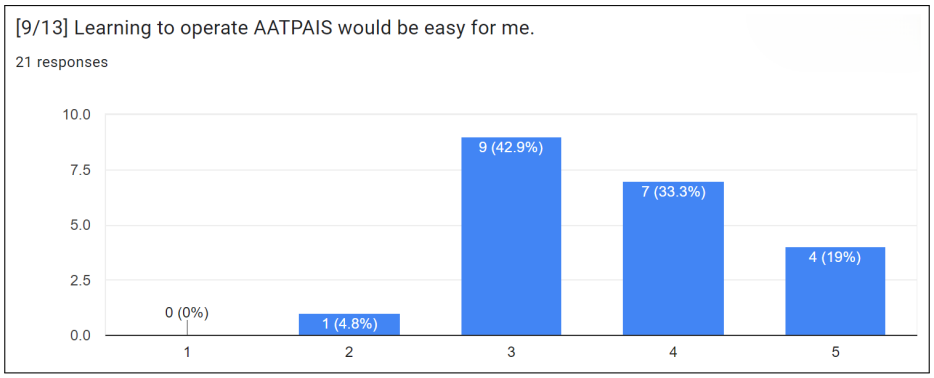


Figure 5.19: Perceived Usefulness and Ease-of-Use Questionnaire - Question 9

Nine respondents (42.9%) provided a neutral rating (rating of 3), indicating some ambivalence or uncertainty about the ease of learning to use the tool. One respondent (4.8%) slightly disagreed (rating of 2), showing a minor level of concern about the learning curve. The significant portion of neutral responses highlights an area for potential improvement in user training and onboarding processes to ensure that new users can more easily become proficient with AATPAIS.

Question 10: The test cases seemed to be clearly organized to me

The responses to this question (Fig. 5.20) reflect a strong positive perception regarding the organization of test cases in AATPAIS. Out of 21 respondents, 8 respondents (38.1%) strongly agreed (rating of 5) that the test cases seemed to be clearly organized. An additional 9 respondents (42.9%) agreed (rating of 4), indicating a broad consensus on the clarity of test case organization.

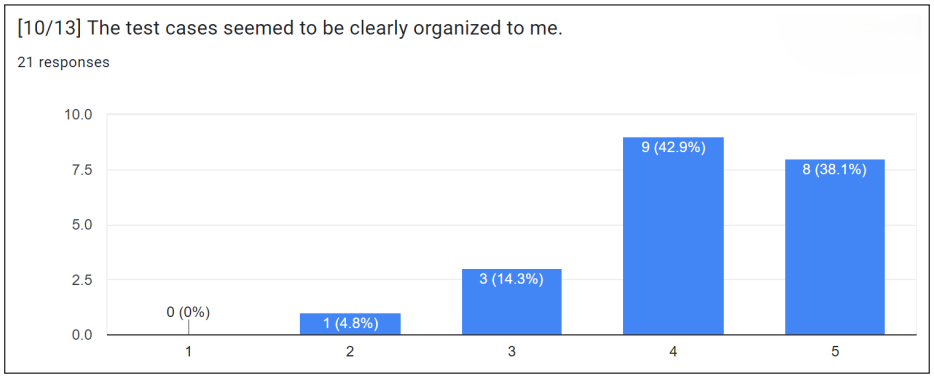


Figure 5.20: Perceived Usefulness and Ease-of-Use Questionnaire - Question 10

Three respondents (14.3%) provided a neutral rating (rating of 3), suggesting some ambivalence about the clarity of the test case organization. Only a single respondent (4.8%) slightly disagreed (rating of 2), indicating a minor level of dissatisfaction.

Overall, 81% of the respondents rated AATPAIS as either 4 or 5, demonstrating a strong consensus that the test cases are well-organized and easy to understand. This positive feedback underscores the effectiveness of AATPAIS in presenting test cases in a clear and organized manner, enhancing the overall user experience and usability of the tool. One possible reason for this positive perception is the use of Robot Framework as the RPA tool, which is known for its well-structured and well-documented format. The proven organization and clarity of Robot Framework syntax likely contribute to the overall ease of understanding and usability experienced by the respondents.

Question 11: My interaction with AATPAIS would be clear and understandable

The responses to this question (Fig. 5.21) indicate a generally positive perception of the clarity and understandability of interactions with AATPAIS. Out of 21 respondents, 7 respondents (33.3%) strongly agreed (rating of 5) that their interaction with AATPAIS would be clear and understandable. An additional 7 respondents (33.3%) agreed (rating of 4), suggesting a substantial majority found the interactions intuitive.

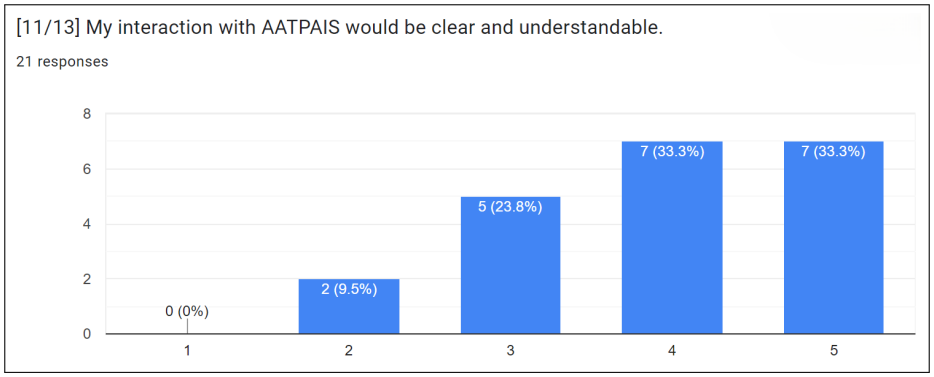


Figure 5.21: Perceived Usefulness and Ease-of-Use Questionnaire - Question 11

Five respondents (23.8%) provided a neutral rating (rating of 3), indicating some ambivalence about the clarity of their interactions with the tool. Two respondents (9.5%) slightly disagreed (rating of 2), suggesting a minor level of difficulty or confusion.

Overall, two-thirds of the respondents rated AATPAIS as either 4 or 5, highlighting a strong consensus that interactions with the tool are clear and understandable. The clarity and ease of understanding are likely enhanced by the structured design principles of Robot Framework, which contributes to making user interactions with AATPAIS more intuitive and straightforward.

Question 12: I would find AATPAIS easy to use

The responses to this question (Fig. 5.22) indicate a concern regarding the perception of the ease of use of AATPAIS. Out of 21 respondents, 5 respondents (23.8%) strongly agreed (rating of 5) that they would find AATPAIS easy to use. An additional 4 respondents (19.0%) agreed (rating of 4), indicating that a majority found the tool user-friendly.

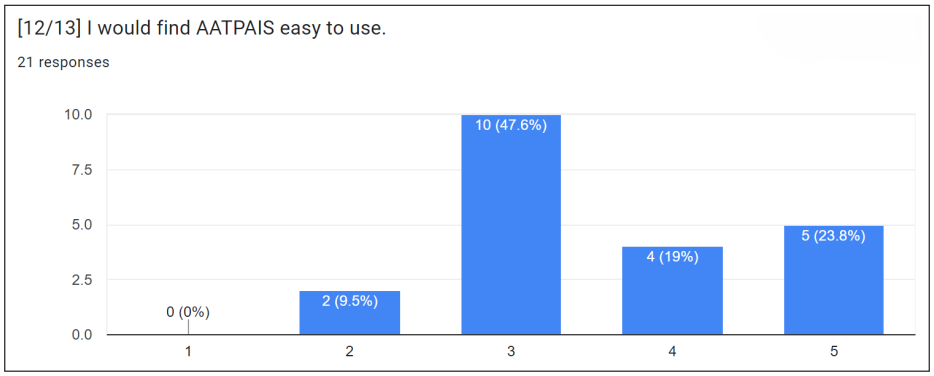


Figure 5.22: Perceived Usefulness and Ease-of-Use Questionnaire - Question 12

However, a significant portion of respondents, 10 respondents (47.6%), provided a neutral rating (rating of 3), suggesting ambivalence about the ease of use. This is the highest number of neutral ratings recorded in all of 13 questions, with the second highest being Question 9, in which the respondents expressed neutrality towards the easiness in operating AATPAIS. Furthermore, two respondents (9.5%) slightly disagreed (rating of 2), indicating some challenges or difficulties in using the tool.

Overall, the notable percentage of neutral responses highlights the need for further improvements in the user interface and user experience to ensure that AATPAIS is more universally perceived as easy to use. This feedback suggests that while AATPAIS is generally considered user-friendly and well-organized, there are areas where the tool could be refined to enhance overall usability and address any user concerns or difficulties.

Question 13: It would be easy for me to become skillful at using AATPAIS

The responses to this question (Fig. 5.23) suggest a generally positive outlook on the ease of becoming skillful at using AATPAIS, though with some variation in perceptions. Out of 21 respondents, only 4 respondents (19.0%) strongly agreed (rating of 5) that it would be easy for them to become skillful at using AATPAIS. An additional 9 respondents (42.9%) agreed (rating of 4), indicating that the majority believe they could quickly gain proficiency with the tool.

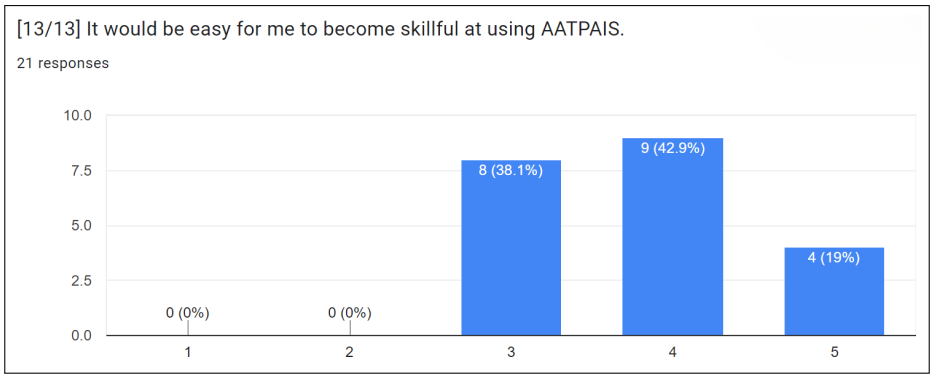


Figure 5.23: Perceived Usefulness and Ease-of-Use Questionnaire - Question 13

Eight respondents (38.1%) provided a neutral rating (rating of 3), suggesting some uncertainty or ambivalence about how easy it would be to become skillful at using AATPAIS. Notably, no respondents rated the tool with a 1 or 2, which indicates that there were no strong disagreements or significant concerns about the ability to become proficient. This, again, could be explained by the use of Robot Framework as the RPA tool, which counts with an extensive and well-known documentation base.

Overall, this positive feedback underscores the potential for users to quickly become adept at using AATPAIS, enhancing their ability to leverage the tool effectively in their work. The feedback suggests that while most users feel confident in their ability to learn and use AATPAIS proficiently, continued efforts to streamline the learning process and provide comprehensive training resources could further improve user experience and satisfaction.

5.4 Discussion

The direct execution of automation scripts achieved 100% Human Interaction Coverage in 18 out of the 21 processes. In the three cases where complete coverage was not possible, it was due to exclusive gateway conditions that required specific field values not generated by the Faker Library. Simple customization of the RF automation scripts to meet these gateway conditions would achieve complete interface coverage and would free testers from tedious and repetitive tasks, although some time is still required to interpret result logs.

Regarding Path Coverage (PC), the execution achieved 100% coverage in 15 out of the 21 selected process models, eliminating the need for further customization by testers or developers. This approach further frees testers from tedious and repetitive tasks. These 15 process models can be continuously tested using a regression testing strategy, facilitating the identification of issues and bugs stemming from changes in the code or interface that are not directly related to process model changes.

For the six process models that did not achieve total path coverage, issues included the inability to trigger specific events, such as Message and Timer Boundary Events, and unsatisfied exclusive gateway conditions due to specific field requirements that were not met by the Faker Library's random text generation. In the case of inclusive gateways, the conditions were incompatible with the single selection nature of the `Type` field, preventing multiple conditions from being satisfied simultaneously.

The Simulation results underscore the potential of AATPAIS to enhance various aspects of the testing process within BPMN-based PAIS, despite some notable limitations. On the other hand, the Survey results reveal a dichotomy in the perceptions of AATPAIS's usefulness and ease-of-use.

The responses to the questions regarding perceived usefulness were overwhelmingly positive, indicating that users generally found AATPAIS helpful and beneficial for their tasks. For instance, 90.4% of respondents agreed or strongly agreed that AATPAIS was helpful for creating and specifying executable test cases, and 80.9% believed it would expedite the testing process. Similarly, a strong majority felt that AATPAIS would increase productivity and improve testing performance, with 90.4% and 76.2% positive responses, respectively.

The high levels of agreement reflect a consensus among users that AATPAIS can significantly improve their development workflows, boost productivity, and streamline job functions. This positive reception highlights the tool's effectiveness in addressing key challenges in test case development and execution within the context of a PAIS.

However, the feedback on the perceived ease-of-use presents a more nuanced picture. While the majority of respondents found AATPAIS relatively easy to use, there were notable concerns regarding the learning curve and ease of operation. For example, only 52.4% of respondents agreed or strongly agreed that they would find it easy to get AATPAIS to do what they wanted, and a significant 38.1% provided neutral responses, indicating uncertainty.

Similarly, when asked about the ease of learning to operate AATPAIS, only 52.3% of respondents agreed or strongly agreed, while 42.9% remained neutral. This ambivalence suggests that while AATPAIS is generally user-friendly given the usage of Robot Framework's syntax, there are areas that require improvement to make the tool more accessible and intuitive for new users.

The organization of test cases was one area where AATPAIS received strong positive feedback regarding the ease-of-use, with 81.0% of respondents agreeing or strongly agreeing that the test cases were clearly organized. This clarity is likely due to the structured design principles of Robot Framework, which AATPAIS leverages for its RPA functionality. The well-documented and organized format of Robot

Framework contributes to the overall ease of understanding and usability experienced by respondents.

Nonetheless, the survey results indicate a need for further refinement in the user interface and training materials. Ensuring that new users can easily learn and become proficient with AATPAIS is crucial for a wider adoption and effectiveness. Enhanced training resources and documentation, more intuitive interface designs, and comprehensive onboarding processes could help address the concerns raised by respondents and improve the overall user experience.

In conclusion, while the survey results affirm the usefulness and potential of AATPAIS, they also highlight areas for improvement in ease-of-use. By addressing these concerns, AATPAIS can better meet user expectations and fully realize its potential as a powerful tool for Automated Acceptance Testing in BPMN-based PAIS.

5.5 Threats to Validity

The evaluation of AATPAIS's *Usefulness and Ease-of-Use* through the survey provided valuable insights, but several potential threats to validity specific to the survey should be considered.

5.5.1 Sampling Bias

The survey was distributed through the main communication channels of AgileKIP, Robot Framework Foundation, and Camunda user groups. While these channels are appropriate for reaching the intended audience, they may introduce sampling bias. The participants who are active in these communities might be more familiar with the tools and technologies being evaluated, which could skew the results positively.

5.5.2 Limited Sample Size

With only 21 respondents, the sample size is relatively small. This limited number of responses may not fully capture the diversity of opinions and experiences of the broader user base. Consequently, the findings might not generalize to all potential users of AATPAIS. A larger sample size would provide a more robust dataset for analysis.

5.5.3 Self-Selection Bias

Participation in the survey was voluntary, which introduces self-selection bias. Those who chose to respond might have a particular interest or positive disposition towards

BPMN, RPA, or the AgileKIP, Robot Framework, and Camunda communities. This could result in an overrepresentation of positive feedback and an underrepresentation of more critical views.

5.5.4 Demonstration Bias

A potential bias in the survey arises from the fact that the respondents were presented with a detailed tutorial and demonstration video of AATPAIS, rather than using the tool firsthand. While this approach provided a comprehensive overview of AATPAIS's functionality, it may have influenced their perception of the tool, as watching a demonstration does not fully replicate the hands-on experience of interacting with it. This lack of practical experience could impact the feedback on the perceived ease of use and usefulness, as challenges and nuances that might emerge from actual usage, such as learning curves, unexpected behaviors, or workflow limitations, would not be fully evident in a video demonstration.

As a result, respondents might have developed an overly optimistic or incomplete understanding of the tool's usability and effectiveness, leading to perceptions based on how the tool was presented rather than on direct usage. Future evaluations could benefit from incorporating a hands-on trial phase to allow respondents to engage with the tool directly before providing feedback.

5.5.5 Experience and Expertise Variability

The respondents' varying levels of experience and expertise with PAIS and RPA tools could influence their feedback. While the survey included participants with a range of experience levels, those with more familiarity and expertise might provide more informed and favorable responses. Conversely, less experienced users might find the tool more challenging to use, which could be underrepresented in the survey results.

5.5.6 Recency of Technology Adoption

The adoption of BPMN and RPA technologies is relatively recent, which means that best practices and user experiences are still evolving. This might affect the respondents' ability to provide comprehensive feedback based on long-term use. The limited longitudinal data available for these technologies could influence the perceived usefulness and ease-of-use of AATPAIS.

5.5.7 Subjectivity in Responses

Survey responses are inherently subjective and can be influenced by individual preferences, expectations, and experiences. This subjectivity can introduce variability in

the results, making it challenging to draw definitive conclusions. The interpretation of questions and the Likert scale ratings might vary among respondents, affecting the consistency of the feedback.

By acknowledging these potential threats to validity, future research can aim to address these limitations through more extensive and diverse sampling, longitudinal studies, and the development of standardized evaluation frameworks for BPMN-based PAIS and RPA integrations.

Chapter 6

Conclusion

6.1 Introduction

Our Automated Acceptance Testing solution for BPMN-based Process-Aware Information Systems (PAIS), named AATPAIS, represents a novel approach for testing automation of this type of software systems. It not only streamlines testing processes but also frees developers and testers from manual testing efforts, enabling them to focus on spot customization and higher-value activities. Through AATPAIS, we have demonstrated significant advancements in achieving comprehensive path coverage in the majority of selected process models.

6.2 Contributions

By leveraging the flexibility and customization capabilities of our solution, organizations can effectively overcome these challenges and ensure comprehensive testing of their systems. Additionally, AATPAIS offers significant time savings and efficiency, allowing organizations to optimize their testing practices and accelerate their development cycles.

From the perspective of generating automation scripts from scripts associated with the 21 processes in Table 5.1, the simplest ones are related to the `CF-only-suggestion-compliment` process model, which can be seen in Figure 6.1.

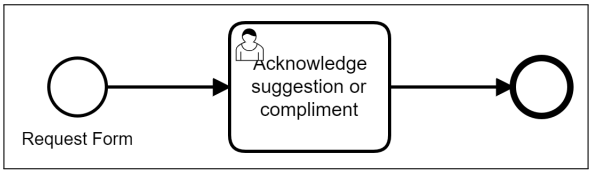


Figure 6.1: Model of the Customer Feedback Process with only a single User Task.

However, even the simplest process model automation scripts result in `test` and

`resources` files containing around a hundred lines of Robot Framework code each. Importantly, the time required to manually write these lines of code was not spent by a developer or tester, allowing them to focus on more critical tasks.

To illustrate the time-saving benefits not only during the generation but also the execution of test suites, a subset of three business processes with the highest scores in `size` and `complexity` was selected for comparison. These processes were tested both manually and using the proposed solution:

- TP-OR;
- TP-XOR;
- CF-inclusive-all-types-with-scalation (referred as "`CF-iatwe`" in the table).

The results, shown in Table 6.1, illustrate the significant time savings involved in such tests.

The *Manual* column represents the average duration necessary to guarantee 100% Path and NOHA coverage. Each process model was executed the number of times indicated in the *Paths* column, in a threefold strategy performed by a tester fully acquainted with the platform. The *Automated* column shows the average time necessary to execute three independent sets of 30 automated executions for each process model.

Table 6.1: Comparison of Manual and Automated Test Suite Execution Times

	Size&Complexity		Manual	Automated		
Process Model	Paths	NOHA	Time	Time	PC	NOHAC
TP-OR	4	4	3min25s	4min43s	100%	100%
TP-XOR	4	4	3min12s	4min48s	100%	100%
CF-iatwe	5	4	3min50s	2min59s	60%	100%

These two types of executions achieve similar results and require comparable time. However, the automated approach eliminates the need for a human operator to execute tests manually, freeing developers or testers to focus on more meaningful tasks while the RPA executes the tests.

In both simple and complex process models, if 100% test coverage is not achieved directly from automated generation and execution, the RF scripts can be further customized to reach the desired coverage. In such cases, the tester only needs to spend time on specific customizations, as the core structure of the scripts is already generated by the solution.

This approach offers several benefits by enabling the use of modern IDE capabilities for file comparison and editing, as well as `git` for version control. This allows

for parallel customization by different testers and developers in a team. Once these adjustments are made and the results are satisfactory, the curated set of test suites can be utilized for an automated regression testing strategy in the following manner:

- New process
    1. Generate the new Test Suite from the files (BPMN and JSONs);
    2. Customize the RF files, if necessary;
    3. Save these RF files in a separate folder to use for Regression Testing.
- Evolving process
    1. Generate the new Test Suite from the new version of the files (BPMN and JSONs);
    2. Compare the previous and new RF files to accept, ignore or propose further changes;
    3. Save this new version of RF files in the Regression Testing folder.

6.3 Limitations

However, challenges remain in scenarios where specific conditions prevent complete path coverage, emphasizing the need for tailored customization to effectively address such complexities. Despite the benefits provided by AATPAIS, these challenges must still be tackled to ensure comprehensive testing.

The current implementation has not yet addressed gateway-specific conditions, such as those mentioned in items **1(b)**, **2**, and **5** regarding path coverage in the Results discussion in Section 5.2.5. Similarly, issues related to Timer and Message Intermediate and Boundary Events decrease overall path coverage and may render some human interaction points unreachable, as noted in items **1(a)** and **4(a)** of Section 5.2.6.

The Robot Framework Camunda Library[1] offers a solution for handling Message Intermediate and Boundary Events through its `Deliver Message` keyword implementation. Given the focus on user interface testing, no mechanism for tracking the outcomes of Service Tasks was developed. However, a clear path for implementing such a mechanism can be established using the Robot Framework Requests Library[2] in combination with the aforementioned Robot Framework Camunda Library.

Given the absence of a mechanism to track Service Task outcomes and the process's default synchronous execution, concerns about potential testing deadlocks

[1] https://pypi.org/project/robotframework-camunda/
[2] https://docs.robotframework.org/docs/different_libraries/requests

arise. The process engine's default behavior is to revert the process token to the last "checkpoint" following an unsuccessful Service Task execution. This could lead to a scenario where the process continuously cycles back to the same User Task after each failed Service Task execution.

Another potential deadlock scenario can occur when a Gateway-specific condition is never met, resulting in an endless loop back to the same User Task upon evaluation. This contrasts with the case described in item **4** of the Results discussion in Sections 5.2.5 and 5.2.6. In that instance, the Exclusive Gateway condition of the process model named `TP-LOOP` was never fulfilled, preventing the return path from being taken. Conversely, if the condition were continually met, the process could become stuck in a deadlock situation.

To mitigate these potential deadlock scenarios, careful design and evaluation of Gateway conditions and Service Task outcomes are essential to ensure they can be satisfactorily met during process execution. Additionally, implementing timeout mechanisms or fallback strategies can help prevent test cases from becoming indefinitely stuck in a deadlock state.

Lastly, automated tests are performed by the RPA using a superuser with `Admin` privileges. Consequently, test cases involving restrictions on viewing and executing tasks based on specific roles or profiles have not been addressed. Incorporating these aspects is an important topic for future development.

6.4 Future Work

While our current study has highlighted several critical aspects of automated acceptance testing for BPMN-based Process-Aware Information Systems (PAIS), there remain numerous opportunities for future exploration and enhancement.

First and foremost, implementing a stop condition for identified deadlocks, such as loops and Service Tasks, is essential to ensure the robustness and reliability of our testing approach. Addressing Timer and Message Boundary and Intermediate Events is also paramount.

Additionally, creating more detailed execution logs would provide invaluable insights into the testing process, facilitating more comprehensive and expedited analysis and debugging. This improvement would reduce the time required to review execution logs and verify path coverage, thereby accelerating the overall testing process.

Furthermore, enhancing the `Arguments` and `Tags` strategy could greatly improve the flexibility and efficiency of our regression testing framework, facilitating the passing of specific parameters to test cases and keywords. The Survey results clearly indicate the necessity for this improvement.

Moreover, efforts to facilitate changes in field types used by the RPA for simulated data and input behavior—including enabling random selection from predetermined lists and improving constraints for Date and Time—would greatly contribute to the versatility and effectiveness of our automated testing framework.

By enabling the automated generation and execution of User Acceptance Tests for BPMN-based PAIS, the proposed solution significantly advances the development of methodologies that support automated testing based on human interactions in process-aware systems. This foundation serves as a basis for further exploration into automated testing that accommodates the increasing complexity of these interactions, ultimately aiming to enhance software quality.

References

[1] PFLEEGER, S. L., ATLEE, J. M. *Software Engineering: Theory and Practice*. 4th ed. USA, Prentice Hall PTR, 2009. ISBN: 0130290491.

[2] PEFFERS, K., TUUNANEN, T., ROTHENBERGER, M., et al. "A design science research methodology for information systems research", *Journal of Management Information Systems*, v. 24, pp. 45–77, 2007. doi: 10.2753/MIS0742-1222240302.

[3] TELEMACO-NETO, U., OLIVEIRA, T. C., PILLAT, R. M., et al. "Scaffolding Process-Aware Information Systems with the AKIP Platform". In: *Web Information Systems and Technologies*, pp. 60–83. Springer Nature Switzerland, 2023. ISBN: 978-3-031-43088-6. doi: 10.1007/978-3-031-43088-6_4.

[4] "IEEE Standard for Software Quality Assurance Processes", *IEEE Std 730-2014*, pp. 1–138, 2014. doi: 10.1109/IEEESTD.2014.6835311.

[5] "ISO Standard for Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models", *ISO/IEC 25010:2011*, 2011.

[6] "IEEE Standard Glossary of Software Engineering Terminology", *ANSI/IEEE Std 729-1983*, pp. 1–40, 1983. doi: 10. 1109/IEEESTD.1983.7435207.

[7] IEEE. "IEEE Standard for System, Software, and Hardware Verification and Validation", *IEEE Std 1012-2016*, pp. 1–260, 2017. doi: 10.1109/IEEESTD.2017.8055462.

[8] DUMAS, M., VAN DER AALST, W. M. P., TER HOFSTEDE, A. H. M. *Process-aware information systems : bridging people and software through process technology*. USA, Wiley-Interscience, 2005. ISBN: 978-0-471-66306-5. doi: 10.1002/0471741442.

[9] VAN DER AALST, W. M. P. "Process-Aware Information Systems: Lessons to Be Learned from Process Mining", *T. Petri Nets and Other Models of Concurrency*, v. 2, pp. 1–26, 2009. doi: 10.1007/978-3-642-00899-3_1.

[10] TELEMACO-NETO, U., OLIVEIRA, T. C., PILLAT, R. M., et al. "AKIP Process Automation Platform: A Framework for the Development of Process-Aware Web Applications". In: *Proceedings of the 18th International Conference on Web Information Systems and Technologies - WEBIST*, pp. 64–74. INSTICC, SciTePress, 2022. ISBN: 978-989-758-613-2. doi: 10.5220/0011550000003318.

[11] OBJECT MANAGEMENT GROUP. "Business Process Model and Notation (BPMN), Version 2.0". December 2010. Disponível em: <http://www.omg.org/spec/BPMN/2.0>.

[12] "ISO Standard for Information technology — Object Management Group Business Process Model and Notation", *ISO/IEC 19510:2013*, 2013.

[13] SEQERLOO, A. Y., AMIRI, M. J., PARSA, S., et al. "Automatic test cases generation from business process models", *Requirements Engineering*, v. 24, pp. 119–132, 3 2019. ISSN: 1432010X. doi: 10.1007/s00766-018-0304-3.

[14] SCHNEID, K., STAPPER, L., THÖNE, S., et al. "Automated Regression Tests: A No-Code Approach for BPMN-based Process-Driven Applications". In: *2021 IEEE 25th International Enterprise Distributed Object Computing Conference (EDOC)*, pp. 31–40, 2021. doi: 10.1109/EDOC52215.2021.00014.

[15] UTTING, M., LEGEARD, B., BOUQUET, F., et al. "Chapter Two - Recent Advances in Model-Based Testing". v. 101, *Advances in Computers*, pp. 53–120, Elsevier, 2016. doi: 10.1016/bs.adcom.2015.11.004.

[16] MOURA, J. L., CHARÃO, A. S., LIMA, J. C. D., et al. "Test case generation from BPMN models for automated testing of Web-based BPM applications". In: *2017 17th International Conference on Computational Science and Its Applications (ICCSA)*, pp. 1–7, 2017. doi: 10.1109/ICCSA.2017.7999652.

[17] FEWSTER, M., GRAHAM, D. *Software test automation: effective use of test execution tools*. USA, ACM Press/Addison-Wesley Publishing Co., 1999. ISBN: 0201331403.

[18] RAMLER, R., KLAMMER, C. "Enhancing Acceptance Test-Driven Development with Model-Based Test Generation". In: *2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, pp. 503–504, 2019. doi: 10.1109/QRS-C.2019.00096.

[19] ENRÍQUEZ, J. G., JIMÉNEZ-RAMÍREZ, A., DOMÍNGUEZ-MAYO, F. J., et al. "Robotic process automation: a scientific and industrial systematic mapping study", *IEEE Access*, v. 8, pp. 39113–39129, 2020. doi: 10.1109/ACCESS.2020.2974934.

[20] GMEINER, J., RAMLER, R., HASLINGER, J. "Automated testing in the continuous delivery pipeline: A case study of an online company". In: *2015 IEEE Eighth International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pp. 1–6. IEEE, 2015. doi: 10.1109/ICSTW.2015.7107423.

[21] HUMBLE, J., FARLEY, D. *Continuous delivery: reliable software releases through build, test, and deployment automation*. Pearson Education, 2010.

[22] LOPES, T., GUERREIRO, S. "Assessing business process models: a literature review on techniques for BPMN testing and formal verification", *Business Process Management Journal*, v. 29, pp. 133–162, 2023. ISSN: 14637154. doi: 10.1108/BPMJ-11-2022-0557.

[23] PAIVA, T. M., OLIVEIRA, T. C., PILLAT, R. M., et al. "Supporting the Automated Generation of Acceptance Tests of Process-Aware Information Systems". In: *Proceedings of the 19th International Conference on Web Information Systems and Technologies - WEBIST*, v. 1, pp. 128–139, 2023. doi: 10.5220/0012211400003584.

[24] PAIVA, T. M., SANTOS, R. F. "O sistema portuário brasileiro: um panorama dos processos informacionais de importação e exportação de cargas conteinerizadas", *Revista Gestão & Tecnologia*, v. 22, n. 2, pp. 149–174, 2022. doi: 10.20397/2177-6652/2022.v22i2.2181.

[25] JEEVA PADMINI, K., PERERA, I., DILUM BANDARA, H. M. N. "Applying agile practices to avoid chaos in User Acceptance Testing: A case study". In: *2016 Moratuwa Engineering Research Conference (MERCon)*, pp. 96–101. IEEE, 2016. doi: 10.1109/MERCon.2016.7480122.

[26] LEUNG, H. K. N., WONG, P. W. L. "A study of user acceptance tests", *Software quality journal*, v. 6, n. 2, pp. 137–149, 1997. doi: 10.1023/A:1018503800709.

[27] MELNIK, G., READ, K., MAURER, F. "Suitability of FIT User Acceptance Tests for specifying functional requirements: Developer perspective". In: *Conference on Extreme Programming and Agile Methods*, pp. 60–72. Springer, 2004. doi: 10.1007/978-3-540-27777-4_7.

[28] HAUGSET, B., HANSSEN, G. K. "Automated Acceptance Testing: A Literature Review and an Industrial Case Study". In: *Agile 2008 Conference*, pp. 27–38, 2008. doi: 10.1109/Agile.2008.82.

[29] WEISS, J., SCHILL, A., RICHTER, I., et al. "Literature Review of Empirical Research Studies within the Domain of Acceptance Testing". In: *2016 42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pp. 181–188, 2016. doi: 10.1109/SEAA.2016.33.

[30] UTTING, M., PRETSCHNER, A., LEGEARD, B. "A taxonomy of model-based testing approaches", *Software Testing Verification and Reliability*, v. 22, pp. 297–312, 8 2011. ISSN: 09600833. doi: 10.1002/stvr.456.

[31] DIAS-NETO, A. C., TRAVASSOS, G. H. "A Picture from the Model-Based Testing Area: Concepts, Techniques, and Challenges", *Advances in Computers*, v. 80, pp. 45–120, 1 2010. ISSN: 00652458. doi: 10.1016/S0065-2458(10)80002-6.

[32] LABICHE, Y., SHAFIQUE, M. *A systematic review of model based testing tool support*. Relatório técnico, Carleton University, CA, 2010. Disponível em: <`https://carleton.ca/squall/wp-content/uploads/TR_-SCE-10-04.pdf`>.

[33] MAKKI, M., LANDUYT, D. V., JOOSEN, W. "Automated regression testing of BPMN 2.0 processes: A capture and replay framework for continuous delivery", *ACM SIGPLAN Notices*, v. 52, pp. 178–189, 2016. ISSN: 15232867. doi: 10.1145/2993236.2993257.

[34] LÜBKE, D., VAN LESSEN, T. "BPMN-based Model-Driven Testing of Service-based Processes". In: *Enterprise, Business-Process and Information Systems Modeling: 18th International Conference, BPMDS 2017, 22nd International Conference, EMMSAD 2017*, p. 119–133. Springer International Publishing, 2017. ISBN: 978-3-319-59466-8. doi: 10.1007/978-3-319-59466-8_8.

[35] GAROUSI, V., MESBAH, A., BETIN-CAN, A., et al. "A systematic mapping study of web application testing", *Information and Soft-*

ware Technology, v. 55, pp. 1374–1396, 2013. ISSN: 09505849. doi: 10.1016/j.infsof.2013.02.006.

[36] AHMED, A. "Test automation for graphical user interfaces: A review". In: *2014 World Congress on Computer Applications and Information Systems (WCCAIS)*, pp. 1–6. IEEE, 2014. doi: 10.1109/WCCAIS.2014.6916544.

[37] BÖHMER, K., RINDERLE-MA, S. "A systematic literature review on process model testing: Approaches, challenges, and research directions", *CoRR*, 2015. doi: 10.48550/arXiv.1509.04076.

[38] KITCHENHAM, B., CHARTERS, S. *Guidelines for performing Systematic Literature Reviews in Software Engineering*. Relatório técnico, EBSE Technical Report, 2007.

[39] PAIVA, A. C. R., FLORES, N. H., FARIA, J. P., et al. "End-to-end Automatic Business Process Validation", *Procedia Computer Science*, v. 130, pp. 999–1004, 2018. ISSN: 1877-0509. doi: 10.1016/j.procs.2018.04.104.

[40] YOTYAWILAI, P., SUWANNASART, T. "Design of a tool for generating test cases from BPMN". In: *2014 International Conference on Data and Software Engineering (ICODSE)*, pp. 1–6, 2014. doi: 10.1109/ICODSE.2014.7062692.

[41] KHADER, S., YOUSEF, R. "Utilizing Business Process Models to Generate Software Test Cases", *International Journal of Computer Science Issues*, v. 13, pp. 1–7, 2016. ISSN: 16940814. doi: 10.20943/01201602.17.

[42] "ISO/IEC/IEEE International Standard - Systems and software engineering–Vocabulary", *ISO/IEC/IEEE 24765:2017(E)*, 2017. doi: 10.1109/IEEESTD.2017.8016712.

[43] DAVIS, F. D. "Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology", *MIS Quarterly*, v. 13, n. 3, pp. 319–340, 1989. ISSN: 02767783, 21629730. doi: 10.2307/249008.

Appendix A

Evaluated Process Models

A.1 Customer Feedback Repository

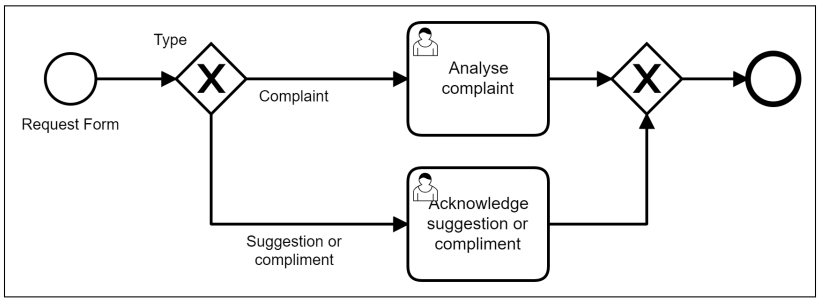


Figure A.1: Process model of the Customer Feedback Process with all 3 types of feedback and exclusive gateway without escalation (`CF-exclusive-all-types-no-scalation` in the Evaluation).

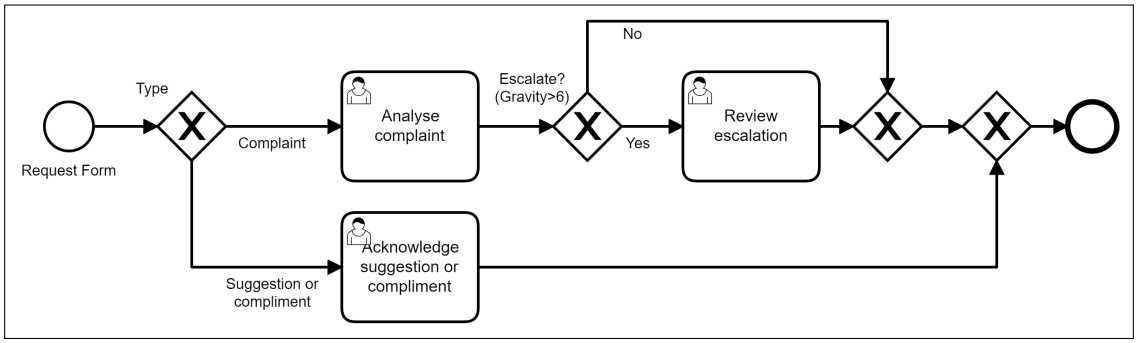


Figure A.2: Process model of the Customer Feedback Process with all 3 types of feedback and exclusive gateway with escalation (`CF-exclusive-all-types-with-scalation` in the Evaluation).

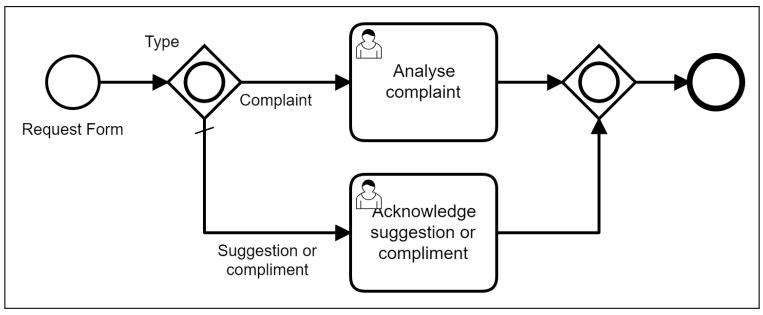


Figure A.3: Process model of the Customer Feedback Process with all 3 types of feedback and inclusive gateway without escalation (in the Evaluation).

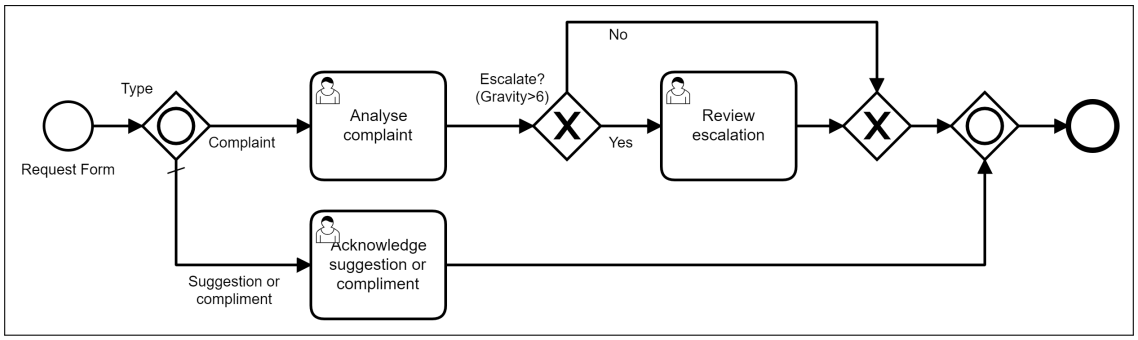


Figure A.4: Process model of the Customer Feedback Process with all 3 types of feedback and inclusive gateway with escalation (`CF-inclusive-all-types-with-scalation` in the Evaluation).

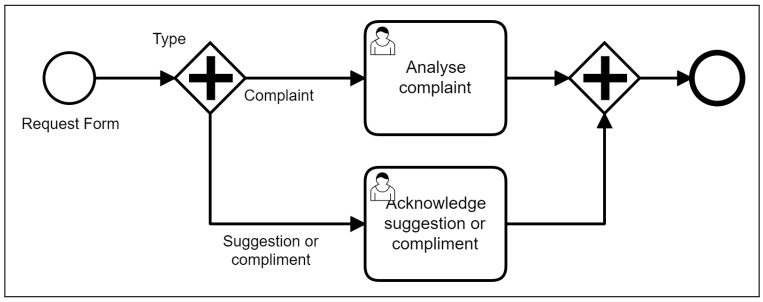


Figure A.5: Process model of the Customer Feedback Process with all 3 types of feedback and parallel gateway without escalation (`CF-parallel-all-types-no-scalation` in the Evaluation).

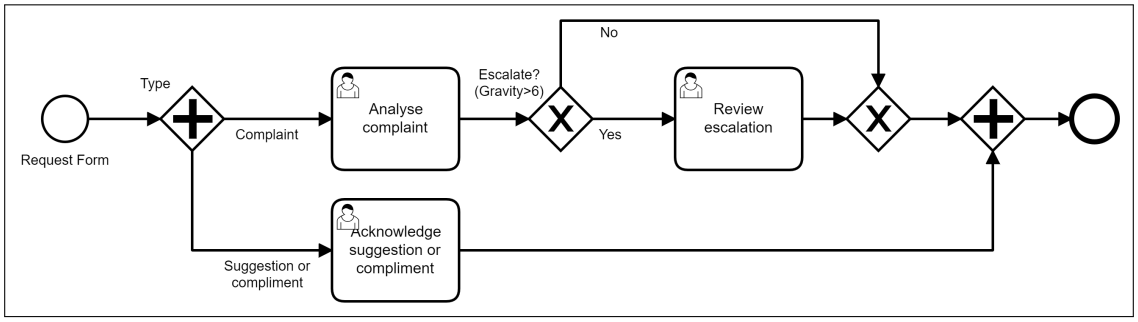


Figure A.6: Process model of the Customer Feedback Process with all 3 types of feedback and parallel gateway with escalation (`CF-parallel-all-types-with-scalation` in the Evaluation).

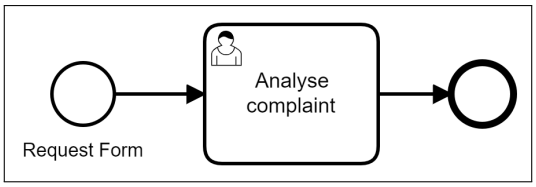


Figure A.7: Process model of the Customer Feedback Process with only complaint as feedback and without escalation (`CF-only-complaint-no-scalation` in the Evaluation).

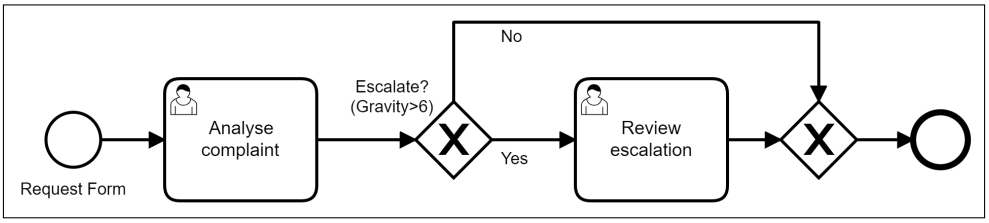


Figure A.8: Process model of the Customer Feedback Process with only complaint as feedback and with escalation (`CF-only-complaint-with-scalation` in the Evaluation).

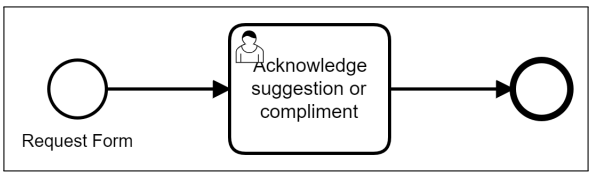


Figure A.9: Process model of the Customer Feedback Process with only suggestion or compliment as feedback and without escalation (`CF-only-suggestion-compliment` in the Evaluation).

A.2 AgileKIP's Travel Plan Tutorial Repository

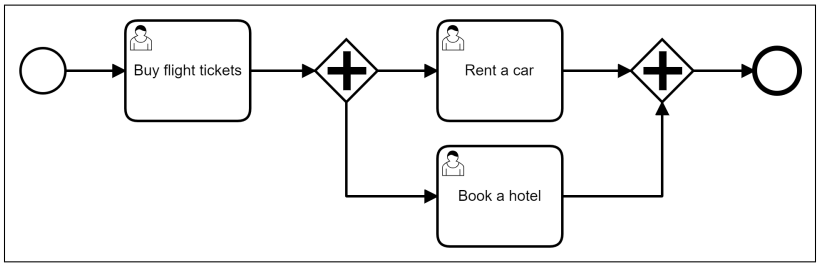


Figure A.10: Process model of the Travel Plan Process with parallel gateway (`TP-AND` in the Evaluation).

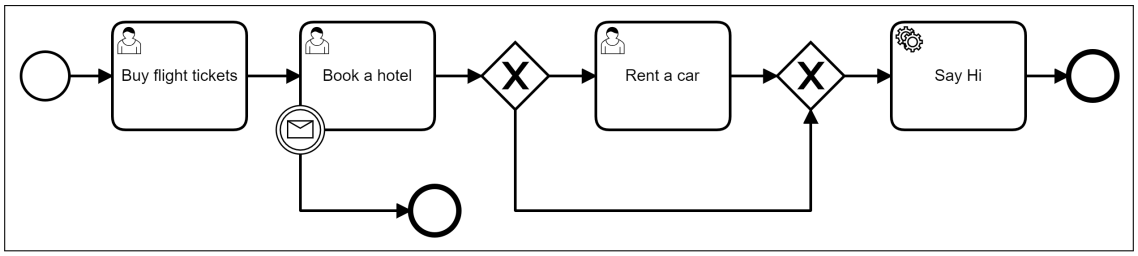


Figure A.11: Process model of the Travel Plan Process with a boundary event message (`TP-EMSG` in the Evaluation).

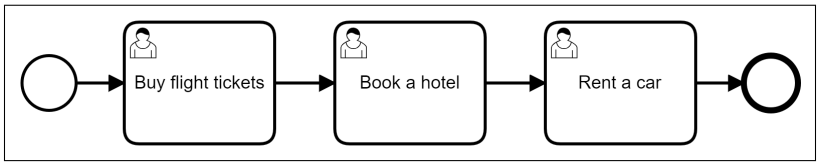


Figure A.12: Process model of the Travel Plan Process with version 1 of entities (`TP-ENTITIES` in the Evaluation).

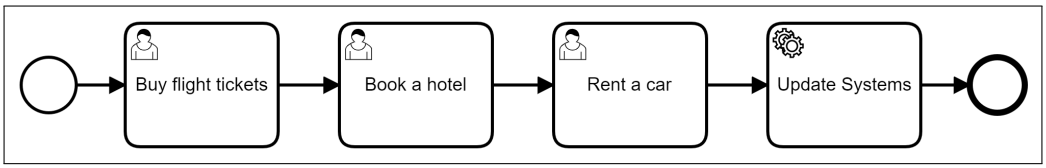


Figure A.13: Process model of the Travel Plan Process with version 2 of entities (`TP-ENTITIES2` in the Evaluation).

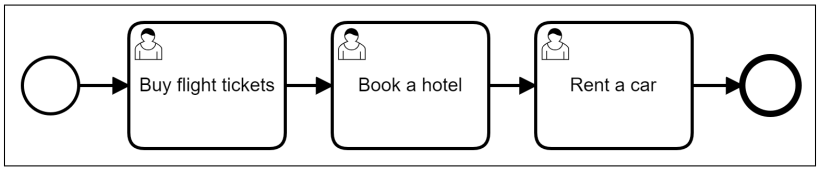


Figure A.14: Process model of the Travel Plan Process with version 3 of entities (`TP-ENTITIES3` in the Evaluation).

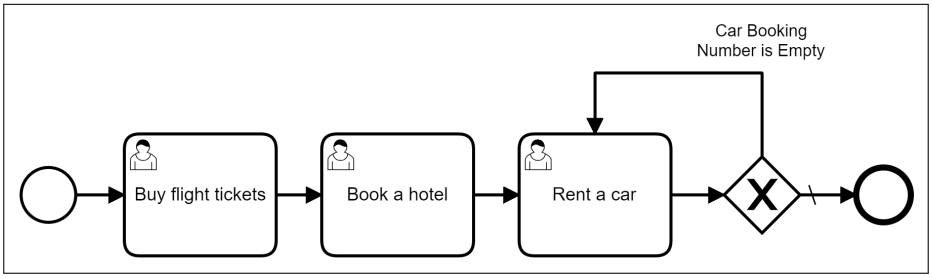


Figure A.15: Process model of the Travel Plan Process with a loop (`TP-LOOP` in the Evaluation).

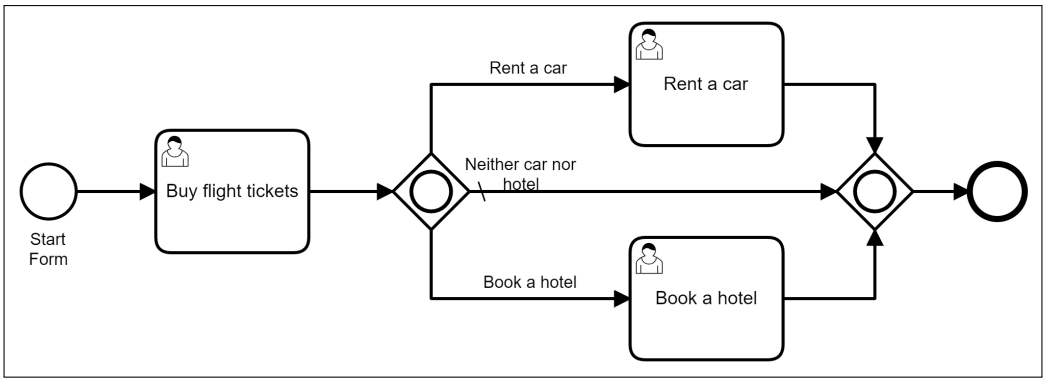


Figure A.16: Process model of the Travel Plan Process with an inclusive gateway (`TP-OR` in the Evaluation).

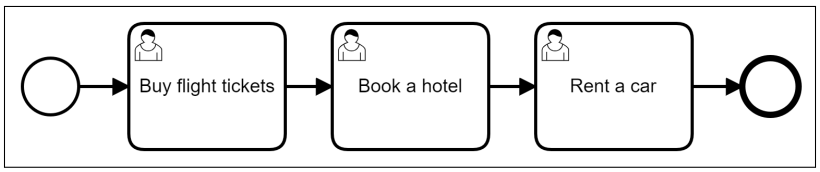


Figure A.17: Process model of the Travel Plan Process without gateways and a simple domain (`TP-SIMPLE` in the Evaluation).

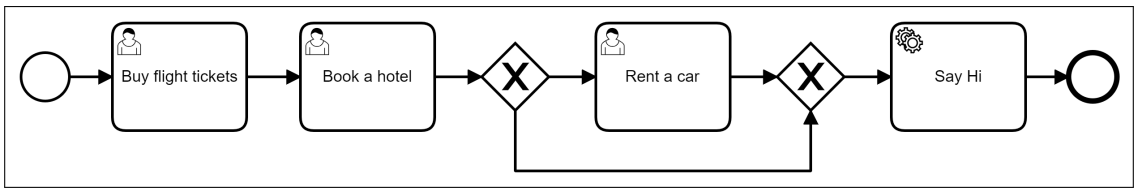


Figure A.18: Process model of the Travel Plan Process with a service task (`TP-SRV` in the Evaluation).

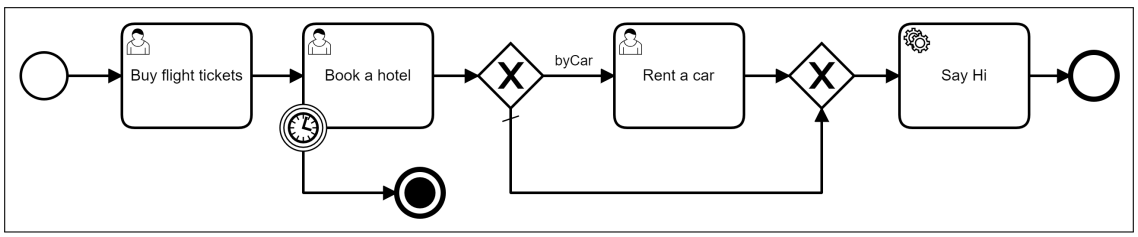


Figure A.19: Process model of the Travel Plan Process with a timer boundary event (`TP-TIMER` in the Evaluation).

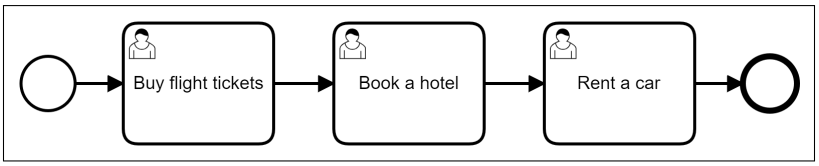


Figure A.20: Process model of the Travel Plan Process with a simple validation in the domain (`TP-VAL` in the Evaluation).

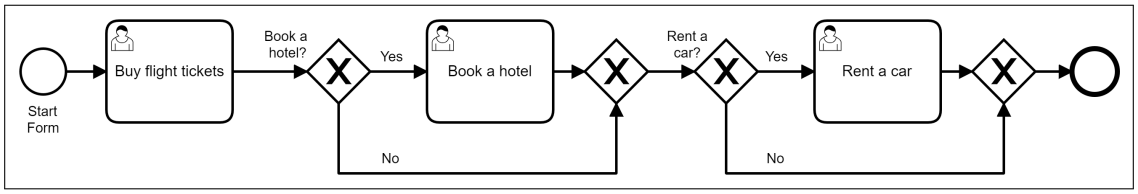


Figure A.21: Process model of the Travel Plan Process with exclusive gateways (`TP-XOR` in the Evaluation).

Appendix B

Survey's Questionnaire

Questionnaire AATPAIS

The present questionnaire has the goal of rating AATPAIS's usefulness and ease-of-use.
It consists of **13 questions using the Likert scale**.

Completing the following sections should take **around 15min** of your time in total:

1. Demographic Information: 1~2min
2. AATPAIS Explanation and Tutorial video: 9min
3. Perceived Usefulness: 2~3min
4. Perceived Ease-of-Use: 2~3min

Informed Consent Form
I declare that I am over 18 years old and agree to willingly and voluntarily participate in an academic research study conducted by Tales Mello Paiva, under the supervision of Professors Toacy Cavalcante de Oliveira and Raquel Mainardi Pillat Basso, for the Master's degree in the Systems and Computer Engineering Program at the Universidade Federal do Rio de Janeiro, Brazil (PESC/COPPE/UFRJ).

1. Objective
The objective of the research is to evaluate AATPAIS, a proposal for the Automated Acceptance Testing of a Process-Aware Information System.

2. Procedure
Participants are required to individually respond to a questionnaire containing professional data and information about usefulness and ease-of-use of the proposed solution.

3. Confidentiality
All information collected in this research is confidential.
The participant's name and email address will not be disclosed.
Likewise, I commit not to disclose my results until the research is concluded.

4. Freedom to Withdraw
I understand that, upon completion of the research, the information I provide will be studied to better understand the desirable characteristics for the automated testing of this type of system.
I understand that I am free to ask questions at any time, request that any information related to me not be included in the study, or communicate my withdrawal from participation in the research.

* Indicates required question

1. Consent Record *

Mark only one oval.

- ◯ I have read and agree to the terms above.
- ◯ I don't agree to the terms above.
  Skip to section 6 (Thank you for your participation!)

Demographic Information

Please answer the following **9 questions** in order to help us better understand the profile of our participants.

2. [1/9] What is your preferred email address? (This information won't be disclosed) *

_____

3. [2/9] What is your age? *

Mark only one oval.

- ◯ Under 18 years old
- ◯ 18-24 years old
- ◯ 25-34 years old
- ◯ 35-44 years old
- ◯ 45-54 years old
- ◯ Over 55 years old

4. [3/9] What is your gender? *

Mark only one oval.

- ◯ Male
- ◯ Female
- ◯ Other: _____

5. [4/9] What is your highest academic degree? *

Mark only one oval.

- ◯ High School Diploma
- ◯ Ongoing Bachelor's Degree
- ◯ Completed Bachelor's Degree
- ◯ Ongoing Master's Degree
- ◯ Completed Master's Degree
- ◯ Ongoing Doctoral Degree (Ph.D., D.Sc., etc.)
- ◯ Completed Doctoral Degree (Ph.D., D.Sc., etc.)
- ◯ Other: _____

6. [5/9] Which sector do you primarily work in? *

Mark only one oval.

- ◯ Industry
- ◯ Academia
- ◯ Both
- ◯ Other: _____

7. [6/9] What is your primary role in your organization? *

Mark only one oval.

- ◯ Software Developer
- ◯ QA Engineer/Test Engineer
- ◯ IT Manager/Project Manager
- ◯ Academic/Researcher
- ◯ Other: _____

8. [7/9] Do you work with IT? How many years of experience with IT do you have? *

Mark only one oval.

- ◯ 0 (Never worked with IT)
- ◯ Less than 1 year
- ◯ 1 to 3 years
- ◯ 3 to 5 years
- ◯ 5 to 8 years
- ◯ More than 8 years

9. [8/9] Have you used a Process-Aware Information System (PAIS) before? Other names for a PAIS are Process-Driven Application (PDA) and Workflow Management System (WfMS). *

Mark only one oval.

- ◯ Yes
- ◯ No
- ◯ Not sure

10. [9/9] Have you used a Robotic Process Automation (RPA) tool before? *

Mark only one oval.

- Yes
- No
- Not sure

AATPAIS Explanation and Tutorial

The main artifacts for the present tutorial and survey are made available at:
https://github.com/talesmp/AATPAIS-Survey

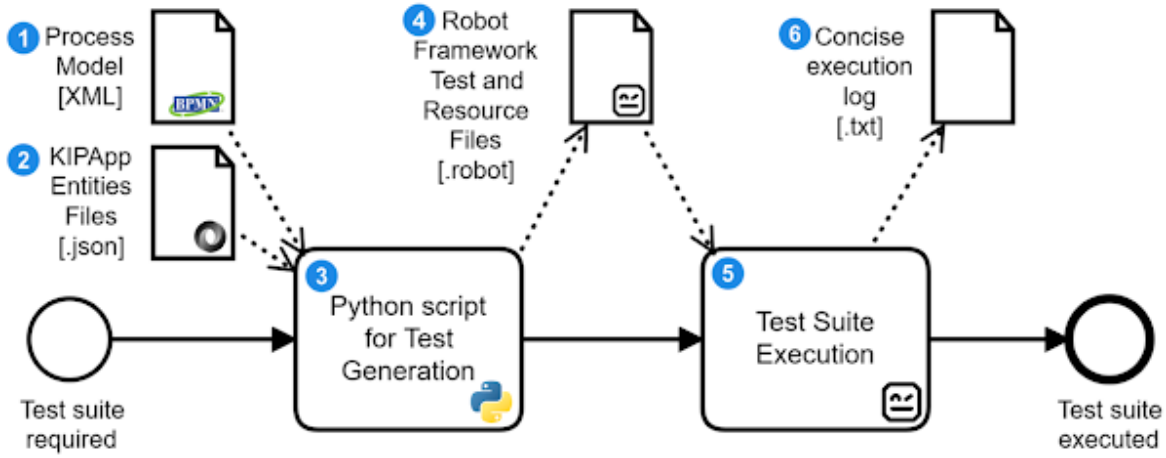
A video with the explanation and tutorial is available below.
If you prefer to follow a YouTube link, you can watch it here:
https://youtu.be/uWQ8DSWheXc

Tutorial Video

To watch it in **fullscreen**, please click on the **YouTube icon** and watch it on YouTube.
https://youtu.be/uWQ8DSWheXc

http://youtube.com/watch?v=uWQ8DSWheXc

The AATPAIS schema



The Perceived Usefulness and Ease-of-Use Questionnaire

The following questionnaire is inspired on the *Technology Acceptance Model (TAM)* proposed by Fred D. Davis (1989), and is designed to give you an opportunity to rate this solution's usefulness and ease-of-use.
It also draws inspiration from the evaluation strategy presented in "*Automated Regression Tests: A No-Code Approach for BPMN-based Process-Driven Applications*" (Schneid et al, 2021).

It contains **13 questions** altogether.
To as great an extent as possible, think about all the tasks that you would do with the product while you answer these questions.

Please read the statements carefully, but don't spend a lot of time on each item.
Your first impression is fine. Don't overthink it.

Category: Perceived Usefulness

Note that for this questionnaire, **all items have a positive tone**, so greater levels of agreement (to the **right of the scale**) indicate a **better user experience**.

11. [1/13] I found AATPAIS helpful for the creation and specification of executable test cases. *

Mark only one oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

12. [2/13] Using AATPAIS would enable testing to be accomplished more quickly. *

Mark only one oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

13. [3/13] Using AATPAIS would increase the productivity. *

Mark only one oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

14. [4/13] AATPAIS increases my motivation and willingness to create test cases for process models. *

Mark only one oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

15. [5/13] Using AATPAIS would improve testing performance. *

Mark only one oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

16. [6/13] Using AATPAIS would make it easier to do my job. *

Mark only one oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

17. [7/13] I would find AATPAIS useful in my job. *

Mark only one oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

Category: Perceived Ease-of-Use

Note that for this questionnaire, **all items have a positive tone**, so greater levels of agreement (to the **right of the scale**) indicate a **better user experience**.

18. [8/13] I would find it easy to get AATPAIS to do what I want it to do. *
(e.g. chosen strings for fields Name and Airline Company).

Mark only one oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

19. [9/13] Learning to operate AATPAIS would be easy for me. *

Mark only one oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

20. [10/13] The test cases seemed to be clearly organized to me. *

Mark only one oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

21. [11/13] My interaction with AATPAIS would be clear and understandable. *

Mark only one oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

22. [12/13] I would find AATPAIS easy to use. *

Mark only one oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

23. [13/13] It would be easy for me to become skillful at using AATPAIS. *

Mark only one oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

24.

You may already have **ideas and suggestions** for improving AATPAIS.
Please indicate them below:

Thank you for your participation!

Please, don't forget to click on the "**Submit**" (or "**Enviar**") button bellow to confirm your participation.

Thank you for taking the time to participate in our academic survey.
We deeply appreciate your willingness to share your perspectives.

Your valuable insights and contributions are crucial to the success of our research, and we are grateful for your support.

Thank you once again for your participation and for contributing to the advancement of knowledge in Automated Software Testing and Quality Assurance.

Sincerely,
Tales Mello Paiva