UMA FERRAMENTA WEB VISUAL PARA MODELAGEM ESTRUTURADA DE CENÁRIOS DE CASO DE USO ORIENTADA À INTEROPERABILIDADE EM AMBIENTES DE ARQUITETURA ORIENTADA A MODELOS

Gabriel Gonçalves Costa, Eduarda M. De Souza, Maria Sophia Pardinho Da Silva, Pedro L. M. Fortunato, Flávio Henrique de Oliveira.

RESUMO

A Engenharia de Requisitos desempenha papel central na definição de sistemas de software, e os casos de uso, juntamente com a descrição de seus cenários, destacam-se como técnica consolidada para descrever o comportamento do sistema sob a perspectiva das interações entre atores e sistema. Contudo, observa-se que as principais ferramentas UML open-source oferecem suporte principalmente à diagramação, carecendo de mecanismos integrados e estruturados para especificação de cenários (fluxos principal, alternativos e de exceção) e de recursos de interoperabilidade em formatos padronizados. Este trabalho apresenta uma ferramenta web que integra a modelagem gráfica de Casos de Uso à especificação detalhada de seus cenários, permitindo a edição visual estruturada e a exportação/importação de todo o modelo em um esquema XML bem definido. A pesquisa seguiu uma abordagem exploratória e incremental, envolvendo análise comparativa de ferramentas existentes, definição de requisitos mínimos e prototipação iterativa do editor visual e do esquema XML, validados por meio de testes práticos. Como principal contribuição, a ferramenta propicia uma representação formal e interoperável dos requisitos comportamentais, favorecendo sua utilização como entrada em processos de desenvolvimento orientado a modelos. Argumenta-se que essa abordagem reduz ambiguidades, fortalece a rastreabilidade e aumenta o potencial de automação no ciclo de vida do desenvolvimento de software.

Palavras-chave: Caso de uso; Engenharia de Software; Cenários de caso de uso;

A Visual Web Tool for Structured Modeling of Use Case Scenarios Oriented to Interoperability in Model-Driven Architecture Environments.

Gabriel Gonçalves Costa, Eduarda M. De Souza, Maria Sophia Pardinho Da Silva, Pedro L. M. Fortunato, Flávio Henrique de Oliveira.

ABSTRACT

Requirements Engineering plays a central role in the definition of software systems, and use cases, together with the description of their scenarios, stand out as a consolidated technique for describing system behavior from the perspective of interactions between actors and the system. However, the main open-source UML tools are observed to primarily support diagramming, lacking integrated and structured mechanisms for scenario specification (main, alternative, and exception flows) and for interoperability features based on standardized formats. This paper presents a web-based tool that integrates the graphical modeling of use cases with the detailed specification of their scenarios, enabling structured visual editing and the export/import of the entire model into a well-defined XML schema. The research followed an exploratory and incremental approach, involving a comparative analysis of existing tools, the definition of minimal requirements, and the iterative prototyping of both the visual editor and the XML schema, which were validated through practical tests. As its main contribution, the tool provides a formal and interoperable representation of behavioral requirements, fostering its use as input to model-driven development processes. It is argued that this approach reduces ambiguities, strengthens traceability, and increases the potential for automation in the software development lifecycle.

Keywords: Use Cases; Software Engineering; Use Case Scenarios.

1. INTRODUÇÃO

A Engenharia de Requisitos (ER) é uma disciplina fundamental no ciclo de vida do desenvolvimento de software, estabelecendo a ponte entre as necessidades das partes interessadas e a especificação técnica do sistema. Dentre as técnicas mais consolidadas para a elicitação e descrição de requisitos funcionais, os Casos de Uso (*Use Cases*) destacam-se por sua eficácia em descrever a interação entre atores e o sistema sob uma perspectiva comportamental (JACOBSON et al., 1992). Contudo, a eficácia dos Casos de Uso não reside apenas em seus diagramas, mas, principalmente, na descrição detalhada de seus cenários com fluxo principal de sucesso e os fluxos alternativos e de exceção que capturam a complexidade das regras de negócio.

Paralelamente, a Engenharia Orientada a Modelos (*Model-Driven Engineering* - MDE), e sua principal implementação, a Arquitetura Orientada a Modelos (Model-Driven Architecture - MDA), têm ganhado proeminência (OMG, 2003). O MDA propõe o desenvolvimento de software através da criação e transformação sucessiva de modelos, desde Modelos Independentes de Plataforma (PIM), que focam na lógica de negócio, até Modelos Específicos de Plataforma (PSM). Para que essa abordagem atinja seu potencial máximo de automação e consistência, é imperativo que os modelos de entrada, como os Casos de Uso, sejam formais e estruturados.

Buscando solucionar esta carência, foi realizada uma pesquisa exploratória das principais ferramentas de modelagem UML *open-source* disponíveis ao público. A análise revelou que, embora a maioria ofereça suporte robusto à diagramação de Casos de Uso, elas falham em prover um mecanismo integrado, visual e intuitivo para a especificação estruturada de seus cenários (fluxo principal e alternativos). Mais importante, verificou-se a ausência de ferramentas que combinem essa facilidade de modelagem com a capacidade de exportar e importar essa estrutura de cenários em um formato de dados padronizado e interoperável, como o XML.

Para endereçar essa problemática, este artigo apresenta uma ferramenta web, projetada especificamente para preencher essa lacuna. A ferramenta oferece um

ambiente visual integrado que permite aos analistas de requisitos não apenas desenhar diagramas de caso de uso, mas também especificar detalhadamente cada etapa dos fluxos principal e alternativos de forma estruturada. A contribuição central da ferramenta é sua capacidade de exportar (e importar) o modelo completo, diagramas e cenários, para um esquema XML bem definido. Este artefato XML é projetado para servir como entrada direta para ferramentas subsequentes no processo de desenvolvimento, especialmente motores de transformação MDA que necessitam de uma representação formal dos requisitos comportamentais.

Para o desenvolvimento da ferramenta, adotou-se uma metodologia incremental e exploratória, iniciando pela análise comparativa das ferramentas open-source existentes e pela identificação de lacunas relacionadas à modelagem estruturada de cenários e à ausência de formatos padronizados de interoperabilidade. Com base nesses achados, definiu-se um conjunto mínimo de requisitos, priorizando simplicidade de uso, aderência aos conceitos da UML e independência de plataforma.

A implementação seguiu um processo iterativo de prototipação, no qual versões sucessivas do editor visual e do esquema XML foram refinadas por meio de testes práticos. Essa estratégia permitiu validar tanto a ergonomia do ambiente quanto a consistência da estrutura exportada, resultando em uma solução capaz de integrar, de forma clara e formal, a modelagem gráfica de casos de uso e a especificação detalhada de seus cenários.

Este trabalho detalha a arquitetura da ferramenta e o esquema XML proposto para interoperabilidade e demonstra sua aplicação prática. Argumenta-se que, ao formalizar a especificação de cenários de forma visual e garantir sua exportabilidade, a ferramenta facilita a integração entre a Engenharia de Requisitos e as abordagens de desenvolvimento orientado a modelos, reduzindo a ambiguidade e aumentando o potencial de automação no ciclo de vida do software.

2. OBJETIVO GERAL

Desenvolver uma ferramenta web visual que integre a modelagem de diagramas de Caso de Uso à especificação estruturada de seus cenários, permitindo a exportação e importação de modelos em formato XML para promover a interoperabilidade em ambientes de Arquitetura Orientada a Modelos (MDA) e Inteligência Artificial.

3. OBJETIVOS ESPECÍFICOS

- Analisar as limitações de ferramentas UML open-source atuais quanto ao suporte integrado para a especificação textual de fluxos (principal, alternativos e de exceção).
- Projetar e implementar um editor visual com suporte a recursos para a criação de atores e casos de uso, integrado a um módulo de edição estruturada de cenários com fluxo principais e alternativos.
- Desenvolver um esquema XML bem definido para representar formalmente a estrutura completa do modelo (diagramas e cenários), garantindo a persistência e a rastreabilidade dos requisitos.
- Validar a ferramenta por meio de testes práticos, verificando a integridade dos arquivos XML gerados, a capacidade de reimportação de dados e a usabilidade da interface.

4. JUSTIFICATIVA

Como justificativa para este trabalho, foi realizado um levantamento das principais ferramentas open source de modelagem UML voltadas à representação de casos de uso e à descrição de cenários de caso de uso. Para isso, conduziu-se uma análise comparativa das três ferramentas mais representativas desse ecossistema, identificando-se, em cada uma, os recursos disponíveis para diagramação e documentação, bem como suas limitações no suporte à especificação textual e ao refinamento dos fluxos (principal, alternativos e de exceção) associados aos casos de

uso. Essa investigação fundamenta a necessidade de discutir lacunas práticas entre a modelagem visual e a formalização de cenários, motivando a proposta desenvolvida ao longo deste capítulo.

A comparação entre Modelio, Eclipse Papyrus e Gaphor evidencia um padrão recorrente nas ferramentas open source de modelagem UML: há maturidade na diagramação de Casos de Uso, porém permanece uma lacuna relevante na especificação estruturada de cenários (fluxo principal, alternativos e exceções) de forma integrada, tratável e interoperável, justamente o nível de detalhe necessário para reduzir ambiguidade e alimentar abordagens como MDA/MDE.

No caso do Modelio, observa-se um ambiente de modelagem amplo e voltado a padrões, com suporte consistente à criação de diagramas UML e mecanismos formais de interoperabilidade. O próprio ecossistema do Modelio disponibiliza importação/exportação XMI para troca de modelos UML2 entre ferramentas, incluindo suporte a múltiplas versões do padrão OMG. Apesar disso, o mesmo material de referência explicita restrições: a cobertura de XMI não abrange todo o padrão UML2, o que por si só já cria fricções quando se busca intercâmbio completo de informações.

Além disso, embora o Modelio possua recursos e módulos para geração de documentação a partir do modelo (com templates e editor de texto rico), essa documentação tende a operar no nível de texto descritivo associado aos elementos, não como uma estrutura formal de passos/condições/ramificações dos cenários.

Na prática, isso significa que o diagrama de Caso de Uso é bem atendido, mas o cenário (fluxo principal e variações) permanece frequentemente como texto livre ou documentação paralela, com baixa capacidade de reutilização automática, geração sistemática de modelos comportamentais, validação de consistência e transformação orientada a modelos.

O Eclipse Papyrus, por sua vez, é apresentado como uma ferramenta *open source* de nível industrial para engenharia baseada em modelos, com forte alinhamento ao ecossistema Eclipse e às práticas de MBE/MDE.

Em guias e tutoriais, o Papyrus explicita que Casos de Uso tipicamente são descritos como sequência de passos, podendo ser ilustrados por diagramas de sequência e descritos textualmente.

O problema é que o suporte textual oferecido tende a ser genérico: o Papyrus fornece uma *Documentation View* para anexar texto aos elementos do modelo, o que é útil, mas não equivale a um editor estruturado de cenários com campos e controles para fluxo principal, condições, extensões, exceções e rastreabilidade.

Já o Gaphor se posiciona como uma ferramenta UML/SysML, destacando que implementa um modelo de dados UML 2 plenamente compatível, não sendo apenas um editor de desenhos.

Contudo, a própria filosofia declarada do Gaphor toda informação colocada no modelo é visível nos diagramas, sem painéis ocultos e sem páginas extensas de propriedades, favorece uma experiência leve e visual, mas não prioriza a captura de cenários textuais estruturados como um artefato de requisitos com passos, ramificações e exceções gerenciáveis.

Assim, apesar de ser muito adequado para criar e comunicar diagramas UML, tende a empurrar a descrição detalhada de cenários para fora da ferramenta (documentos e descrições livres), reduzindo a capacidade de manter o diagrama e os cenários sincronizados e, principalmente, de exportar essa informação em um formato semântico apropriado para automação posterior.

Essa combinação de fatores fundamenta a justificativa para uma ferramenta (ou módulo) específica que permita modelar Casos de Uso e, no mesmo ambiente, editar cenários de forma estruturada (fluxo principal, alternativos e exceções) e exportar/importar essa estrutura em um XML padronizado (ou esquema formal equivalente), apto a alimentar processos MDA/MDE, fortalecer rastreabilidade e reduzir ambiguidades em ER.

5. FUNDAMENTAÇÃO TEÓRICA

Na literatura de Engenharia de Software, casos de uso consolidaram-se como uma forma pragmática de capturar e comunicar requisitos funcionais a partir da perspectiva do usuário/ator, organizando o comportamento esperado do sistema em torno de objetivos e de interações observáveis. Ao longo do tempo, essa abordagem foi incorporada de maneira ampla em processos de análise e projeto, tanto como mecanismo de entendimento do domínio quanto como insumo para decisões de design, rastreabilidade e validação com stakeholders.

Na Engenharia de Software, casos de uso são empregados para representar, de forma orientada a objetivos, como um sistema se comporta quando interage com atores externos, descrevendo sequências de ações que podem incluir variações (por exemplo, caminhos alternativos e situações excepcionais) e que culminam em um resultado observável com valor para o ator (JACOBSON; BOOCH; RUMBAUGH, 1999). Sob essa perspectiva, um caso de uso não se limita a um único roteiro: ele pode ser entendido como uma descrição abrangente das possíveis sequências de interação entre o sistema e um ou mais atores, disparadas por um estímulo inicial proveniente de um desses atores, permitindo explicitar diferentes comportamentos esperados conforme condições e escolhas ao longo da execução (RUMBAUGH, 1994). Complementarmente, a literatura enfatiza que o caso de uso atua como um artefato agregador de cenários, reunindo múltiplas possibilidades de interação entre o sistema em análise e seus atores externos, sempre relacionadas a um objetivo específico, o que favorece a comunicação entre stakeholders e a delimitação do escopo funcional a ser atendido (COCKBURN, 2000).

De acordo com Jacobson et al. (1992), os casos de uso representam a interação entre um ator e o sistema para atingir um objetivo específico, descrevendo um conjunto de sequências de ações que produzem um resultado observável.

Um cenário de caso de uso é composto principalmente por um fluxo principal (cenário de sucesso), que descreve a sequência básica de passos correspondente ao caminho ideal para o atingimento do objetivo, por fluxos alternativos, que representam variações do comportamento previsto no fluxo principal e ainda assim

conduzem a um resultado válido, e por fluxos de exceção, que caracterizam desvios destinados ao tratamento de erros, condições inválidas ou eventos inesperados que podem ocorrer durante a interação entre ator e sistema.

Na literatura de Engenharia de Software, casos de uso consolidaram-se como uma forma pragmática de capturar e comunicar requisitos funcionais a partir da perspectiva do usuário/ator, organizando o comportamento esperado do sistema em torno de objetivos e de interações observáveis (JACOBSON, 2004; RATCLIFFE; BUDGEN, 2005). Ao longo do tempo, essa abordagem foi incorporada de maneira ampla em processos de análise e projeto, servindo tanto como mecanismo de entendimento do domínio quanto como insumo para decisões de design e para apoiar rastreabilidade e validação com stakeholders (RATCLIFFE; BUDGEN, 2005; FIRESMITH, 1999).

Em paralelo, a noção de cenários ganhou força como técnica de elicitação e validação por privilegiar descrições concretas de uso (situações, sequência de ações/eventos, contexto e exceções), favorecendo entendimento compartilhado e detecção precoce de lacunas (POTTS; TAKAHASHI; ANTÓN, 1994; HSIA et al., 1994). Estudos empíricos em ambiente industrial indicam que o uso de cenários é altamente diverso em forma, conteúdo, propósito e ciclo de vida, frequentemente excedendo prescrições metodológicas e impondo desafios de estruturação e gerenciamento desses artefatos (WEIDENHAUPT et al., 1998).

Nesse sentido, propostas de classificação buscam reduzir ambiguidades conceituais e permitir que cenários sejam comparados e utilizados de modo mais sistemático, destacando dimensões como forma, conteúdo, propósito e evolução ao longo do projeto (ROLLAND et al., 1998).

A relação entre os dois conceitos é usualmente tratada de forma complementar: cenários podem representar caminhos (básico, alternativos e exceções) dentro de um caso de uso, apoiando a elaboração incremental de requisitos e a verificação de consistência e completude (SUTCLIFFE et al., 1998). Diretrizes de modelagem e estratégias de inspeção aparecem como resposta a problemas recorrentes de qualidade em especificações baseadas em casos de uso,

reforçando a necessidade de padronização e revisão sistemática desses artefatos (FIRESMITH, 1999; THELIN; RUNESON; WOHLIN, 2003).

6. METODOLOGIA

O presente estudo caracteriza-se como uma pesquisa aplicada, de natureza exploratória e tecnológica, orientada ao desenvolvimento de um artefato de software voltado à modelagem de casos de uso e à especificação estruturada de cenários. Para conduzir o desenvolvimento, adotou-se uma abordagem incremental e iterativa de prototipação, organizada em etapas sucessivas de levantamento de requisitos, definição arquitetural, implementação e validação.

Na etapa de análise de requisitos, foram identificadas e priorizadas as funcionalidades essenciais para apoiar a unificação entre diagrama e cenário, contemplando a criação de atores, casos de uso e relacionamentos, bem como a documentação dos fluxos (principal, alternativos e de exceção) e a necessidade de uma representação exportável e reutilizável do projeto.

Na etapa de modelagem e arquitetura, definiu-se uma arquitetura web orientada ao front-end, implementada com React e organizada sobre o ecossistema Vite + TypeScript. O gerenciamento de estado da aplicação foi estruturado com Zustand, enquanto a persistência dos dados do projeto foi realizada por meio do LocalStorage, permitindo armazenamento local e recuperação do modelo sem dependência de serviços externos.

Em seguida, na etapa de implementação, o editor visual foi desenvolvido com suporte à manipulação de nós e conexões para representar os elementos do diagrama, integrando, no mesmo ambiente, a edição estruturada dos cenários associados aos casos de uso. Paralelamente, foi projetado um mecanismo de exportação/importação em XML, por meio da definição de um esquema capaz de representar a estrutura do modelo (atores, casos de uso, relacionamentos e fluxos), visando preservar rastreabilidade e permitir interoperabilidade com processos orientados a modelos (por exemplo, transformações entre níveis CIM/PIM).

Por fim, a etapa de validação foi conduzida por testes práticos e verificações de consistência do artefato, abrangendo: (i) a correta construção e edição dos elementos do diagrama e dos cenários; (ii) a integridade do arquivo XML gerado e sua capacidade de reimportação sem perda de informações; e (iii) a clareza do fluxo de interação na interface, por meio de inspeções e testes manuais orientados a cenários de uso. Adicionalmente, foram empregadas práticas de engenharia de software, como versionamento e organização modular do código, para aumentar a reprodutibilidade do desenvolvimento e facilitar a evolução do protótipo.

7. APRESENTAÇÃO E DISCUSSÃO DOS RESULTADOS

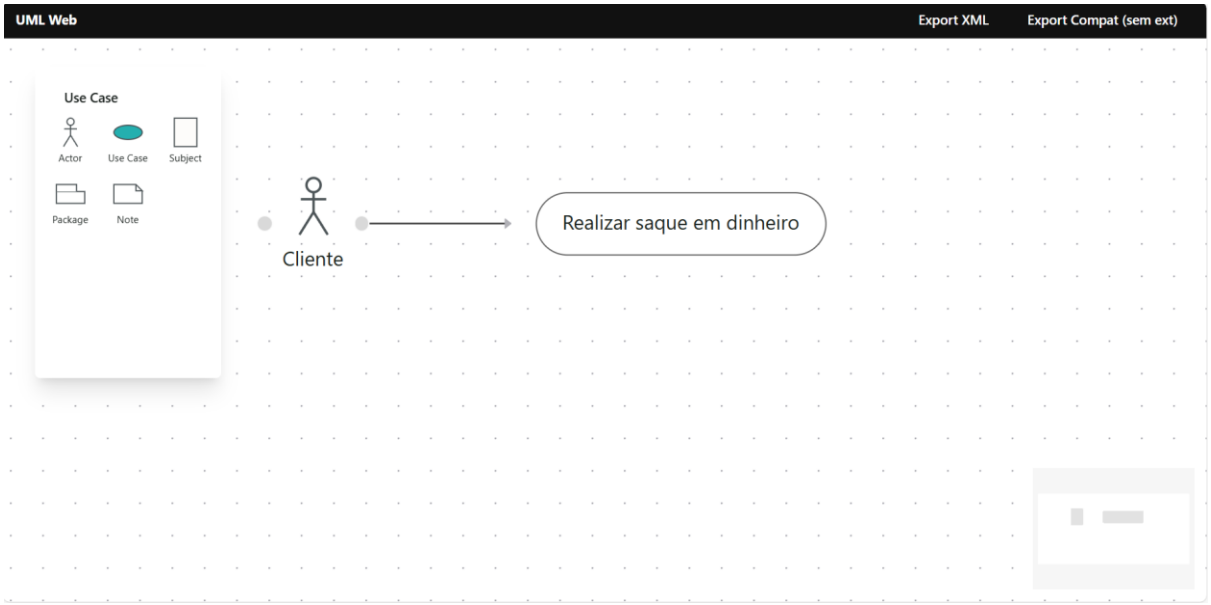
O código-fonte e a documentação técnica da ferramenta estão hospedadas em repositório público (COSTA et al., 2023). A ferramenta disponibiliza uma interface organizada em três módulos integrados. O painel de diagrama é destinado à construção visual dos principais elementos da modelagem, incluindo atores, casos de uso e seus relacionamentos. Complementarmente, o editor de cenários permite a inserção e a edição estruturada dos cenários associados aos casos de uso, contemplando o fluxo principal e os fluxos alternativos. Por fim, o módulo de exportação em XML viabiliza a geração automática de um arquivo no formato .xml, no qual é registrada a estrutura completa do projeto, abrangendo atores, fluxos, etapas/fases e relacionamentos.

A usabilidade da ferramenta foi planejada para facilitar o trabalho do analista. O recurso de *Drag and Drop* permite criar, mover e relacionar atores e casos de uso de forma direta e intuitiva, tornando o processo de modelagem mais ágil e reduzindo a complexidade operacional. A Figura 1 ilustra a interface e sua organização geral.

Os testes realizados indicam que o XML gerado é claro, estruturado e compatível com validação via XSD, podendo ser utilizado como entrada em mecanismos de transformação baseados em MDA. Essa padronização assegura interoperabilidade e consistência entre a modelagem visual e as etapas posteriores do desenvolvimento orientado a modelos.

Observa-se na Figura 1 o exemplo visual de um diagrama de caso de uso ao qual podemos adicioanr diversos atores e casos de uso.

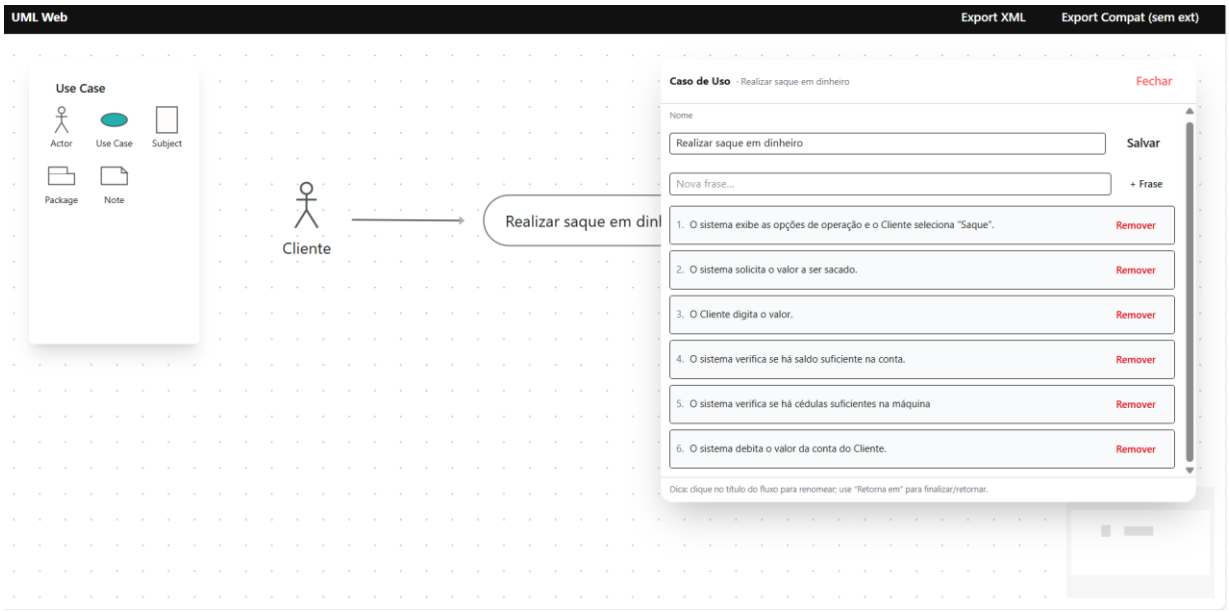
Figura 1 – Exemplo construção diagrama de caso de uso



Fonte: Elaboração Própria.

Observa-se, na Figura 2, o ambiente de modelagem em que a especificação do caso de uso é construída de forma integrada, combinando representação gráfica e detalhamento textual. No diagrama, o ator Cliente é associado ao caso de uso “Realizar saque em dinheiro”, enquanto, no painel de edição à direita, o respectivo cenário é descrito por uma sequência ordenada de passos que explicita a interação usuário–sistema, incluindo a seleção da operação de saque, a solicitação e inserção do valor, bem como as validações de saldo e disponibilidade de cédulas, culminando no débito do montante na conta do cliente. Essa organização evidencia a rastreabilidade entre os elementos do diagrama e o fluxo operacional do caso de uso, favorecendo consistência e precisão na documentação do comportamento esperado.

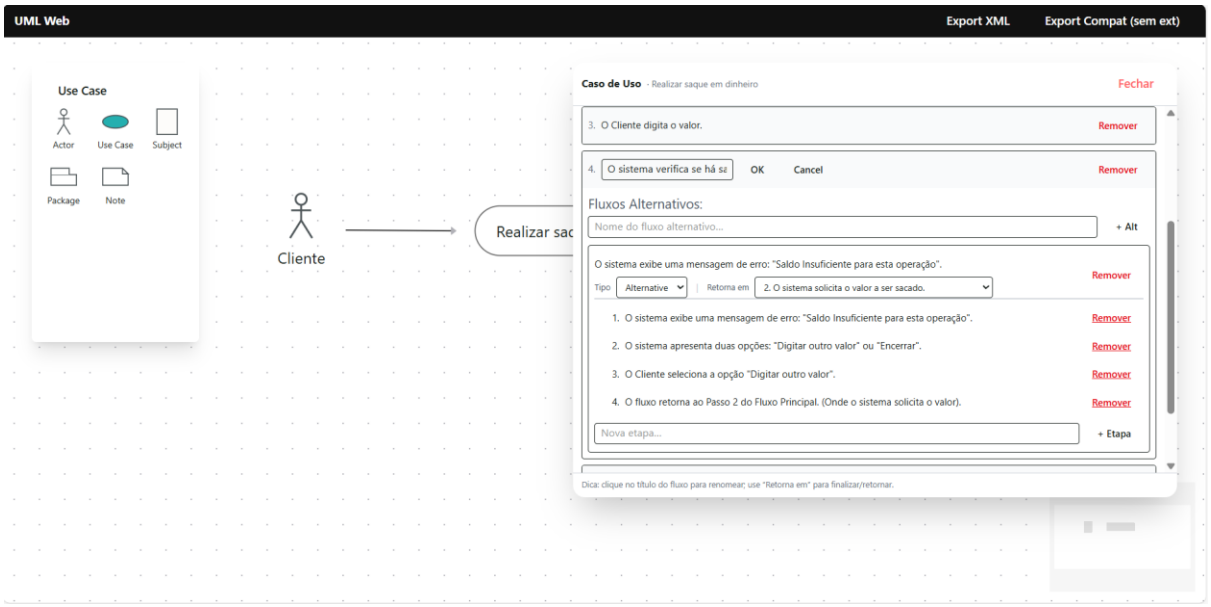
Figura 2 – Exemplo construção fluxo principal cenário de caso de uso



Fonte: Elaborado pelo autor (2025).

Observa-se, na Figura 3, um exemplo de especificação de fluxo alternativo associado ao caso de uso, evidenciando o suporte da ferramenta para a modelagem de variações comportamentais e tratamento de situações não previstas no fluxo principal. No editor, o analista pode classificar o desvio como alternativo ou de exceção, além de registrar a condição que o dispara (por exemplo, saldo insuficiente) e descrever, em passos sequenciais, as ações executadas pelo sistema e pelo ator, incluindo mensagens de erro e opções de continuidade oferecidas ao usuário. Adicionalmente, a interface permite indicar o ponto de retorno ao fluxo principal, reforçando a consistência do cenário e a rastreabilidade entre o desvio especificado e a etapa correspondente no fluxo base.

Figura 3 – Exemplo construção fluxo alternativo cenário de caso de uso



Fonte: Elaborado pelo autor (2025).

Observa-se, no Quadro 1, um exemplo da saída final em XML gerada pela ferramenta, a qual materializa a estrutura do projeto em um formato serializável e padronizado. Essa representação favorece tanto a exportação e importação do modelo entre ambientes distintos quanto a reconstrução e visualização gráfica dos elementos modelados, preservando atores, casos de uso, relacionamentos e fluxos. Além disso, a disponibilização do conteúdo em XML viabiliza a integração com ferramentas externas, incluindo soluções baseadas em IA, para apoiar a extração e transformação do modelo em requisitos estruturados, bem como reforça a interoperabilidade em contextos de Model-Driven Architecture (MDA), ao facilitar a troca de artefatos e a automação de etapas de análise e refinamento.

Quadro 1 – Estrutura XML gerada pela ferramenta para o caso de uso.

```
<?xml version="1.0" encoding="UTF-8"?>
<root xmlns:ext="urn:umlweb:v1">
  <use_case id="e3f16cfe-f645-44a9-8669-25acb0e67144">
    Use Case
    <main_flow>
      <phrase id="62368a85-6fdc-4794-883a-fc0097fabe40">O sistema exibe as opções de
operação e o Cliente seleciona "Saque".</phrase>
      <phrase id="370c3909-cdb6-44db-8210-ac127e98bd1d">O sistema solicita o valor a ser
sacado.</phrase>
      <phrase id="c318a210-9e8b-4422-927c-de77fa72b4d4">O Cliente digita o valor.</phrase>
      <phrase id="4b89595b-3775-4275-816b-4c7a7b884c6a">O sistema verifica se há saldo
suficiente na conta.</phrase>
      <phrase id="68d1b613-4957-48f3-b2e0-329a49ad0a3d">O sistema verifica se há cédulas
suficientes na máquina</phrase>
      <phrase id="bab7293d-3de7-407e-8c62-dc7308a1755a">O sistema debita o valor da conta
do Cliente.</phrase>
    </main_flow>
    <alternative_flow id="a9cc3c15-5664-4c87-84a9-231d9ed2710b"
ext:parent_phrase_id="4b89595b-3775-4275-816b-4c7a7b884c6a" ext:return_phrase_id="370c3909-
cdb6-44db-8210-ac127e98bd1d" ext:kind="alternative">
      O sistema exibe uma mensagem de erro: "Saldo Insuficiente para esta operação".
      <flow id="59da589f-9da5-472c-80c5-5712ce581105">O sistema exibe uma mensagem de erro:
"Saldo Insuficiente para esta operação".</flow>
      <flow id="1289f420-fc76-49fa-9fbf-780bddc77f9e">O sistema apresenta duas opções:
"Digitar outro valor" ou "Encerrar".</flow>
      <flow id="6a870733-05a1-459f-bbb8-cf3f85391e4f">O Cliente seleciona a opção "Digitar
outro valor".</flow>
      <flow id="41e1ee92-9056-4c23-95a6-090867012d9a">O fluxo retorna ao Passo 2 do Fluxo
Principal. (Onde o sistema solicita o valor).</flow>
    </alternative_flow>
  </use_case>
</root>
```

Fonte: Elaboração Própria.

8. CONSIDERAÇÕES FINAIS

Este trabalho apresentou uma ferramenta web que integra, em um mesmo ambiente, a modelagem gráfica de casos de uso e a especificação estruturada de seus cenários, contemplando fluxo principal, alternativos e de exceção, além de permitir a exportação e importação do modelo completo por meio de um esquema XML bem definido. A proposta foi motivada pela lacuna observada em ferramentas UML open source, que em geral priorizam a diagramação, mas não oferecem mecanismos integrados para a descrição estruturada de cenários e para interoperabilidade baseada em formatos padronizados.

Nesse sentido, a principal contribuição reside na disponibilização de uma representação formal e interoperável de requisitos comportamentais, favorecendo rastreabilidade, redução de ambiguidades e maior potencial de automação em processos orientados a modelos (MDA/MDE). A construção da solução seguiu uma abordagem incremental e exploratória, envolvendo análise de requisitos, definição arquitetural e prototipação iterativa do editor visual e do esquema XML, com validação por testes práticos. Os resultados indicam que a estrutura exportada em XML é consistente e pode ser validada por XSD, reforçando a viabilidade de utilização como entrada para etapas posteriores de transformação e integração em pipelines MDA.

Como limitação, destaca-se que a avaliação realizada foi predominantemente prática e manual, o que abre espaço para estudos futuros com protocolos mais sistemáticos de usabilidade e qualidade do modelo gerado, incluindo participação de especialistas e comparação controlada com ferramentas consolidadas.

Como trabalhos futuros, propõe-se a evolução da ferramenta para ampliar o suporte aos relacionamentos include e extend, estabelecendo sua ligação completa no modelo e na exportação/importação, além de estender mecanismos de interoperabilidade e validações semânticas dos cenários como ferramentas de inteligência artificial.

REFERÊNCIAS

COCKBURN, Alistair. Writing effective use cases. Harlow: Addison-Wesley, 2000.

COSTA, Gabriel Gonçalves et al. uml-web: Ferramenta Web Visual para Modelagem Estruturada de Cenários de Caso de Uso. Versão 1.0. [S. l.]: GitHub, 2023. Disponível em: https://github.com/ExiledSpirit/uml-web. Acesso em: 17 dez. 2025.

FIRESMITH, Donald G. Use case modeling guidelines. In: TECHNOLOGY OF OBJECT-ORIENTED LANGUAGES AND SYSTEMS (TOOLS 30), 1999. Proceedings… Los Alamitos: IEEE Computer Society, 1999. p. 184–193. DOI: 10.1109/TOOLS.1999.787548.

HSIA, Pei; SAMUEL, James; GAO, Jian; KUNG, David; TOYOSHIMA, Yukiharu; CHEN, Charles. Formal approach to scenario analysis. IEEE Software, v. 11, n. 2, p. 33–41, 1994. DOI: 10.1109/52.268953.

JACOBSON, I. et al. Object-oriented software engineering: a use case driven approach. Reading: Addison-Wesley, 1992.

JACOBSON, Ivar. Use cases – yesterday, today, and tomorrow. Software and Systems Modeling, v. 3, n. 3, p. 210–220, 2004.

JACOBSON, Ivar; BOOCH, Grady; RUMBAUGH, James. The unified software development process. Reading, Mass.: Addison-Wesley, 1999.

OBJECT MANAGEMENT GROUP (OMG). MDA Guide Version 1.0.1. Needham: Object Management Group, 2003.

POTTS, Colin; TAKAHASHI, Kenji; ANTÓN, Annie I. Inquiry-based requirements analysis. IEEE Software, v. 11, n. 2, p. 21–32, 1994.

RATCLIFFE, Martyn; BUDGEN, David. The application of use cases in systems analysis and design specification. Information and Software Technology, v. 47, n. 9, p. 623–641, 2005.

ROLLAND, Colette et al. A proposal for a scenario classification framework. Requirements Engineering, v. 3, n. 1, 1998.

RUMBAUGH, James. Getting started: using use cases to capture requirements. Journal of Object-Oriented Programming (JOOP), v. 7, n. 5, 1994.

SUTCLIFFE, Alistair G. et al. Supporting scenario-based requirements engineering. IEEE Transactions on Software Engineering, v. 24, n. 12, 1998. DOI: 10.1109/32.738340.

THELIN, Thomas; RUNESON, Per; WOHLIN, Claes. Prioritized use cases as a vehicle for software inspections. IEEE Software, v. 20, n. 4, 2003. DOI: 10.1109/MS.2003.1207451.

WEIDENHAUPT, Klaus et al. Scenarios in system development: current practice. IEEE Software, v. 15, n. 2, 1998.

[Recebido/Received: Abril 30, 2023; Aceito/Accepted: Agosto 29, 2023]