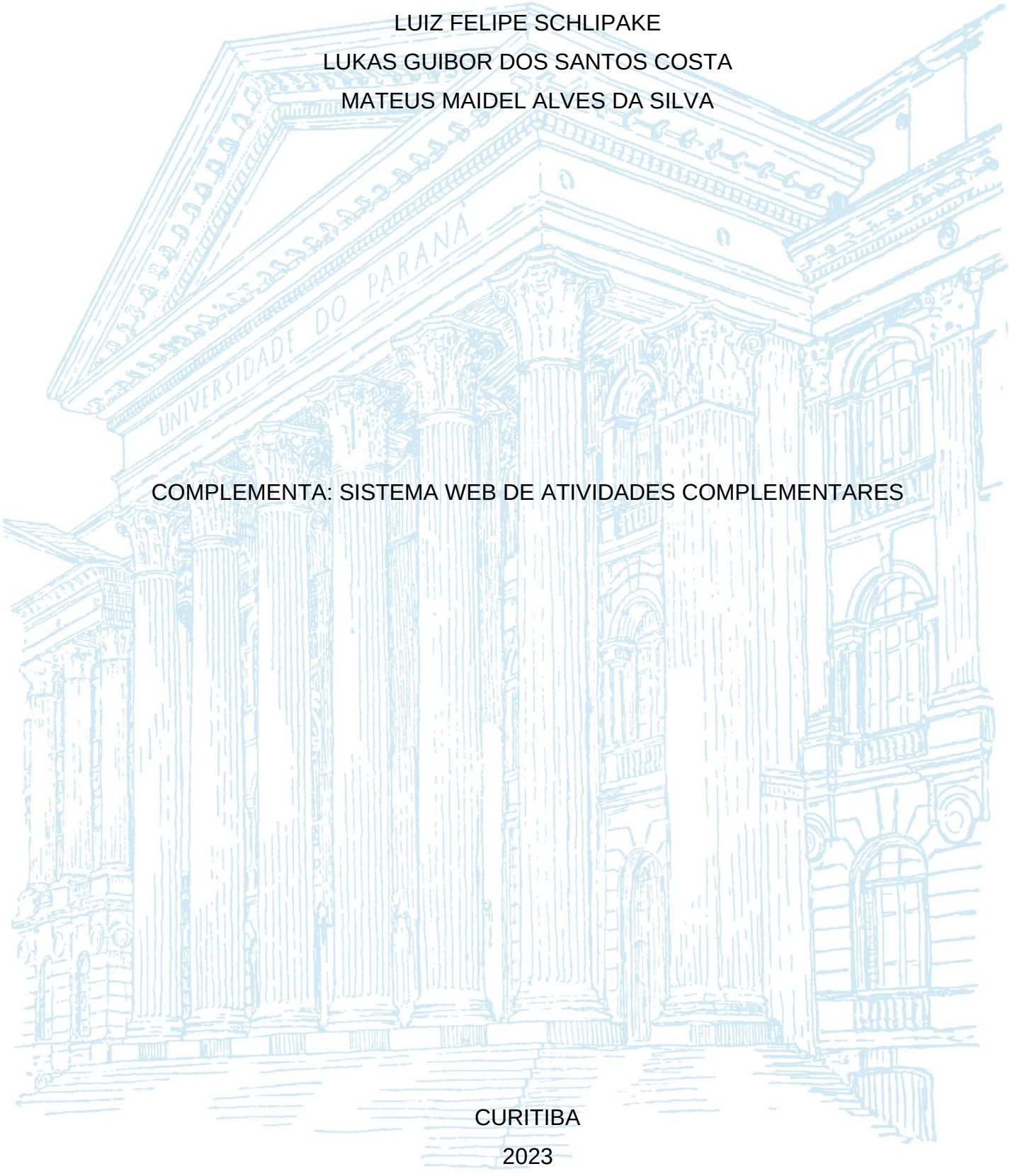
UNIVERSIDADE FEDERAL DO PARANÁ

LEONARDO HENRIQUE DE SOUZA HORTMANN
LUIZ FELIPE SCHLIPAKE
LUKAS GUIBOR DOS SANTOS COSTA
MATEUS MAIDEL ALVES DA SILVA



COMPLEMENTA: SISTEMA WEB DE ATIVIDADES COMPLEMENTARES

CURITIBA
2023

LEONARDO HENRIQUE DE SOUZA HORTMANN
LUIZ FELIPE SCHLIPAKE
LUKAS GUIBOR DOS SANTOS COSTA
MATEUS MAIDEL ALVES DA SILVA

COMPLEMENTA: SISTEMA WEB DE ATIVIDADES COMPLEMENTARES

Trabalho de Conclusão de Curso apresentado ao curso de Tecnologia em Análise e Desenvolvimento de Sistemas, Setor de Educação Profissional e Tecnológica, Universidade Federal do Paraná, como requisito parcial à obtenção do grau de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientador: Prof. Dr. Razer Anthom Nizer Rojas Montaño.

CURITIBA
2023

Utilize o estilo **Texto**. Dedicatória dedicatória dedicatória dedicatória dedicatória dedicatória dedicatória dedicatória dedicatória dedicatória dedicatória dedicatória dedicatória dedicatória dedicatória dedicatória dedicatória dedicatória.

AGRADECIMENTOS

Utilize o estilo **Texto**. Texto texto.

Texto texto.

A epígrafe possui formatação livre. Epígrafe epígrafe epígrafe epígrafe epígrafe epígrafe epígrafe epígrafe epígrafe epígrafe epígrafe epígrafe epígrafe epígrafe epígrafe. (AUTOR, ano, p.)

RESUMO

Utilize o estilo **Resumo**. Digite seu texto com no mínimo 150 palavras e no máximo 500 palavras em parágrafo único. Digite seu texto.

Palavras-chave: Palavra-chave 1. Palavra-chave 2. Palavra-chave 3. Palavra-chave 4. Palavra-chave 5.

ABSTRACT

Utilize o estilo **Resumo**. Digite seu texto com no mínimo 150 palavras e no máximo 500 palavras em parágrafo único. Digite seu texto.

Keywords: Keyword 1. Keyword 2. Keyword 3. Keyword 4. Keyword 5.

LISTA DE FIGURAS

LISTA DE GRÁFICOS

Nenhuma entrada de índice de ilustrações foi encontrada.

LISTA DE QUADROS

LISTA DE TABELAS

NENHUMA ENTRADA DE ÍNDICE DE ILUSTRAÇÕES FOI ENCONTRADA.

LISTA DE ABREVIATURAS OU SIGLAS

SIGLA - Nome por extenso
SIGLA - Nome por extenso
SIGLA - Nome por extenso
SIGLA - Nome por extenso
SIGLA - Nome por extenso

LISTA DE SÍMBOLOS

© - copyright

@ - arroba

® - marca registrada

Σ - somatório de números

Π - produtório de números

SUMÁRIO

1 INTRODUÇÃO

A universidade possui um importante papel como ambiente que, além de propiciar o desenvolvimento humano e profissional de todos da comunidade acadêmica, pode reproduzir, mais intensamente, demandas sociais encontradas no dia a dia, por meio da realização de trabalhos em equipe e interações com pessoas de diferentes características sociais e pessoais (SOARES *et al*., 2016).

Desta forma, tal espaço é local propício para a criação de uma robusta rede de relacionamentos, a qual pode ser potencializada devido à grande variedade e diversidade presente neste ambiente. Ademais, o universo acadêmico envolve conexões com pessoas das mais diversas áreas do conhecimento, e as relações concebidas no ambiente universitário podem facilmente ser estendidas ao âmbito corporativo, de modo a gerar um ciclo virtuoso de conhecimento.

Para tal, ferramentas que propiciem a interação entre colegas de diferentes cursos e ciências se mostram aliadas, principalmente se puderem oferecer, além das vantagens supracitadas, contrapartidas positivas voltadas à uma obrigação necessária a todos os alunos para a diplomação em curso superior na Universidade Federal do Paraná – a integração de atividades complementares, também conhecidas como Atividades Formativas.

De acordo com a resolução Nº 70/04 do Conselho de Ensino, Pesquisa e Extensão (CEPE), as Atividades Formativas “são constituídas de atividades complementares em relação ao eixo fundamental do currículo, objetivando sua flexibilização” (CEPE-UFPR, 2004, p. 1). Isto posto, tais atividades possuem caráter pedagógico e interdisciplinar, devendo estar alinhadas a cada curso de Graduação ou ensino profissionalizante, conforme complementa a referida resolução (CEPE-UFPR, 2004).

Isto posto, o objetivo deste projeto se caracteriza pelo desenvolvimento de uma aplicação capaz de interligar pessoas da comunidade acadêmica, aquelas que possuam necessidades específicas às que possam atendê-las, em contrapartida à obtenção de horas complementares.

1.1 PROBLEMA

Dentre as Atividades Formativas constituídas na UFPR e aprovadas pelos Colegiados de Cursos, estão desde disciplinas eletivas à estágios obrigatórios, passando por participações em seminários, eventos, congressos e programas de voluntariado, ou ainda sendo, por fim, a realização de atividades culturais, de extensão, de pesquisa ou monitoria. Como acrescenta Pivetta *et al*. (2011), vislumbrando atividades de ensino e pesquisa que estejam ligadas às demandas sociais e às vivências da comunidade, a inserção da extensão nos Planos Pedagógicos dos Cursos de Graduação tornou-se tópico de reflexão e discussão constante entre as Direções e Coordenações de Cursos.

À vista disso, eventualmente, em projetos de iniciação científica, pesquisas acadêmicas ou na elaboração de trabalhos de conclusão de curso (TCC), um aluno poderá precisar de um conhecimento específico não pertinente aos domínios cobertos pelas ementas de seu curso, e obter tais conhecimentos seria, provavelmente, custoso ou demandaria muito tempo e dedicação. A título de exemplo, poderá haver um projeto de extensão do curso de Medicina Veterinária que necessita de pequenas alterações em seu site de divulgação. Ou ainda, em um cenário similar, haveria um aluno de Análise e Desenvolvimento de Sistemas que poderá precisar de assistência para a realização de um cálculo estatístico mais complexo para seu Trabalho de Conclusão de Curso (TCC).

A Universidade Federal do Paraná possui uma ampla gama de cursos nas mais variadas áreas da ciência, tendo alunos altamente capacitados em cada uma delas. Considerando esta máxima, os problemas citados poderiam ser resolvidos rapidamente, caso um aluno de TADS no primeiro exemplo se dispusesse a auxiliar na alteração do site do projeto de extensão, e um aluno do curso de Estatística, no segundo cenário, prestasse uma consultoria para o aluno aprimorar seu TCC.

No entanto, obter a colaboração de outros departamentos pode exigir ainda mais esforço, uma vez que, frequentemente, é necessário a construção de considerável *network* para acesso a alguém com as habilidades necessárias para a resolução da adversidade e que tenha disponibilidade para prestar determinada assistência.

1.2 JUSTIFICATIVA

Ao longo das últimas décadas os avanços computacionais, digitais e eletrônicos – além do fator globalização, o qual vale a citação – têm impactado e alterado a forma como as pessoas se relacionam e se comunicam, seja no âmbito pessoal, profissional ou mesmo acadêmico.

A respeito dos dois últimos – profissional e acadêmico, em especial –, os computadores e *smartphones*, assim como toda a tecnologia embarcada nestes aparelhos e a própria Internet, mudaram (e continuam mudando) a vida das pessoas – ao ponto de, apenas no intervalo de uma única geração, algumas profissões terem sido completamente extintas, enquanto inúmeras outras estejam surgindo. Inclusive, as novas tecnologias desenvolvidas originaram a chamada Revolução Tecnocientífica, posterior às 1ª e 2ª Revoluções Industriais (KOHN e KRUEL, 2016).

Neste contexto, e por uma série de fatores, os cursos superiores se tornaram mais acessíveis e fundamentais na busca de melhores colocações no mercado de trabalho. Por outro lado, isto faz com que o Brasil possua atualmente um dos sistemas de ensino superior mais privatizados do mundo, de modo que o ensino superior seja, por muitas vezes, mais caracterizado como uma mercadoria do que como um direito social de papel transformador (PEREIRA; TELLES; LOPES, 2021). Ademais, os autores ressaltam ainda que as instituições de ensino superior (IES) brasileiras são, em sua maior parte, instituições não universitárias – não marcadas, desta forma, pela pesquisa e extensão.

Por fim, o desenvolvimento do software descrito tem como objetivo a criação de uma plataforma que permita a colaboração mútua entre os estudantes da universidade, de modo que eles possam aplicar em projetos distintos os conhecimentos captados durante o seu próprio curso. Aliado a isso, os usuários ainda terão a oportunidade de interagir com estudantes de diversas áreas e obter horas complementares, este último como contrapartida ao esforço empregado na realização de tarefas ofertadas na plataforma.

1.3 OBJETIVOS

Considerando o contexto apresentado e as dificuldades envolto a esta questão, este trabalho se propõe a construir um Portal Web, com o objetivo de permitir que alunos e professores da UFPR possam pedir, mais facilmente, auxílio a outros alunos da instituição na execução de tarefas acadêmicas pontuais. Como

compensação ao aluno executante, seriam oferecidas horas formativas compatíveis ao esforço empregado.

1.3.1 Objetivo específicos

A aplicação proposta busca, como objetivos específicos, os itens a seguir.

- Proporcionar um ambiente em que um aluno que possua uma demanda específica encontre um aluno capaz e disposto a atendê-la;
- Proporcionar um ambiente em que um aluno que deseja obter horas complementares, conhecimento, ou realizar conexões interpessoais possa buscar e localizar tarefas específicas, as quais ele possa executar;
- Promover a interação entre acadêmicos de diferentes setores, cursos, esferas sociais, áreas de atuação profissional e localidades, aproveitando a presença à nível estadual da universidade;
- Estimular a aquisição de conhecimento, o pensamento crítico e desenvolvimento de pesquisas e estudos de caráter científico.

1.4 ORGANIZAÇÃO DO DOCUMENTO

Nesta seção explicar como o documento está organizado. O que tem no Capítulo 2, o que tem no Capítulo 3 etc.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo aborda fundamentos teóricos e pesquisas adicionais que servem como base para respaldar a viabilidade do projeto e embasar as eventuais soluções para o problema apresentado.

2.1 ATIVIDADES DE EXTENSÃO

A extensão universitária pode ser descrita como um conjunto de ações sociais, as quais, em geral, são dirigidas à comunidade e das quais busca-se a obtenção de aprendizados voltados ao ensino e à pesquisa. Como um dos principais intuitos, visa a promoção e desenvolvimento social, emocional e bem-estar físico, a fim de garantir valores, direitos e deveres às pessoas (MENDONÇA *et al*., 2009).

> A universidade tem a importância de transmitir conhecimentos e saberes, nos diversos âmbitos da cultura interna ou externamente, para que o aprendizado saiba transmitir com clareza os valores culturais e humanos. [...] O conhecimento são as experiências humanas, curiosidade, leitura, educação e solidariedade, sempre em busca do melhor não apenas para si e sim para a humanidade. A busca do ser em conhecer o desconhecido e com ele uma ação transformadora para a realidade (MENDONÇA *et al*., 2009, p. 153-154).

Datados de meados da década de 1980, é possível afirmar que o reconhecimento da extensão como instrumento pedagógico na formação universitária e a consequente formalização da extensão na estrutura curricular são processos relativamente recentes, como menciona Coelho (2015). Aliás, é nesta década que se passa a discutir o conceito da Extensão Acadêmica, e a sua desmitificação – enquanto à associação como algo que remetesse à militância política, elevando a Extensão à um conceito de troca, como via de mão dupla e produção de conhecimento.

Em 1987, através da criação do Fórum Nacional de Pró-Reitores de Extensão das Universidades Públicas Brasileiras, tanto a prática da extensão como a discussão conceitual passam a ser coordenadas pela referida instância colegiada (SERRANO, 2006).

Sobre a Extensão Universitária:

> A Extensão Universitária vivencia um momento extremamente importante para sua consolidação como fazer acadêmico, entretanto as práticas institucionais através do próprio fazer extensionista e das normatizações universitárias necessitam melhor dispor-se diante das funções acadêmica, social e articuladora da Universidade. Este não é um desafio pequeno visto que o Plano Nacional de Extensão está longe de ser uma realidade plena nas universidades brasileiras (SERRANO, 2006, p. 13).

Por fim, Bussolotti *et al*. (2016) lembra que as atividades complementares, sobretudo, objetivam estimular a participação dos estudantes em experiências diversificadas e que contribuem tanto para a sua formação, como para sua futura inserção no mercado de trabalho. Afinal, impreterivelmente, tais atividades devem possuir relação direta com as Diretrizes Pedagógicas do Curso.

2.2 MODELOS DE NEGÓCIO – MODALIDADE C2C

O mercado e as transações entre empresas (B2B), entre empresas e consumidores (B2C), e entre consumidores (C2C) passou por uma revolução com a evolução da comunicação eletrônica, permitindo, por exemplo, a realização de operações financeiras, o pagamento de impostos e a realização da declaração de imposto renda, por empresas e pessoas físicas, de forma *online*. Ainda neste contexto, até mesmo a simples tomada de táxis foi facilitada, através dos inúmeros aplicativos *online*. A rapidez propiciada pela Internet contribuiu para o aperfeiçoamento de diversos serviços (READE *et al*., 2015).

Os modelos de comércio – considerando especialmente a ótica do comércio eletrônico (*e-commerce*) – são comumente definidos em quatro diferentes tipos, os quais são baseados nos papéis que envolvem a transação, sendo eles: o *Business to Consumer*, o *Business to Business*, o *Consumer to Business* e o *Consumer to Consumer* (GABRIEL; KISO, 2020).

O *Business to Business* (empresas para empresas), ou B2B, trata-se do modelo em que empresas vendem para outras empresas, tendo como principal característica as vendas do tipo a granel – em que geralmente há taxa de desconto para compras em grandes quantidades.

Já o *Business to Consumer* (empresas para consumidor), ou B2C, é o tipo mais comum no mercado, sendo empresas que vendem produtos em pequenas quantidades aos consumidores.

O *Consumer to Business* (consumidor para empresas), ou C2B, por sua vez, representa a inversão do modelo de negócio B2C. Neste modelo, o consumidor oferece e dispõe o seu serviço às empresas, conforme afirmam Gabriel e Kiso (2020).

Por último, o *Consumer to Consumer* (consumidor para consumidor), ou C2C, corresponde a um modelo de negócios em que a relação comercial ocorre inteiramente entre pessoas físicas. Como exemplo, podem ser citados neste modelo a revenda de produtos entre consumidores (classificados em geral, como o mercado de carros usados), a prestação de serviços entre consumidores (aulas online, serviços domésticos, consultorias) ou a venda de produtos *online* (softwares, bens virtuais e aplicativos).

2.3 ASPECTOS CONCEITUAIS DAS TECNOLOGIAS UTILIZADAS

Nesta seção serão descritas as definições e conceitos que conduziram o processo de modelagem e desenvolvimento do software proposto.

2.3.1 Desenvolvimento Ágil de Software

O modo como um software é produzido, desde a sua metodologia, modelo de ciclo de vida e técnicas implícitas, até as ferramentas usadas e as pessoas que o estão criando, é definido como o processo de desenvolvimento de um software (SCHACH, 2009 apud MASCHIETTO et al., 2021).

Comumente o desenvolvimento de um software é tratado como um projeto à parte ou, ao menos, é feito dentro de um projeto. E como todo projeto, haverá datas de início e fim, uma equipe e recursos, além de caracterizar a execução de um processo. Uma vez bem definido, um processo irá contar com subdivisões – podendo ser chamadas de partes, atividades ou iterações – que irão permitir a avaliação constante do progresso e a correção de problemas, quando identificados (FILHO, 2019).

Atualmente, um dos pontos mais críticos no desenvolvimento de sistemas de software são o desenvolvimento e entrega rápidos. O desenvolvimento rápido de software não foi adaptado por processos de software que almejam especificar totalmente os requisitos para então projetar, construir e testar o sistema. Logo,

aliado a isso e à insatisfação de abordagens existentes à época, surgem os processos ágeis de software, idealizados para reproduzir, rapidamente e em incrementos, softwares úteis. Os processos ágeis são caracterizados por desenvolver o sistema em uma série de versões, por meio de processos de especificação, projeto e implantação intercalados (FONTANA, 2022).

No início de 2001, um grupo de dezessete especialistas se reuniu para discutir o cenário do desenvolvimento de software à época. Acreditando que o desenvolvimento de software era envolto à processos ineficazes, pesados, burocráticos e inadequados à natureza da atividade, o grupo tinha como objetivo elaborar uma abordagem mais efetiva, com maneiras que o desenvolvimento de software pudesse ocorrer de forma mais leve, rápida e centrada em pessoas (MARTIN, 2020; PRIKLADNICKI; WILLI; MILANI, 2014).

Como resultado deste encontro, surgiu um dos movimentos mais influentes na área de software, o Manifesto Ágil. Após a sua publicação, o Manifesto Ágil e seus conceitos foram popularizados e amplamente difundidos, assim como as expressões “Desenvolvimento Ágil de Software” e “Métodos Ágeis”, resultantes dele (PRIKLADNICKI; WILLI; MILANI, 2014).

> Métodos ágeis constituem um grupo de metodologias diferentes entre si, mas caracterizadas por princípios comuns, mais baseados no trabalho cooperativo do que no formalismo e na documentação escrita. Esses princípios foram reunidos por um grupo de metodologistas conhecidos em um documento chamado de Manifesto Ágil. Esse documento proclama os seguintes valores: Indivíduos e interações, em relação a processos e ferramentas; Software funcional, em relação à documentação abrangente; Colaboração com o cliente, em relação a negociações contratuais; Resposta à mudança, em relação a seguir planos (FILHO, 2019, p. 85).

Há diversos métodos ágeis, e vários deles são bem populares. Todavia, o Scrum é um dos mais difundidos, especialmente no Brasil, onde se tornou bastante popular em tempos recentes (FILHO, 2019).

2.3.1.1 Scrum

Scrum é um método de desenvolvimento ágil de software idealizado por Jeff Sutherland e sua equipe, no início da década de 1990. Assim como outras metodologias ágeis, a metodologia Scrum foi bastante influenciada pelos bons procedimentos aplicados pela indústria japonesa, como os casos das companhias

Honda e Toyota. O nome do método, Scrum, provém de um acontecimento presente em partidas de rugby (PRESSMAN; MAXIM, 2021; SBROCCO; MACEDO, 2012).

> A denominação dessa metodologia surgiu da associação dessas equipes de projeto altamente eficazes com uma típica formação do evento esportivo rugby denominada scrum. No rugby, essa formação é utilizada após determinado incidente ou quando a bola sai de campo, ou seja, é utilizada para reiniciar o jogo, reunindo todos os jogadores. O uso dessa terminologia pareceu adequado porque no rugby cada time age em conjunto, como uma unidade integrada, cada membro desempenha um papel específico e todos se ajudam em busca de um benefício comum (BROCCO; MACEDO, 2012, p. 159).

Coerentes com o manifesto ágil, os princípios do Scrum orientam as atividades de desenvolvimento em um processo que abrange as seguintes atividades metodológicas: requisitos, análise, projeto, evolução e entrega. Em cada uma dessas atividades ocorrem, em uma determinada janela de tempo (chamada de *sprint*), tarefas específicas. A duração de uma *sprint* é de, geralmente, 2 a 4 semanas e o volume de trabalho de cada *sprint* é adaptado conforme necessário, podendo ocorrer modificações em tempo real pela equipe Scrum, se preciso (PRESSMAN; MAXIM, 2021).

Em relação à equipe, os chamados papéis essenciais de um time Scrum são dois: o *Product Owner* (dono do produto) e o *Scrum Master*. Além deles, haverá também uma pequena equipe de desenvolvimento. O time Scrum, que é interdisciplinar e auto-organizado, é responsável pelo desenvolvimento dos produtos e deve possuir todas as competências necessárias para tal.

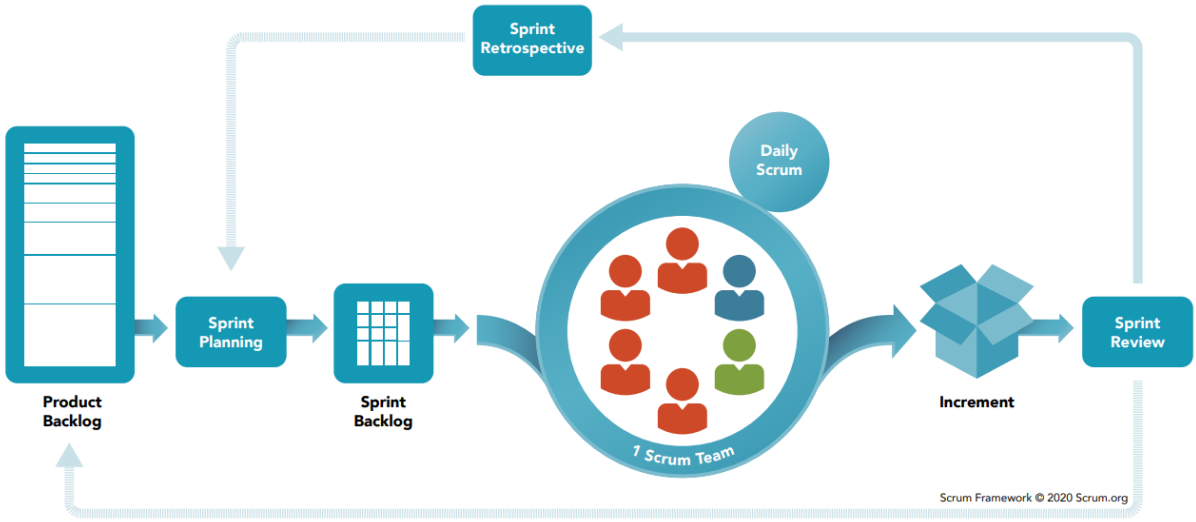
A equipe de desenvolvimento, também auto-organizada, é composta pelos profissionais que irão transformar, definindo seus próprios métodos, o *backlog* do produto em código executável – isto é, realizar o desenvolvimento do produto (FILHO, 2019). O *backlog* do produto, o *backlog* da *sprint* e o incremento de código são os principais artefatos do Scrum.

O *backlog* do produto é uma lista ordenada de requisitos ou características do artefato de tudo que puder ser necessário no produto e que agregam valor de negócio para o cliente. Com o consentimento da equipe e aprovação do *Product Owner*, itens poderão ser adicionados a qualquer momento. O refinamento do *backlog* do produto é constante, sendo cada vez mais detalhado e com estimativas cada vez mais precisas. O *backlog* da *sprint*, por sua vez, corresponde ao conjunto

de itens do *backlog* do produto que foram selecionados para serem incrementados e entregues no período daquela *sprint* (FILHO, 2019; PRESSMAN; MAXIM, 2021).

> O Incremento é a soma de todos os itens do *Backlog* do Produto completados na *Sprint* atual e nas anteriores. Ao final da *Sprint*, o novo Incremento deve atingir o estado Feito (FILHO, 2019, p. 96).

FIGURA 1 – FLUXO DO PROCESSO SCRUM



Fonte: https://www.scrum.org/resources/what-scrum-module.

Por fim, uma *sprint* possui quatro momentos específicos, que são as reuniões da *sprint*: planejamento da *sprint* (*sprint planning meeting*), reunião diária (*daily meeting* ou *daily scrum*); revisão da *sprint* (*sprint review*) e retrospectiva da *sprint* (*sprint retrospective*) (SBROCCO; MACEDO, 2012). Em relação à reunião diária, ela deverá ser um evento com duração média de 15 minutos, que será realizada ao início de cada dia de trabalho buscando o alinhamento das atividades, especialmente aquelas que devem ser realizadas nas próximas 24 horas (PRESSMAN; MAXIM, 2021).

2.3.2 *Unified Modeling Language* – UML

A *Unified Modeling Language* (UML) – ou Linguagem de Modelagem Unificada, em português – é descrita como uma linguagem visual para especificar, construir e documentar os artefatos dos sistemas (OMG, 2003 apud LARMAN, 2011).

A importância da UML advém de seu uso amplo e da padronização dentro da comunidade de desenvolvimento orientado a objetos, aponta Fowler (2011). Ademais, o autor ainda conclui que “[...] a UML se tornou não somente a notação gráfica dominante dentro do mundo orientado a objetos, como também uma técnica popular nos círculos não-orientados a objetos.” (FOWLER, 2011, p. 12).

A concepção da linguagem UML teve início com James Rumbaugh e Grady Booch, a partir do momento em que eles começaram a unificar suas notações diagramáticas e processos já conhecidos. Na sequência, Ivar Jacobson adicionou à linguagem unificada, que já estava em desenvolvimento, os seus casos de uso e mais notações (WAZLAWICK, 2014).

Constantemente revisada, a UML possui uma série de diagramas. Contudo, para Wazlawick (2014), nem todos precisam ser utilizados no desenvolvimento de um sistema, sendo recomendados apenas aqueles que representem informações úteis ao projeto.

2.3.2.1 Diagrama de classes

Para Fowler (2011), os diagramas de classes são a espinha dorsal da UML. Este tipo de diagrama apresenta um conjunto de classes, interfaces e colaborações, além de seus relacionamentos. Amplamente utilizados, os diagramas de classes estão sujeitos à maior variação de conceitos de modelagem, sendo utilizados para fazer a modelagem da visão estática de um sistema (BOOCH; RUMBAUGH; JACOBSON, 2005; FOWLER, 2011).

> Um diagrama de classes descreve os tipos de objetos presentes no sistema e os vários tipos de relacionamentos estáticos existentes entre eles. Os diagramas de classes também mostram as propriedades e as operações de uma classe e as restrições que se aplicam à maneira como os objetos estão conectados (FOWLER, 2011, p. 52).

A importância dos diagramas de classes não está apenas na visualização, especificação e documentação de modelos estruturais, mas está também na construção de sistemas executáveis por intermédio de engenharia de produção e reversa, lembram Booch, Rumbaugh e Jacobson (2005).

2.3.2.2 Diagrama de casos de uso

Buscando ilustrar os nomes dos casos de uso e dos atores, assim como os relacionamentos entre eles, a UML fornece a notação de diagramas de casos de uso, os quais servem como uma ferramenta de comunicação que irá resumir o comportamento do sistema e de seus atores (LARMAN, 2011).

Dentre os diagramas disponíveis na UML para a modelagem de aspectos dinâmicos de sistemas, os diagramas de casos de uso são um dos mais populares. No quesito modelagem do comportamento de um sistema, de um subsistema ou de uma classe, os diagramas de casos de uso possuem um papel central, uma vez que cada um mostra um conjunto de casos de uso e atores e seus relacionamentos. Logo, os diagramas de casos de uso costumam conter o assunto, os atores, os casos de uso e os relacionamentos de dependência, generalização e associação (BOOCH; RUMBAUGH; JACOBSON, 2005).

> Os diagramas de casos de uso são importantes para visualizar, especificar e documentar o comportamento de um elemento. Esses diagramas fazem com que sistemas, subsistemas e classes fiquem acessíveis e compreensíveis, por apresentarem uma visão externa sobre como esses elementos podem ser utilizados no contexto. Os diagramas de casos de uso também são importantes para testar sistemas executáveis por meio de engenharia de produção e para compreendê-los por meio de engenharia reversa (BOOCH; RUMBAUGH; JACOBSON, 2005, p. 241).

Contudo, como lembram Fowler (2011) e Larman (2011), mesmo sendo um diagrama muitas vezes útil e uma ferramenta valiosa no entendimento dos requisitos funcionais de um sistema, ele não é obrigatório. Em relação aos casos de uso, o mais importante é a escrita do texto, estando em segundo plano a ação de diagramar e focalizar os relacionamentos entre os casos de uso. O foco deve ser concentrado na redação de texto, não sendo necessário dispêndio demasiado de tempo e esmero excessivo nos diagramas de casos de uso.

2.3.2.3 Diagrama de sequência

Com o objetivo de representar a sequência de eventos e repostas em um caso de uso, os diagramas de sequência, assim como os diagramas de casos de uso, são também um dos diagramas disponíveis na UML para a modelagem de aspectos dinâmicos de sistemas (FOWLER, 2011; WAZLAWICK, 2014).

Um diagrama de sequência irá, geralmente, registrar o comportamento de um cenário específico, vários exemplos de objetos e mensagens passadas entre

esses objetos dentro de um único caso de uso. Logo, este modelo de diagrama de sequência é recomendado para mostrar as colaborações entre vários objetos dentro de um único caso de uso, conforme Fowler (2011).

Em contrapartida, os diagramas de sequência não são a melhor opção quando a questão é uma definição precisa do comportamento. Neste caso, para observar o comportamento de um único objeto em muitos casos de uso, a recomendação será um diagrama de estados (FOWLER, 2011).

2.3.2.4 Diagrama de estado

Podendo ser denominado como diagrama de estados, diagrama de gráficos de estados, ou diagrama de máquinas de estado, este é o último, dentre os diagramas disponíveis na UML para a modelagem de aspectos dinâmicos de sistemas, a ser abordado neste documento (sendo acrescentado à relação que contém os já citados diagrama de casos de uso e diagrama de sequência).

Os diagramas de estados são uma técnica conhecida para a descrição do comportamento de um sistema, em que são ilustrados os eventos e os estados interessantes de um objeto e o comportamento de um objeto em resposta a um evento, apresentando uma visão dinâmica (FOWLER, 2011; LARMAN, 2011).

> Um diagrama de máquina de estados mostra o ciclo de vida de um objeto: os eventos pelos quais ele passa, as suas transições e os estados em que ele está entre esses eventos. Não é necessário ilustrar todos os eventos possíveis. Se ocorrer um evento que não esteja representado no diagrama, ele será ignorado no que diz respeito ao diagrama de estado. Portanto, podemos criar um diagrama de máquina de estados que descreva o ciclo de vida de um objeto em níveis de detalhe arbitrariamente simples ou complexos, dependendo das nossas necessidades (LARMAN, 2011, p.491).

Para Wazlawick (2014), o diagrama de estados contribui no entendimento do negócio de uma organização, ao representar um conjunto de estados nos quais um sistema, ator ou entidade pode estar em um determinado instante. Graficamente, no diagrama de estados, as transições são mostradas como flechas, rotuladas com seus eventos correspondentes. Os estados, por sua vez, são mostrados em retângulos arredondados (LARMAN, 2011).

2.3.2.5 Diagrama entidade-relacionamento

O modelo entidade-relacionamento, amplamente conhecido como “MER”, é um padrão de modelagem conceitual, sendo um ponto de referência para propostas de modelagem de objetos – como a UML. Já a representação gráfica desde modelo é denominada diagrama entidade-relacionamento, sendo comumente conhecida como “DER” (CARDOSO; CARDOSO, 2015).

No contexto dos diagramas entidade-relacionamento, “uma ‘entidade’ é uma ‘coisa’ ou ‘objeto’ no mundo real, que é distinguível de todos os outros objetos” (SILBERSCHATZ, 2020, p. 134). A entidade é identificada como conjunto pois representa um conjunto de objetos – e não um objeto individualmente. Quando há uma associação entre várias entidades, ela é representada por um relacionamento (CARDOSO; CARDOSO, 2015).

2.4 PLATAFORMAS C2C – *CONSUMER-TO-CONSUMER*

Durante a análise da viabilidade do desenvolvimento do software proposto, foram exploradas quatro plataformas que oferecem serviços no mesmo modelo, o C2C, e podem ser acessadas online. As plataformas analisadas foram o “Fiverr”, o “Workana”, o “99Freelas” e o “GetNinjas”.

- Fiverr

O Fiverr é uma plataforma de mercado online que conecta pessoas que precisam de serviços *freelancers* a profissionais autônomos. Ele abrange uma ampla gama de categorias, como *design* gráfico, redação, programação, marketing, música e muitas outras. O Fiverr resolve o problema de empresas e indivíduos que necessitam de habilidades específicas para projetos temporários ou pontuais.

Como pontos positivos, estão a diversidade de categorias e serviços oferecidos, a possibilidade de encontrar profissionais especializados em diferentes áreas e a existência de opções de preços variados, desde trabalhos mais simples até projetos complexos.

Como pontos negativos, podem ser citadas a qualidade inconsistente em alguns serviços devido a ampla gama de profissionais e a possibilidade de haver a necessidade de filtrar cuidadosamente os perfis dos *freelancers* até encontrar o profissional mais adequado.

- Workana

O Workana é uma plataforma de trabalho *freelancer* que conecta profissionais autônomos a clientes em busca de serviços específicos. Ela engloba áreas como desenvolvimento de software, design, redação, tradução, marketing digital, dentre muitas outras. O objetivo do Workana é facilitar a contratação de talentos *freelancers* para projetos variados.

Como destaques positivos estão a grande variedade de *freelancers* disponíveis em diversas áreas de atuação, o sistema de avaliação e recomendações dos *freelancers*, o que ajuda a tomar uma decisão informada, e a possibilidade de contratar *freelancers* tanto por projeto, quanto por hora de trabalho.

Por outro lado, como pontos negativos, destaca-se a taxa cobrada do profissional que inicia em 20% e decai quanto maior for o valor cobrado pelo serviço prestado (logo, serviços mais baratos custam mais caro para o prestador do que serviços mais caros). Além disso, pode levar tempo para encontrar o *freelancer* adequado para o projeto, exigindo uma análise cuidadosa dos perfis. Por fim, alguns *freelancers* podem ter taxas mais altas, dependendo de suas habilidades e experiência.

- 99Freelas

O 99Freelas é uma plataforma brasileira que conecta *freelancers* a clientes brasileiros. Ele abrange áreas como *design*, programação, redação, tradução, marketing, entre outras diversas categorias de serviços – semelhantes ao Fiverr e ao Workana. Os *freelancers* têm perfis em que podem apresentar suas habilidades e experiências, e os clientes podem ofertar projetos para receber propostas dos *freelancers* interessados.

Nos pontos positivos, há o foco no mercado brasileiro, o que pode resultar em uma melhor adequação cultural e linguística, há a variedade de categorias de serviços oferecidos e a possibilidade de encontrar freelancers com preços competitivos.

Por outro lado, como pontos negativos, a qualidade e a confiabilidade dos *freelancers* podem variar, exigindo análise cuidadosa dos perfis, e a oferta de *freelancers* em determinadas áreas pode ser limitada.

- GetNinjas

O GetNinjas é uma plataforma que conecta pessoas que precisam de serviços locais a profissionais autônomos próximos a eles. Ele abrange uma vasta variedade de categorias, como serviços domésticos, reformas, aulas particulares, eventos, assistência técnica, dentre outros. O objetivo do GetNinjas é simplificar a contratação de serviços locais.

Como pontos positivos estão o fácil acesso a profissionais locais em diferentes áreas de serviço, o sistema de avaliação e recomendações que auxiliam na escolha de profissionais confiáveis e a possibilidade de solicitar orçamentos personalizados e comparar diferentes opções.

Como pontos negativos, a qualidade e a confiabilidade dos profissionais podem variar, exigindo pesquisa e análise cuidadosas, e algumas áreas de serviço podem ter uma oferta limitada de profissionais disponíveis.

2.4.1 Comparativo

Finalmente, após a observação das quatro plataformas, destacam-se o Fiver e o Workana.

O Fiverr é conhecido por sua ampla gama de categorias e serviços disponíveis, incluindo *design* gráfico, redação, programação, marketing, música, dentre outros. Os serviços são oferecidos por profissionais *freelancers* de todo o mundo, permitindo que os clientes encontrem especialistas em diferentes áreas. O Fiverr também é conhecido por sua estrutura de preços, onde os serviços são geralmente oferecidos a partir de um valor fixo de US$ 5, embora também haja opções de preços mais elevados para serviços mais complexos.

Por outro lado, o Workana também é uma plataforma de freelancers, mas com foco na América Latina. Ele abrange uma ampla variedade de categorias semelhantes, como desenvolvimento de software, design, redação, tradução, marketing digital e entre outros. O Workana permite que os clientes contratem

freelancers tanto por projeto quanto por hora de trabalho. Além disso, a plataforma possui um sistema de avaliação e recomendações que ajuda os clientes a tomar decisões informadas ao escolher um *freelancer*.

3 MATERIAIS E MÉTODOS

Este capítulo descreve como o Scrum foi utilizado no escopo do projeto, as adaptações necessárias na metodologia e a distribuição das atividades entre os integrantes. Ademais, nesta seção são também descritas as ferramentas empregadas para a elaboração do projeto, prototipação e as tecnologias que serão utilizadas durante o desenvolvimento.

3.1 METODOLOGIA DE DESENVOLVIMENTO

A metodologia Scrum foi a escolhida para que houvesse entregas semanais do projeto, de modo que pudessem ser avaliadas pelo orientador e pela equipe, assim como permitir a auto-organização e autonomia dos integrantes.

No âmbito de um trabalho acadêmico colaborativo, não houve necessidade de desempenhar papéis de *Product Owner* e *Scrum Master*, pois o *Backlog* do Produto foi definido em conjunto por todos os integrantes.

Definiu-se que cada *sprint* teria duração de duas semanas, com cada integrante entregando 8 horas ou 1 ponto de trabalho por *sprint*. As reuniões foram semanais, realizadas às segundas-feiras. A reunião do início era utilizada como *Review* e *Planning*, e a reunião no meio da *sprint* era um *checkpoint* para remover eventuais impedimentos do desenvolvimento das atividades.

3.1.1 Planejamento e Arquitetura

Na primeira reunião, juntamente com o orientador, foi definido o escopo do projeto e, após comparação com os sistemas descritos na seção 2.4., foi elaborada a seguinte lista de requisitos:

a) Auto cadastro de Alunos;
b) Cadastro de uma Graduação por um Coordenador, assim como suas competências e a complexidades;
c) Cadastro de Projetos por um Professor;
d) Cadastro de Atividades por alunos integrantes de um projeto, para serem realizadas pela comunidade acadêmica;

e) Visualização de Atividades Disponíveis por integrantes da comunidade acadêmica para candidatura e posterior execução;
f) Geração de Certificado de Conclusão com Horas Formativas;
g) Construção de uma rede de contatos dentro da comunidade.

A partir da lista de requisitos foi elaborado o *backlog* de produto, contendo as atividades realizadas por cada integrante a cada *sprint*.

QUADRO 1 – SPRINTS REALIZADAS

Fonte: Os Autores (2023)

Quadro do backlog da próxima release, que é a parte 2;
Para a parte 2 a equipe será dividida em front e backend. Integrantes de cada parte.

(calcular planejamento de releases e montar quadro)
Determinar condições de satisfação
Estimar as histórias de usuário
Priorizar HUs
Estimar velocidade
Selecionar histórias e data de release de cada uma
Foram selecionadas as ferramentas de desenvolvimento, da seção 3.2.

3.1.2 Releases
3.1.2.1 Sprint 1
3.1.2.2 Sprint 2
3.1.2.3 Sprint 3
3.1.2.4 .....

3.2 FERRAMENTAS DE DESENVOLVIMENTO

São apresentadas neste item as ferramentas que foram utilizadas no desenvolvimento do software. Para teste da aplicação das tecnologias, foi escolhida a Prova de Conceito (*Proof of Concept - PoC*), que consiste no desenvolvimento das funcionalidades de auto-cadastro e login de usuário.

3.2.1 API REST

API REST, ou *API RESTful*, é uma interface de programação de aplicações (*API* ou *API web*). Ela está em conformidade com as restrições do estilo de arquitetura *REST*, permitindo a interação com serviços web *RESTful*. O termo *REST* e a criação desta arquitetura são de autoria do cientista da computação Roy T. Fielding. O termo, que vem do inglês "*Representational State Transfer*", significa transferência de estado representacional. A sigla API, por sua vez, vem de "*application programming interface*" (REDHAT, 2020).

3.2.1.1 Java

O Java é uma das mais populares linguagens de programação e amplamente conhecida por sua robustez, segurança e portabilidade, sendo utilizado em diversos tipos de aplicações. A rica biblioteca de classes do Java e JVM (*Java Virtual Machine*) lhe conferem grande flexibilidade, de modo que esta linguagem seja utilizada como base para uma ampla diversidade de *frameworks* com os mais variados propósitos.

3.2.1.2 Spring Boot

Para implementação da *API RESTful* foi utilizado *o framework Spring Boot*, uma vez que ele fornece ferramentas que facilitam a criação de *endpoints REST*. O *Spring Boot* contém um contêiner de *Apache Tomcat* embutido e um servidor web de código aberto, desenvolvido pela *Apache Software Foundation*, dispensando deste modo a necessidade de configuração de um servidor à parte (SPRING, 2023).

3.2.1.3 JPA

A *Java Persistence API* (JPA) é uma interface de programação para mapeamento objeto-relacional em aplicações Java, facilitando o processo de persistência de dados. A JPA permite que o desenvolvimento interaja com os sistemas gerenciadores de bancos de dados por meio de objetos Java, eliminando a necessidade de lidar com o banco de dados diretamente com o SQL (JAKARTA, 2023).

3.2.1.4 Hibernate

O *Hibernate* é um *framework* de mapeamento objeto-relacional que obedece a especificação do JPA. O *Hibernate* simplifica a interação entre as entidades Java e o banco de dados relacional, fornecendo recursos avançados de gerenciamento de transações, criação e atualização automática de estruturas e consultas de banco de dados (HIBERNATE, 2023).

3.2.2 PostgreSQL

PostgreSQL foi a opção escolhida como Sistema Gerenciador de Banco de Dados (SGBD). Trata-se de um banco de dados SQL objeto-relacional de código-aberto, de fácil utilização, estável e amplamente difundido no mercado, executado em todos os principais sistemas operacionais.

A origem do PostgreSQL data de quase quatro décadas, quando foi parte do projeto POSTGRES da *University of California* em 1986. Desde então são 35 anos de desenvolvimento ativo na plataforma principal, com forte reputação por sua arquitetura comprovada, confiabilidade, integridade de dados, conjunto robusto de recursos e extensibilidade. É visto como o banco de dados relacional de software livre preferido por muitas pessoas e organizações (POSTGRESQL, 2023).

3.2.3 Docker

O Docker permite criar, empacotar e a executar aplicativos em contêineres. Os contêineres são máquinas virtuais isoladas e leves contendo somente o

necessário para executar um aplicativo (código, bibliotecas e dependências). O Docker simplifica o processo de implantação e distribuição de aplicações, oferecendo flexibilidade, portabilidade, isolamento e escalabilidade horizontal.

3.2.4 API Gateway

Atuando como uma camada intermediária entre os serviços de APIs e seus clientes, o *API Gateway* é um padrão de projeto de arquiteturas de sistemas distribuídos e microsserviços, que aprimora e simplifica a interação dos clientes com múltiplas APIs. Se implementado de forma correta, é o único ponto de entrada das requisições exposto aos clientes. Contar o *API Gateway* no sistema, mesmo que a arquitetura do projeto seja monolítica, permite a implementação de recursos adicionais como autenticação, autorização, segurança, monitoramento e controle de tráfego de maneira simplificada (RICHARDSON, 2018).

3.2.4.1 JavaScript

O *JavaScript*, originalmente intitulado *ECMAScript*, é uma linguagem de programação criada em 1995 com o objetivo de prover interatividade e dinamismo em aplicações web, sendo executada diretamente por navegadores. Com o aumento da popularidade da Internet, a linguagem se tornou o grande pilar central de desenvolvimento web, levando à ampliação dos cenários onde pode ser utilizada.

3.2.4.2 Node.js

O Node.js admite a execução do *JavaScript* no servidor, permitindo aos desenvolvedores já familiarizados com a linguagem a construção de aplicações *server-side* com alta escalabilidade e desempenho. O Node.js possui uma abordagem assíncrona baseada em eventos, sendo capaz de lidar com uma grande quantidade de solicitações sem bloquear a *thread* principal. Tendo em vista estas vantagens e a ampla gama de pacotes e bibliotecas disponibilizadas pelo NPM (*Node Package Manager*), o Node.js foi utilizado para a implementação do *API Gateway*.

3.2.5 Angular

Para o desenvolvimento do *front-end* foi utilizado o *framework* Angular. Desenvolvido e mantido pelo Google, o Angular permite a construção de aplicativos web escaláveis e de alto desempenho. O Angular utiliza uma abordagem baseada em componentes reutilizáveis, agilizando o processo de desenvolvimento aplicações web com arquitetura SPA (*Single-Page Application*) robustas e modernas.

3.2.5.1 Typescript

O Angular utiliza como base o TypeScript, um superconjunto do JavaScript desenvolvido pela Microsoft. O TypeScript adiciona tipagem estática, detecção de erros em tempo de compilação e recursos de orientação a objetos avançados ao JavaScript, oferecendo um ambiente de desenvolvimento mais confiável, evitando erros comuns e facilitando a manutenção e legibilidade do código. O TypeScript possui um módulo que transpila, ou seja, converte todo o código em JavaScript puro, o que possibilita sua execução em ambientes JavaScript, como navegadores.

3.2.5.2 Bootstrap

Bootstrap é um *framework front-end* destinado ao desenvolvimento web responsivo. Ele possui uma coleção de estilos CSS predefinidos, *scripts* JavaScript e componentes interativos que facilitam a criação de *layouts* modernos e atraentes, garantindo consistência visual a diversos tamanhos de tela à aplicação.

3.2.6 Astah

Para a realização da modelagem dos dados, a ferramenta utilizada foi o software Astah, o qual se propõe a ser um editor UML integrado com recursos de *Mind Mapping* e é referência na construção dos diagramas presentes na UML. O Astah, desenvolvido pela *Change Vision, Inc*., contém uma grande variedade de diagramas e está disponível para dispositivos Windows, Mac e Linux. A ferramenta fornece transição entre diagramas, tabelas e plataformas, podendo ser utilizado da modelagem de sistemas mais simples aos mais complexos (ASTAH, 2022).

3.2.7 Lucidchart

Lucidchart é uma plataforma voltada para a criação e apresentação de diagramas, organogramas, fluxogramas e mapas mentais. Com uma interface de fácil manuseio e disponível nos principais sistemas operacionais, o Lucidchart possui também aplicativo gratuito para mobile (LUCID, 2023).

3.2.8 Figma

Para a prototipação das telas foi selecionada a plataforma Figma. Amplamente conhecida e utilizada nos mais variados meios, Figma é uma plataforma colaborativa disponível para a construção de design de interfaces e protótipos (VILLAIN, 2022). Lançada em 2015 por Evan Wallace e Dylan Field, a ferramenta de design possui acessibilidade web e funcionalidade de um aplicativo nativo. O Figma possui versões gratuitas (FIGMA, 2023).

4 CONSIDERAÇÕES PARCIAIS

Texto texto.

REFERÊNCIAS

AGILE MANIFESTO. **Manifesto for Agile Software Development**. 2001. Disponível em: <http://agilemanifesto.org/iso/ptbr/manifesto.html>. Acesso em: 19 jun. 2023.

ASTAH. **About Astah & Change Vision, Inc**. 2023. Disponível em: <https://astah.net/about/>. Acesso em: 19 jun. 2023.

BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. **UML: guia do usuário**. 6°Reimpressão. Rio de Janeiro: Elsevier, 2005.

BUSSOLOTTI, Juliana Marcondes, *et al*. **A importância das atividades complementares no processo de aprendizado: percepção dos alunos de cursos de educação a distância da universidade de Taubaté**. Congresso Internacional ABED de Educação a Distância. Vol. 22. 2016.

CARDOSO, Giselle; CARDOSO, Virgínia M. **Sistemas de Banco de Dados: uma abordagem introdutória e aplicada**. 1ª Edição. São Paulo: Editora Saraiva, 2012. E-book. ISBN 9788502162839.

COELHO, G. C. **O papel pedagógico da extensão universitária**. Revista Em Extensão, Uberlândia, MG, v. 13, n. 2, p. 11–24, 2015. DOI: 10.14393/REE-v13n22014_art01. Disponível em: <https://seer.ufu.br/index.php/revextensao/article/view/26682>. Acesso em: 21 mai. 2023.

COELHO, Larissa Martins Santos e RIBEIRO, Thiago de Luca Sant'ana e COSTA, Benny Kramer. **Importância das relações C2C na cocriação de valores em organizações de turismo**. Revista Hospitalidade, v. 15, n. 2, p. 182-193, 2018. Disponível em: <https://doi.org/10.21714/2179-9164.2018v15n2.011>. Acesso em: 30 mai. 2023
.

DOCKER. **Why Docker**. 2023. Disponível em: <https://www.docker.com/why-docker/>. Acesso em: 19 jun. 2023.

FIGMA. **Figma - About**. 2023. Disponível em: <https://www.figma.com/about/>. Acesso em: 19 jun. 2023.

FILHO, Wilson de Pádua P. **Engenharia de Software - Produtos - Vol.1**. São Paulo: Grupo GEN, 2019. E-book. ISBN 9788521636724.

FONTANA, Rafaela Mantovani**. Processo Ágil de Software**. Curitiba, 2022. Informal - Material de aula.

FOWLER, Martin. **UML essencial: um breve guia para a linguagem-padrão de modelagem de objetos**. Porto Alegre: Grupo A, 2011. E-book. ISBN 9788560031382.

GABRIEL, Martha; KISO, Rafael. **Marketing na Era Digital - Conceitos, Plataformas e Estratégias**. São Paulo: Grupo GEN, 2020. E-book. ISBN 9788597025859.

HIBERNATE. **Hibernate ORM**. 2023. Disponível em: <https://hibernate.org/orm/>. Acesso em: 19 jun. 2023.

KOHN, Vivian Helena; KRUEL Alexandra Jochims**. O comércio c2c nas redes sociais: uma análise de grupos no facebook**. DESENVOLVE: Revista de Gestão do Unilasalle, Canoas, v. 5, n. 2, p. 97-125, jul. 2016.

JAKARTA.EE. **Jakarta Persistence**. 2023. Disponível em: <https://jakarta.ee/specifications/persistence/>. Acesso em: 19 jun. 2023.

LARMAN, Craig. **Utilizando UML e padrões**. Porto Alegre: Grupo A, 2011. E-book. ISBN 9788577800476.

LIMA, Fábio. **O comércio electrónico e as plataformas B2C e C2C: contribuições para o estudo do comportamento do consumidor online**. - Lisboa: Escola Superior de Comunicação Social, 2012. - Dissertação de mestrado.

LUCID. **Lucidchart**. 2023. Disponível em: <https://www.lucidchart.com/pages/pt/produto>. Acesso em: 19 jun. 2023.

MARTIN, Robert C. **Desenvolvimento Ágil Limpo**. Rio de Janeiro: Editora Alta Books, 2020. E-book. ISBN 9788550816890.

MASCHIETTO, Luis G.; MORAES, Diego Martins Polla de; ALVES, Nicolli Souza R.; *et al*. **Desenvolvimento de Software com Metodologias Ágeis**. Porto Alegre: Grupo A, 2021. E-book. ISBN 9786556901824.

MENDONÇA, I. B.; COSTA, C. L. N. do A.; SANTOS, B. A. A. dos; SILVA, L. B. da; DANTAS, A. C. L.; DOS SANTOS, A. P.; BARROS, C. C.; IZIDORIO, E. de C. **Extensão universitária em parceria com a sociedade**. Caderno de Graduação - Ciências Humanas e Sociais - UNIT - SERGIPE, [S. l.], v. 1, n. 2, p. 149–155, 2013. Disponível em: <https://periodicos.set.edu.br/cadernohumanas/article/view/535>. Acesso em: 01 jun. 2023.

MENEGASSI, Adriana Ribeiro; MONTE-MOR, Danilo Soares**. Fatores que influenciam o consumo por status de adolescentes: uma análise incluindo o *consumer-to-consumer***. Competência, Porto Alegre, v. 11, n. 2, dez. 2018.

OMG – Object Management Group, 2003. **UML 2.0 *Infrastructure Specification***. Disponível em: <www.omg.org>. Acesso em: 16 jun. 2023.

PEREIRA, Larissa Dahmer; TELLES, Andreza; LOPES, Gabriella de Souza. **Formação em tempos de pandemia: análise das atividades formativas desenvolvidas pelos cursos presenciais de Serviço Social no ano de 2020**. Revista da Faculdade de Serviço Social da Universidade do Estado do Rio de

Janeiro. EM PAUTA, Rio de Janeiro. 2º Semestre de 2021 - n. 48, v. 19, p. 203 – 218.

PIVETTA, H. M. F.; BACKES, D. S.; CARPES, A.; BATTISTEL, A. L. H. T.; MARCHIORI, M. **Ensino, pesquisa e extensão universitária: em busca de uma integração efetiva**. Linhas Críticas, [S. l.], v. 16, n. 31, p. 377–390, 2011. DOI: 10.26512/lc.v16i31.3634. Disponível em: <https://periodicos.unb.br/index.php/linhascriticas/article/view/3634>. Acesso em: 21 mai. 2023.

POSTGRESQL. ***About: What is PostgreSQL?***. 2023. Disponível em: <https://www.postgresql.org/about/>. Acesso em: 19 jun. 2023.

PRESSMAN, Roger S.; MAXIM, Bruce R. **Engenharia de software**. Porto Alegre: Grupo A, 2021. E-book. ISBN 9786558040118.

PRIKLADNICKI, Rafael; WILLI, Renato; MILANI, Fabiano. **Métodos ágeis para desenvolvimento de software**. Porto Alegre: Grupo A, 2014. E-book. ISBN 9788582602089.

READE, Dennis V.; ROCHA, Marcos; OLIVEIRA, Sérgio Luis Ignácio de; CHERNIOGLO, Andréa. **Marketing B2B**. São Paulo: Editora Saraiva, 2015. E-book. ISBN 978-85-02-63884-6.

REDHAT. **API REST**. 2020. Disponível em: < https://www.redhat.com/pt-br/topics/api/what-is-a-rest-api>. Acesso em: 19 jun. 2023.

RICHARDSON, Chris. ***Pattern: API Gateway / Backends for Frontends***. 2018. Disponível em: <https://microservices.io/patterns/apigateway.html>. Acesso em: 19 jun. 2023.

SBROCCO, José Henrique Teixeira de C.; MACEDO, Paulo Cesar de. **Metodologias Ágeis - Engenharia de Software sob Medida**. São Paulo: Editora Saraiva, 2012. E-book. ISBN 9788536519418.

SCRUM. ***What is Scrum?***. 2023. Disponível em: <https://www.scrum.org/resources/what-scrum-module>. Acesso em: 20 jun. 2023.

SERRANO, R. M. S. M. **Conceitos de extensão universitária: um diálogo com Paulo Freire**. In: Pró-reitoria de extensão e assuntos comunitários – PRAC, João Pessoa, fev. 2006.

SILBERSCHATZ, Abraham. **Sistema de Banco de Dados**. Rio de Janeiro: Grupo GEN, 2020. E-book. ISBN 9788595157552.

SOARES, A. B.; GOMES, G.; MAIA, F. A.; GOMES, C. A. O.; MONTEIRO, M. C. **Relações interpessoais na universidade: o que pensam estudantes da graduação em Psicologia?.** Estudos Interdisciplinares em Psicologia, [S. l.], v. 7, n. 1, p. 56–76, 2016. DOI: 10.5433/2236-6407.2016v7n1p56. Disponível em:

<https://ojs.uel.br/revistas/uel/index.php/eip/article/view/23794>. Acesso em: 28 mai. 2023.

SPRING. **Spring**. 2023. Disponível em: <https://spring.io/>. Acesso em: 19 jun. 2023.

TILKOV, Stefan. **Uma rápida introdução ao REST**. 2008. Disponível em: <https://www.infoq.com/br/articles/rest-introduction/>. Acesso em: 19 jun. 2023.

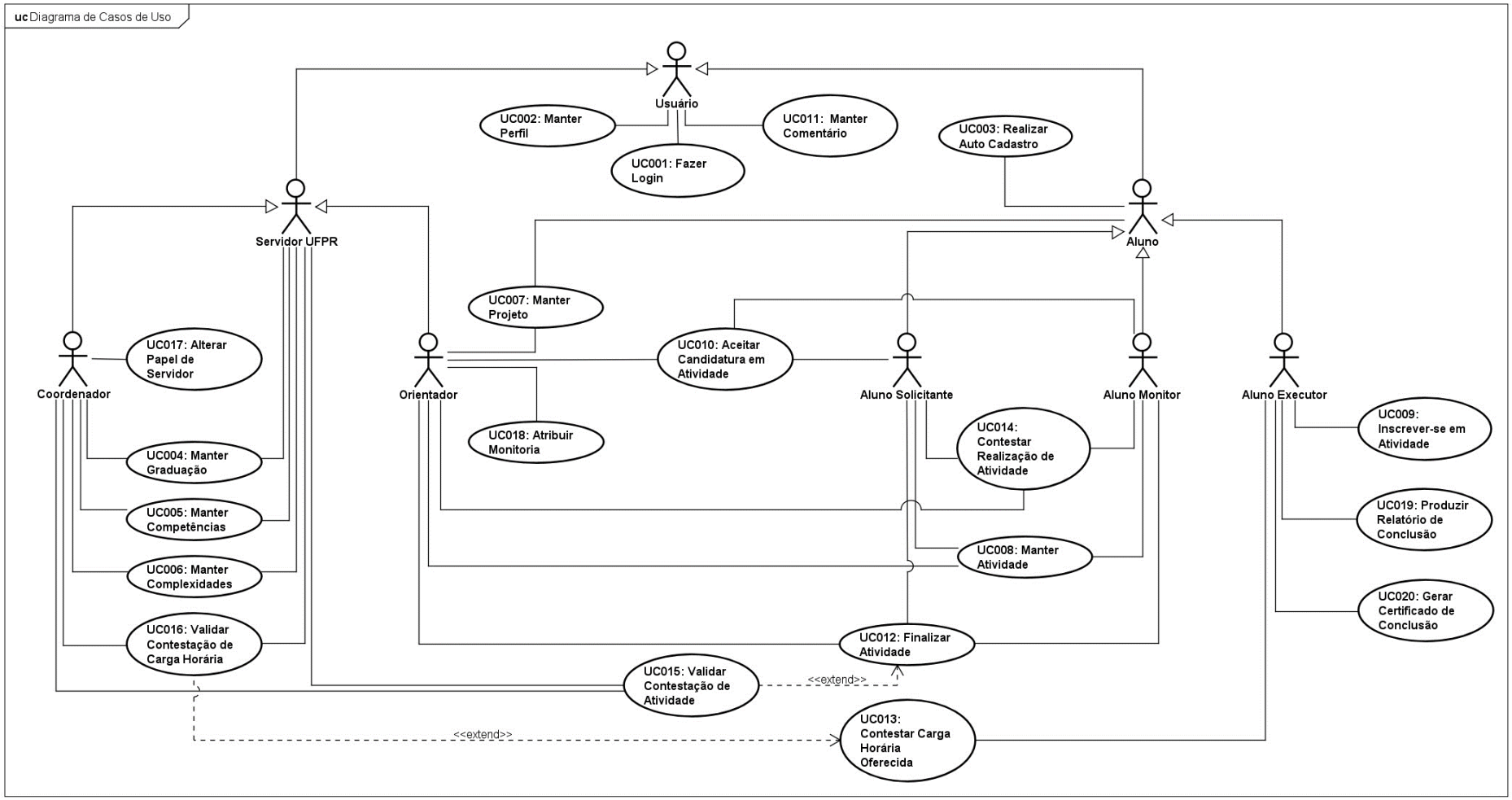
UNIVERSIDADE FEDERAL DO PARANÁ (UFPR). CONSELHO DE ENSINO, PESQUISA E EXTENSÃO (CEPE). **RESOLUÇÃO Nº 70/04**. 2004.

VILLAIN, Mateus. **Figma: o que é a ferramenta, Design e uso**. Alura: 2022. Disponível em: <https://www.alura.com.br/artigos/figma#o-que-e-figma?>. Acesso em: 19 jun. 2023.

WAZLAWICK, Raul S. **Análise e Design Orientados a Objetos para Sistemas de Informação: Modelagem com UML, OCL e IFML**. São Paulo: Grupo GEN, 2014. E-book. ISBN 9788595153653.

APÊNDICE A – DIAGRAMA DE CASOS DE USO

FIGURA 2 – DIAGRAMA DE CASO DE USO



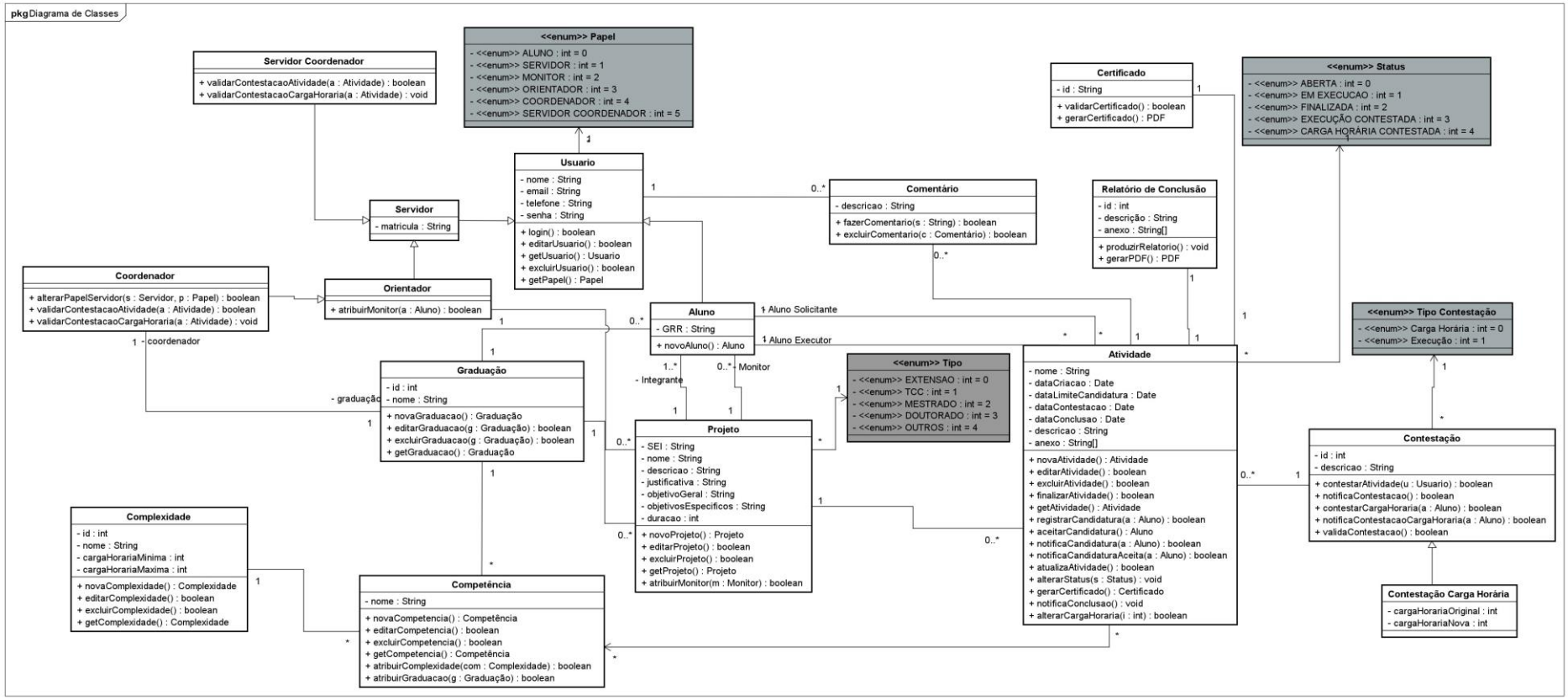
Fonte: Os Autores (2023).

APÊNDICE B – HISTÓRIAS DE USUÁRIO E PROTÓTIPOS DE TELA

Formatação livre.

APÊNDICE C – DIAGRAMA DE CLASSE

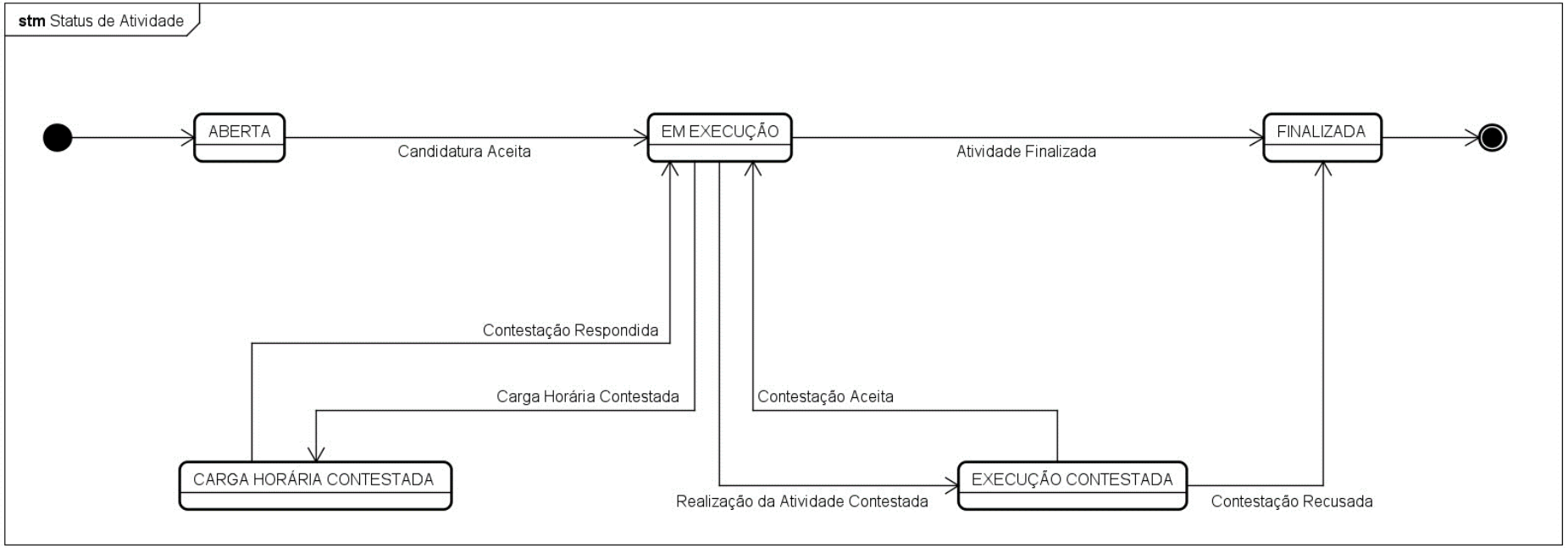
FIGURA 3 – DIAGRAMA DE CLASSES



Fonte: Os Autores (2023).

APÊNDICE D – DIAGRAMA DE ESTADO

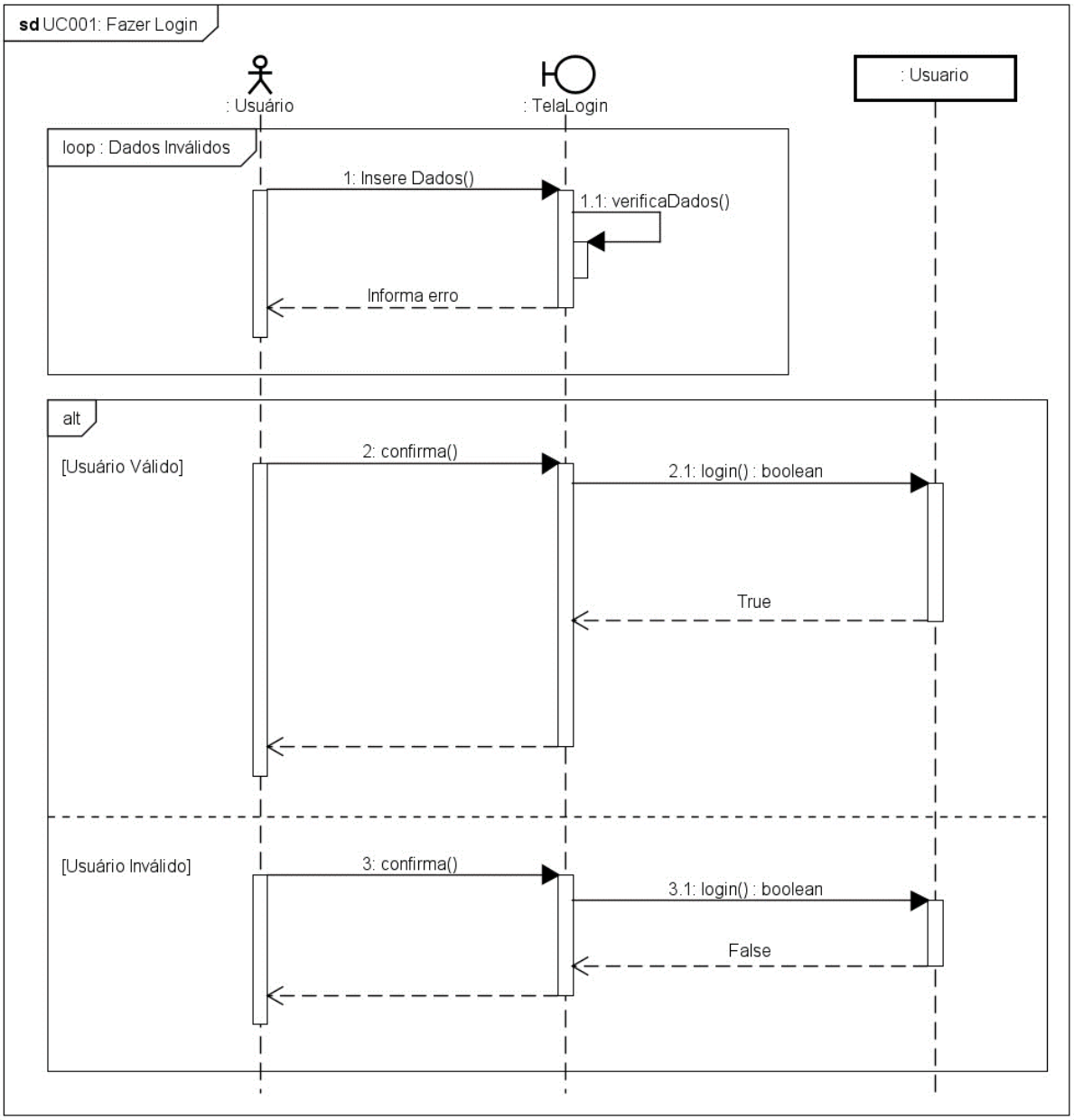
FIGURA 4 – DIAGRAMA DE ESTADO



Fonte: Os Autores (2023).

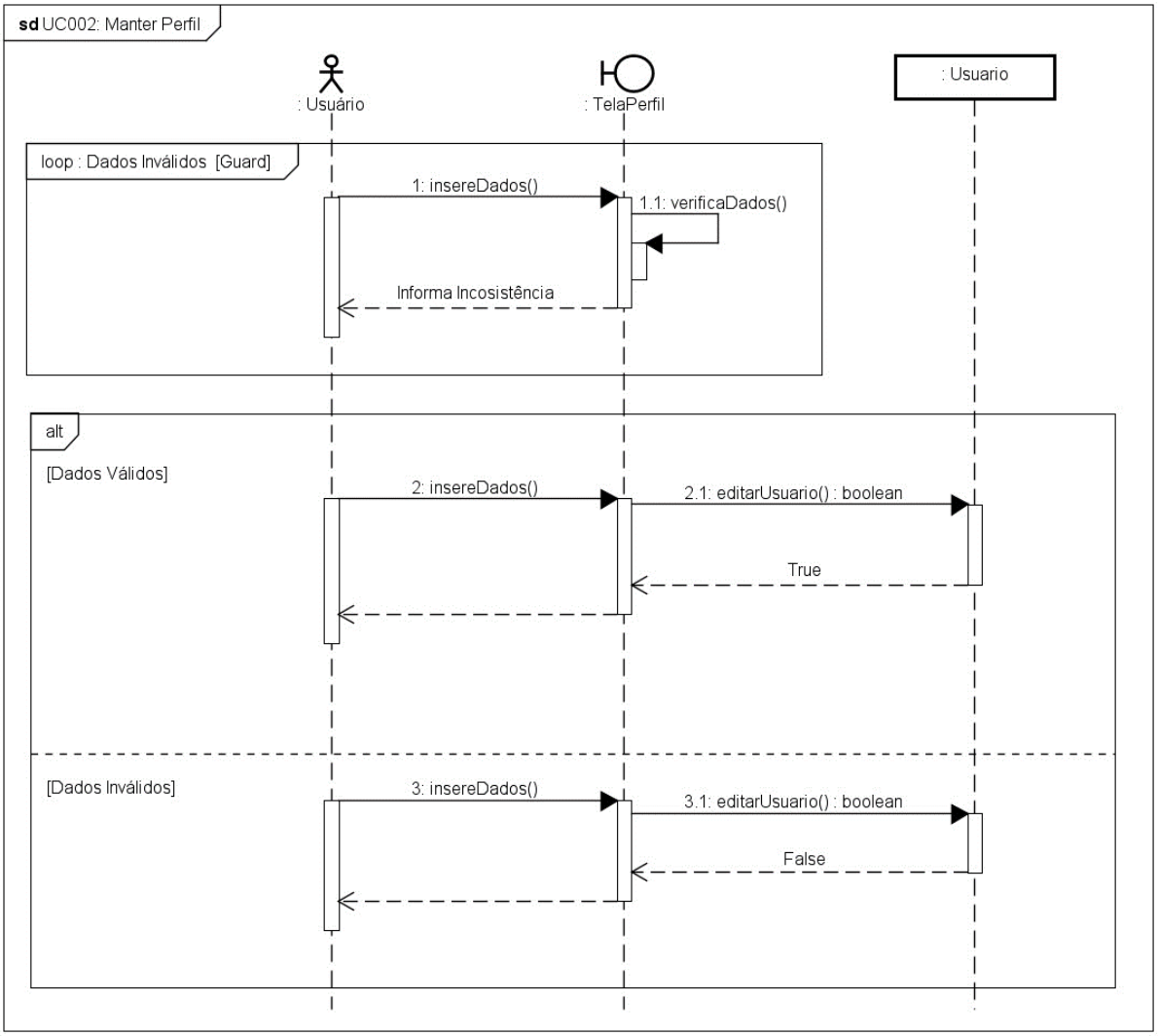
APÊNDICE E – DIAGRAMAS DE SEQUÊNCIA

FIGURA 5 – UC001: FAZER LOGIN



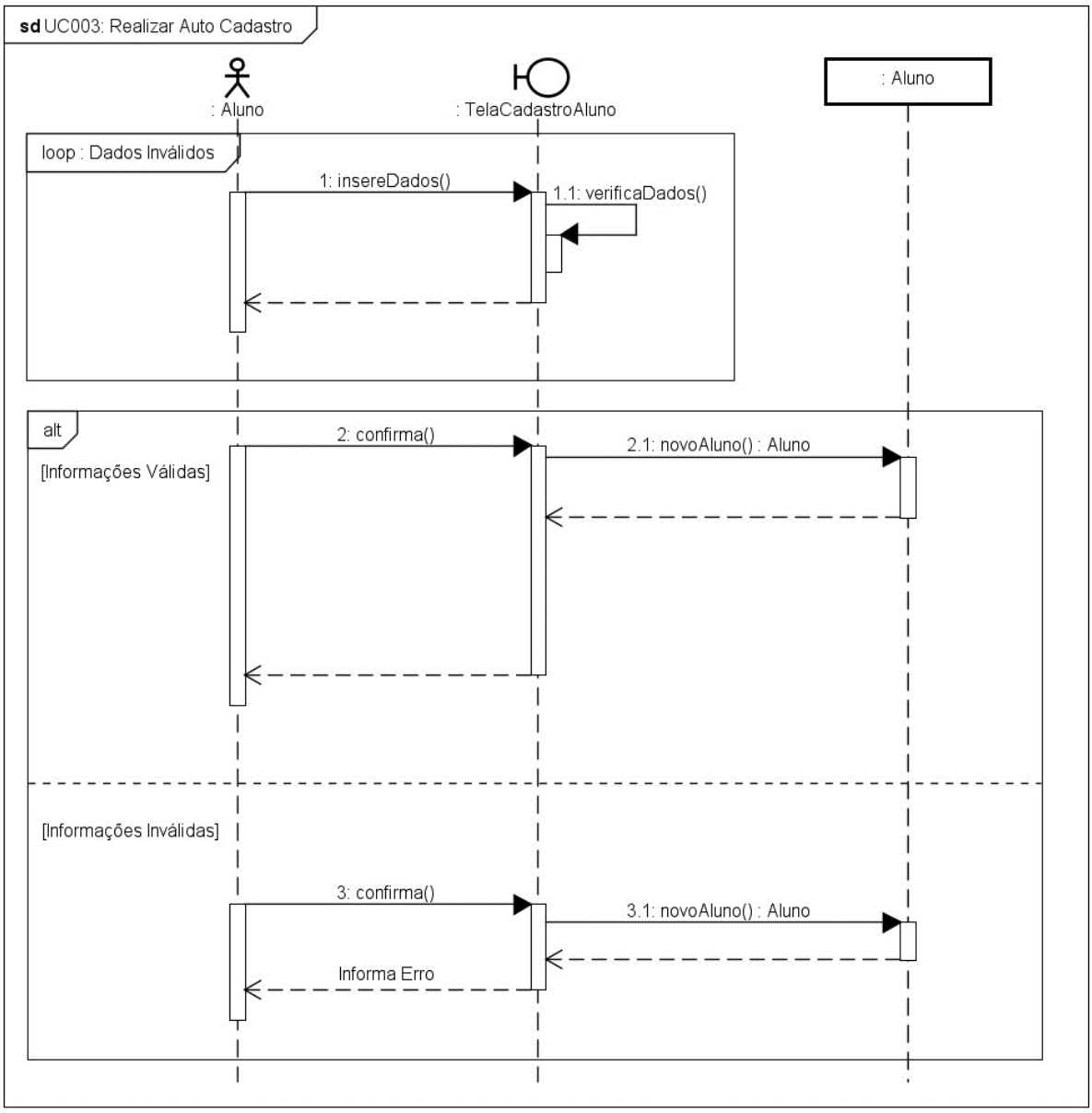
Fonte: Os Autores (2023).

FIGURA 6 – UC002: MANTER PERFIL



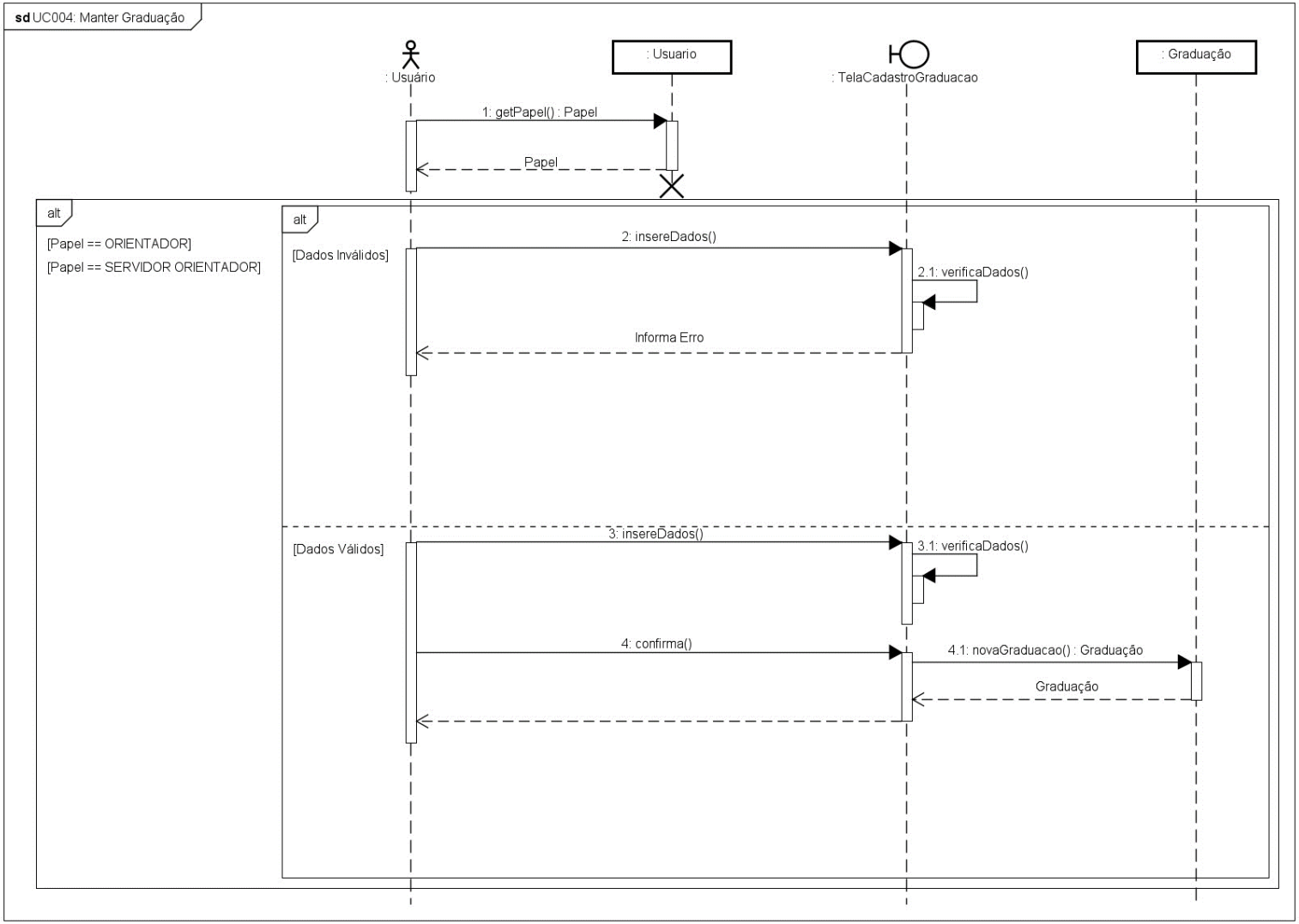
Fonte: Os Autores (2023).

FIGURA 7 – UC003: REALIZAR AUTO CADASTRO



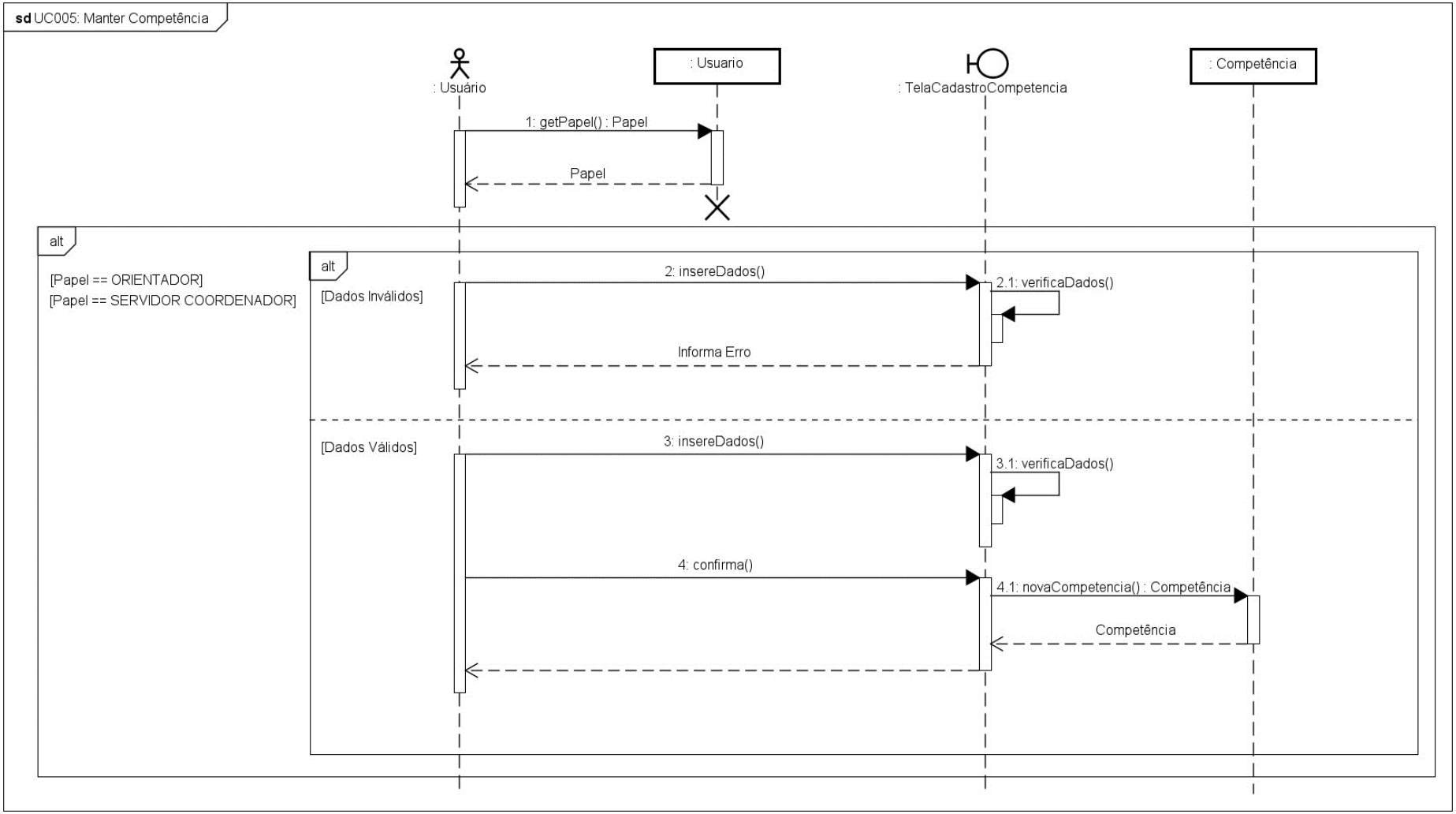
Fonte: Os Autores (2023).

FIGURA 8 – UC004: MANTER GRADUAÇÃO



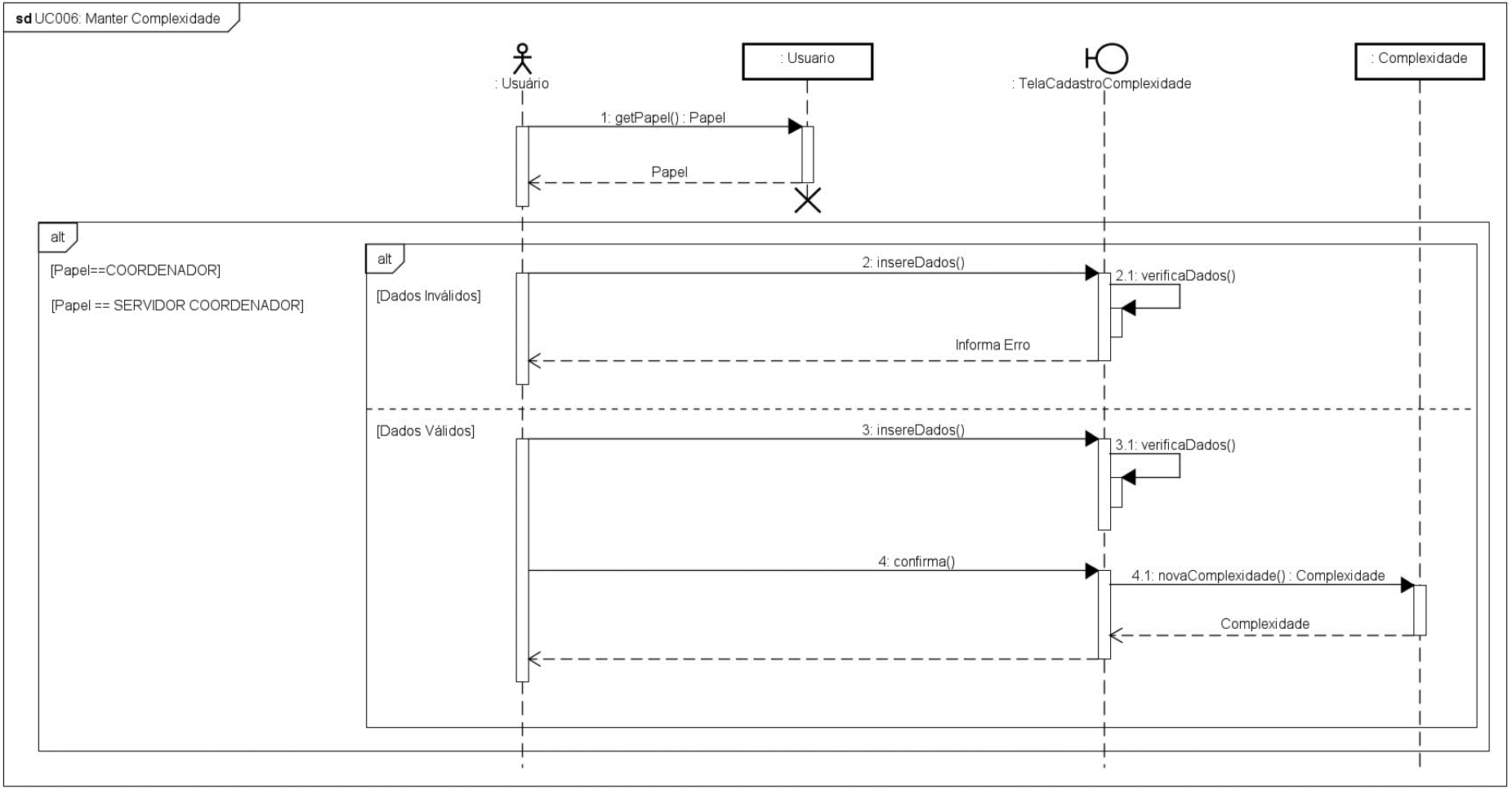
Fonte: Os Autores (2023).

FIGURA 9 – UC005: MANTER COMPETÊNCIA



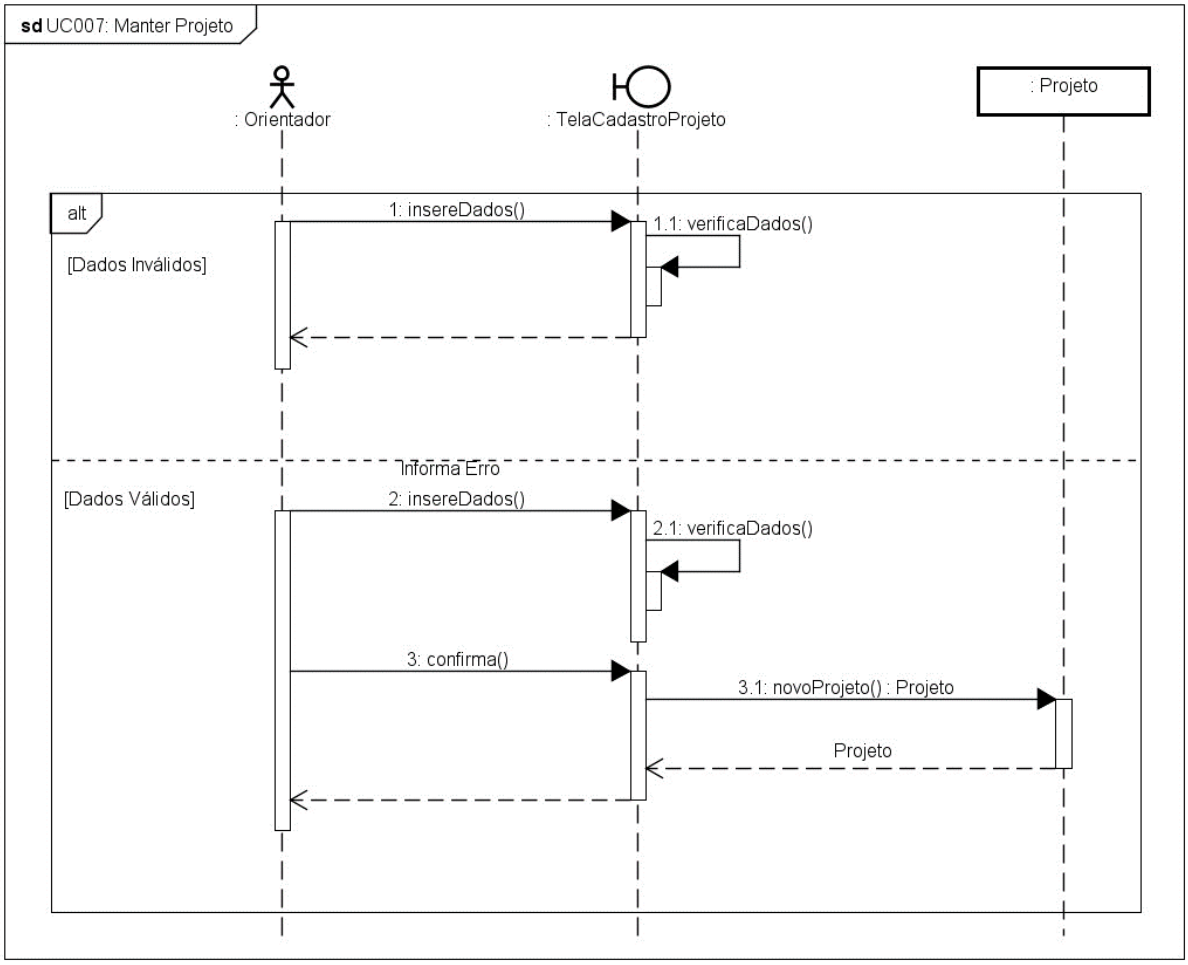
Fonte: Os Autores (2023).

FIGURA 10 – UC006: MANTER COMPLEXIDADE



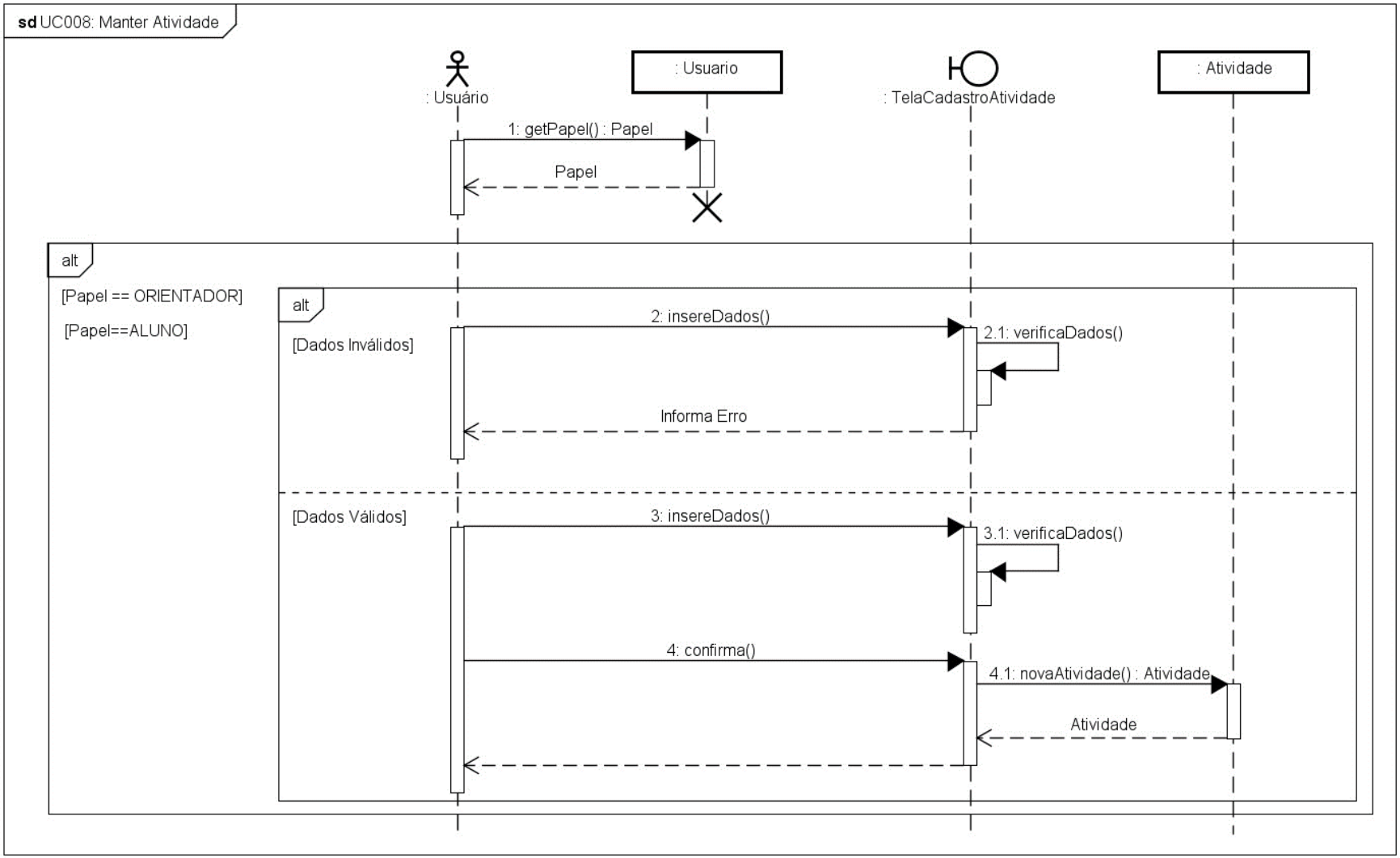
Fonte: Os Autores (2023).

FIGURA 11 – UC007: MANTER PROJETO



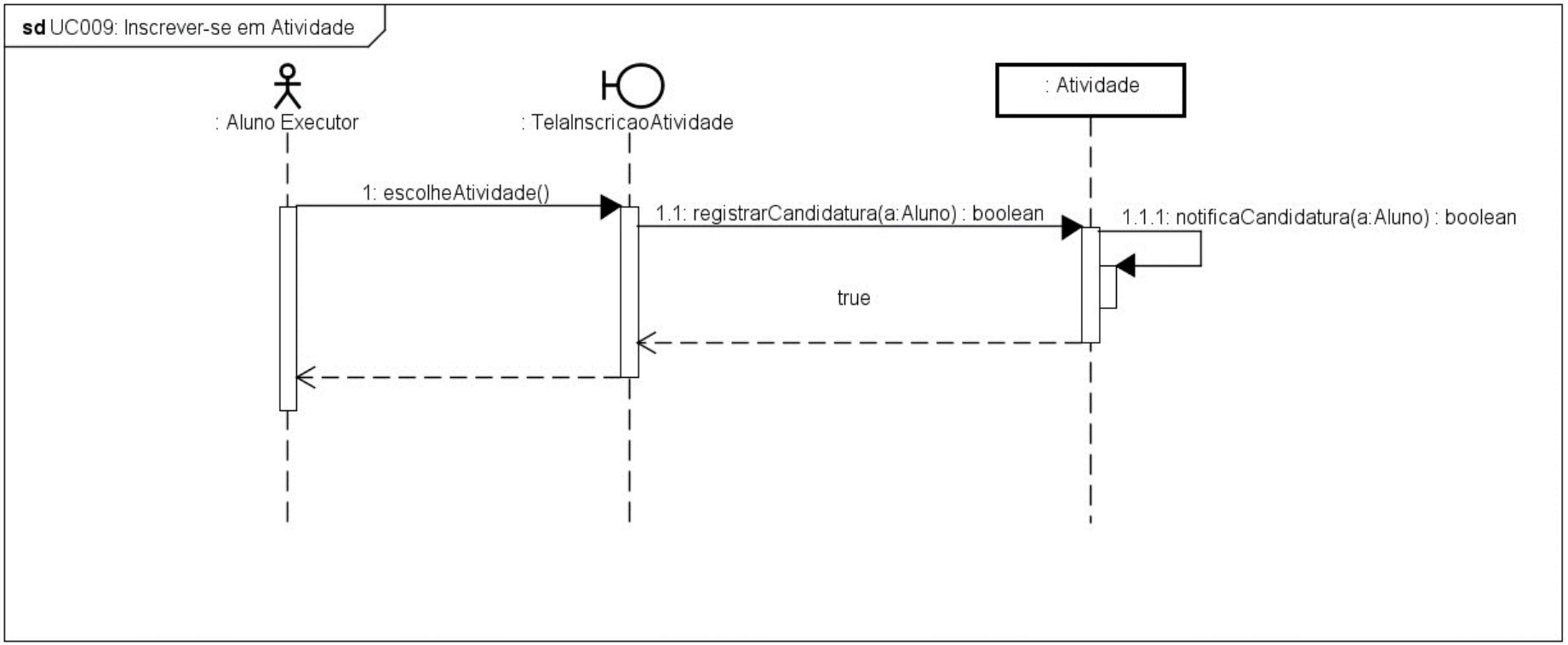
Fonte: Os Autores (2023).

FIGURA 12 – UC008: MANTER ATIVIDADE



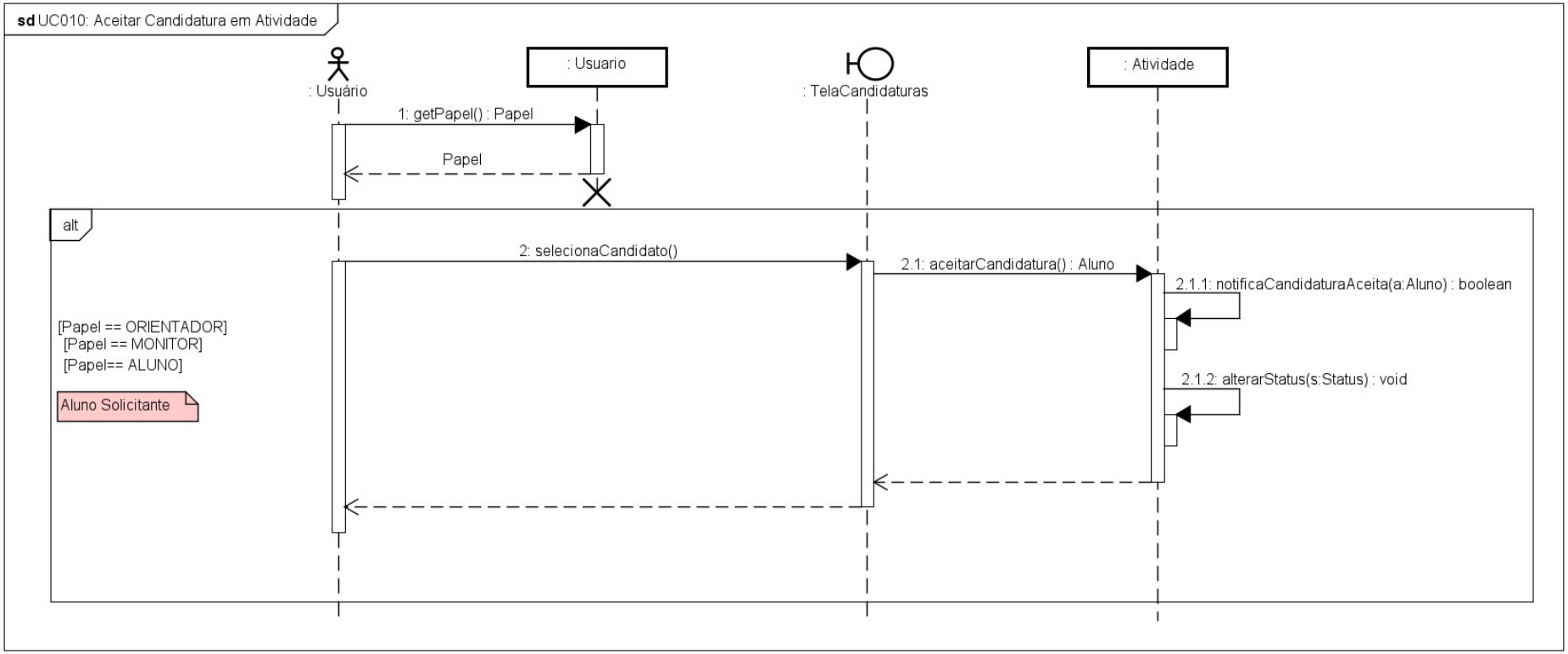
Fonte: Os Autores (2023).

FIGURA 13 – UC009: INSCREVER-SE EM ATIVIDADE



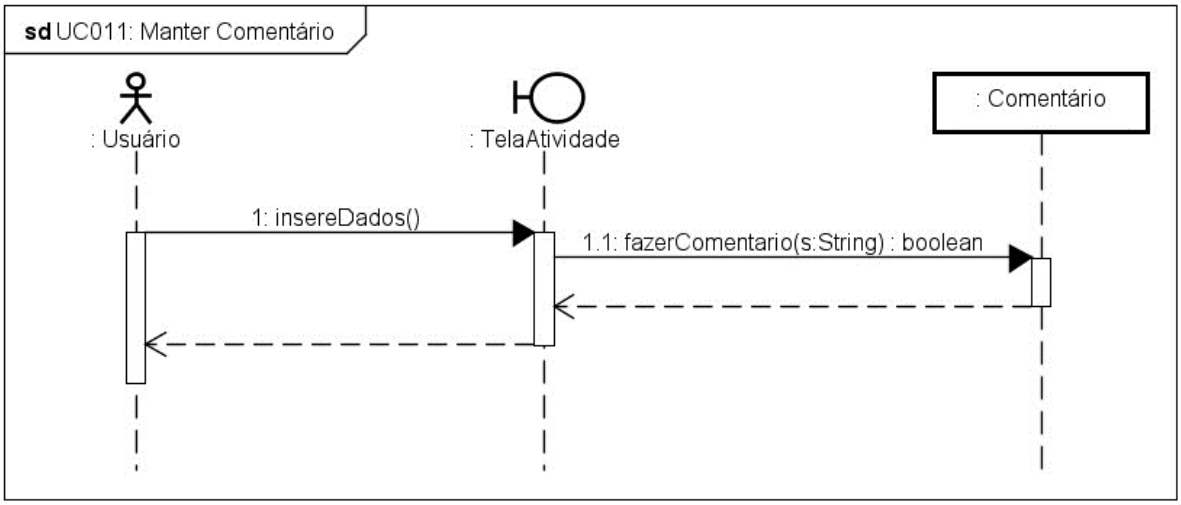
Fonte: Os Autores (2023).

FIGURA 14 – UC010: ACEITAR CANDIDATURA EM ATIVIDADE



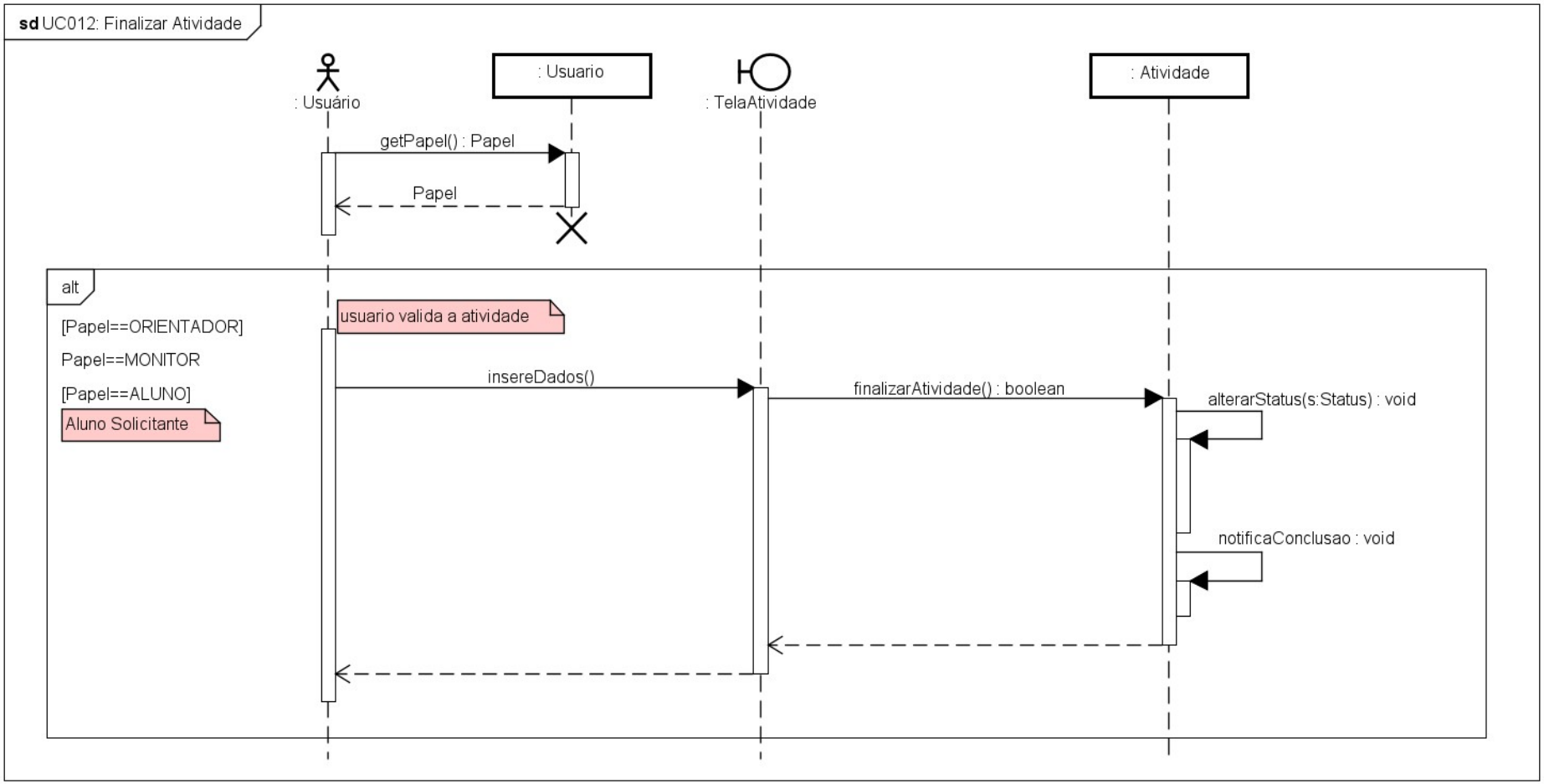
Fonte: Os Autores (2023).

FIGURA 15 – UC011: MANTER COMENTÁRIO



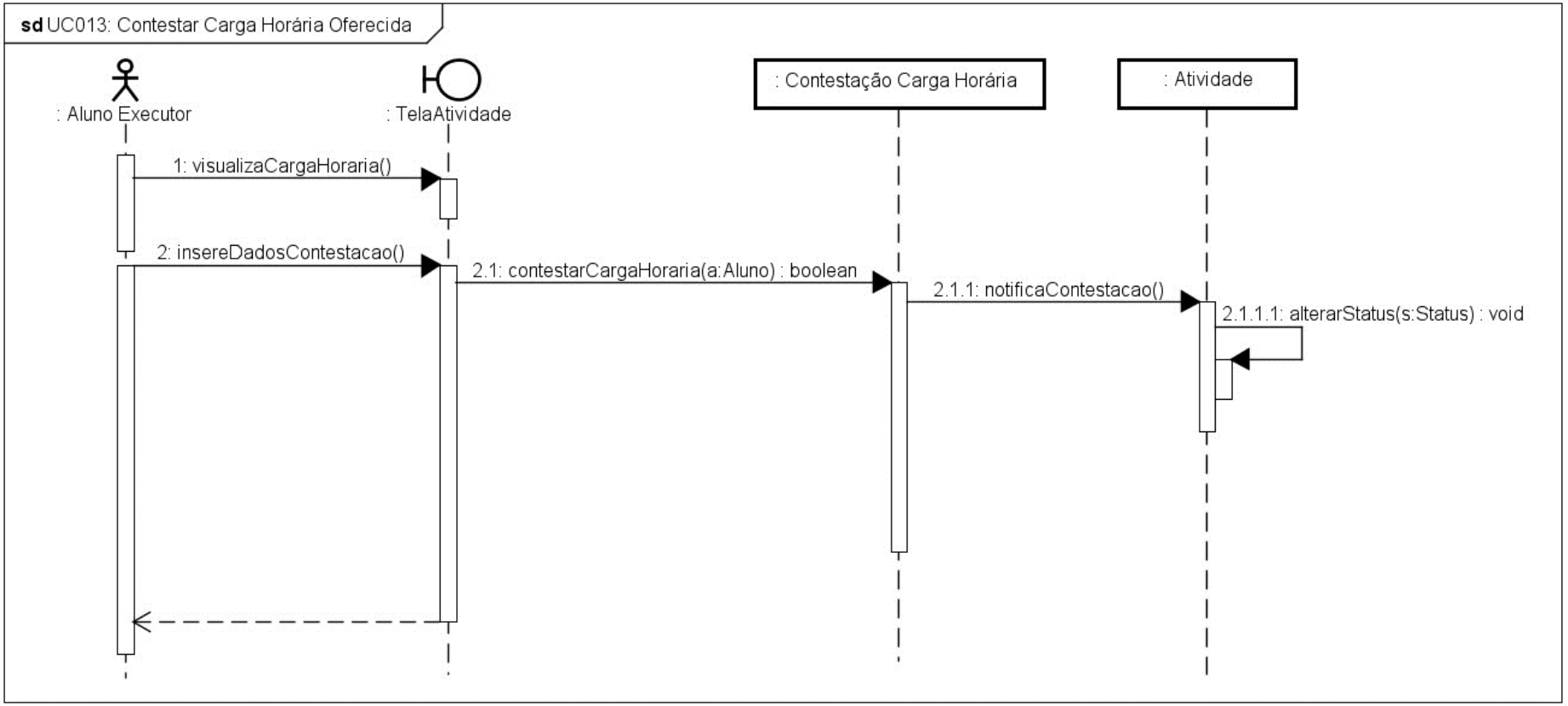
Fonte: Os Autores (2023).

FIGURA 16 – UC012: FINALIZAR ATIVIDADE



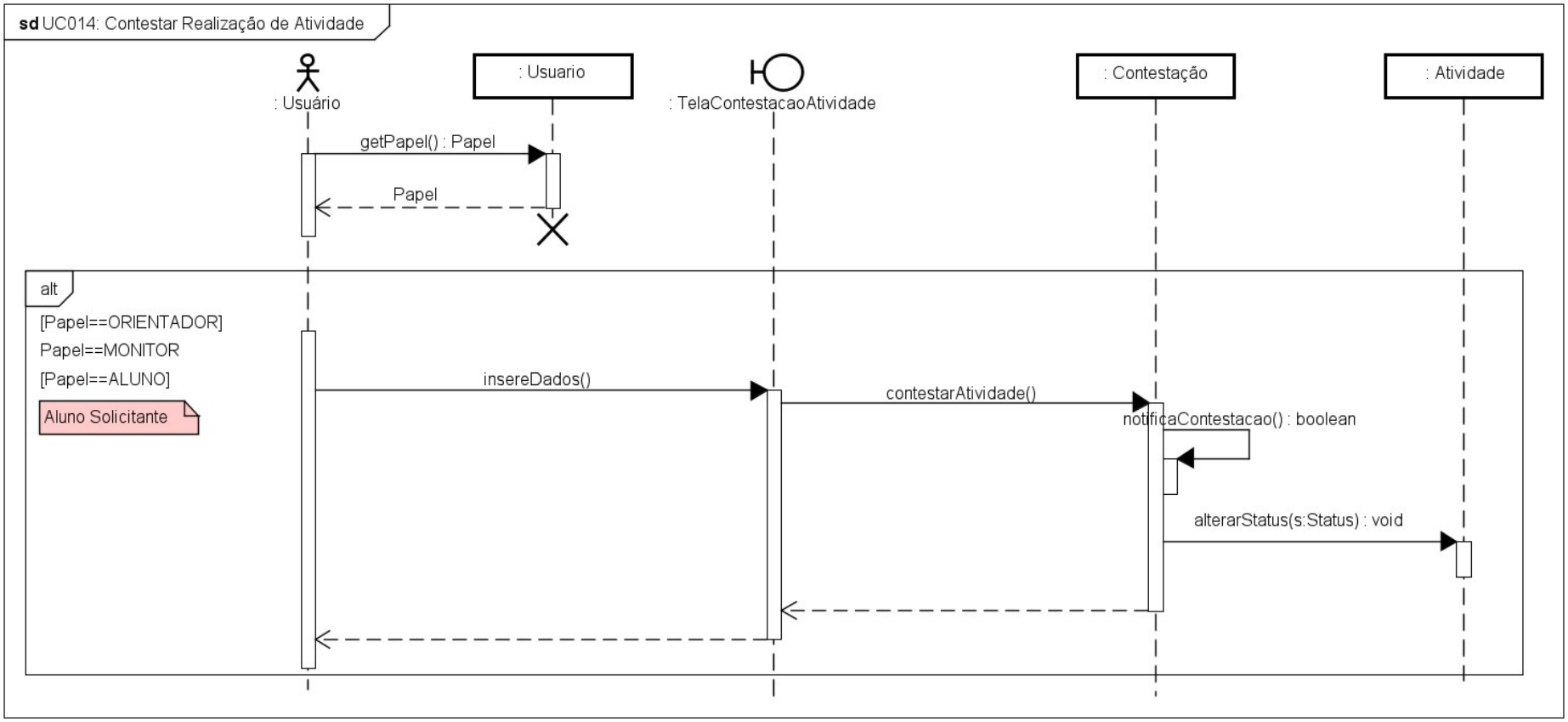
Fonte: Os Autores (2023).

FIGURA 17 – UC013: CONTESTAR CARGA HORÁRIA OFERECIDA



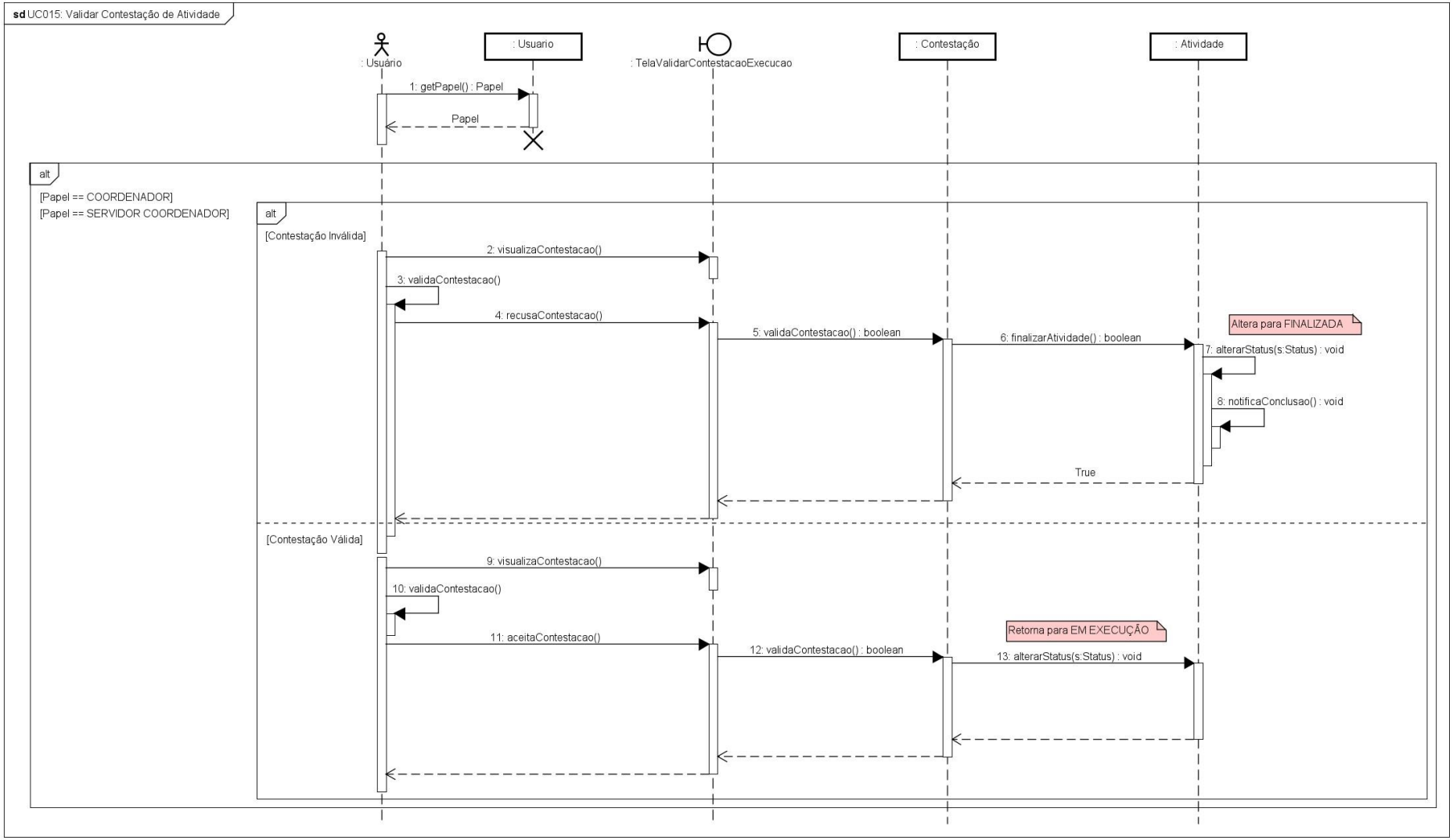
Fonte: Os Autores (2023).

FIGURA 18 – UC014: CONTESTAR REALIZAÇÃO DE ATIVIDADE



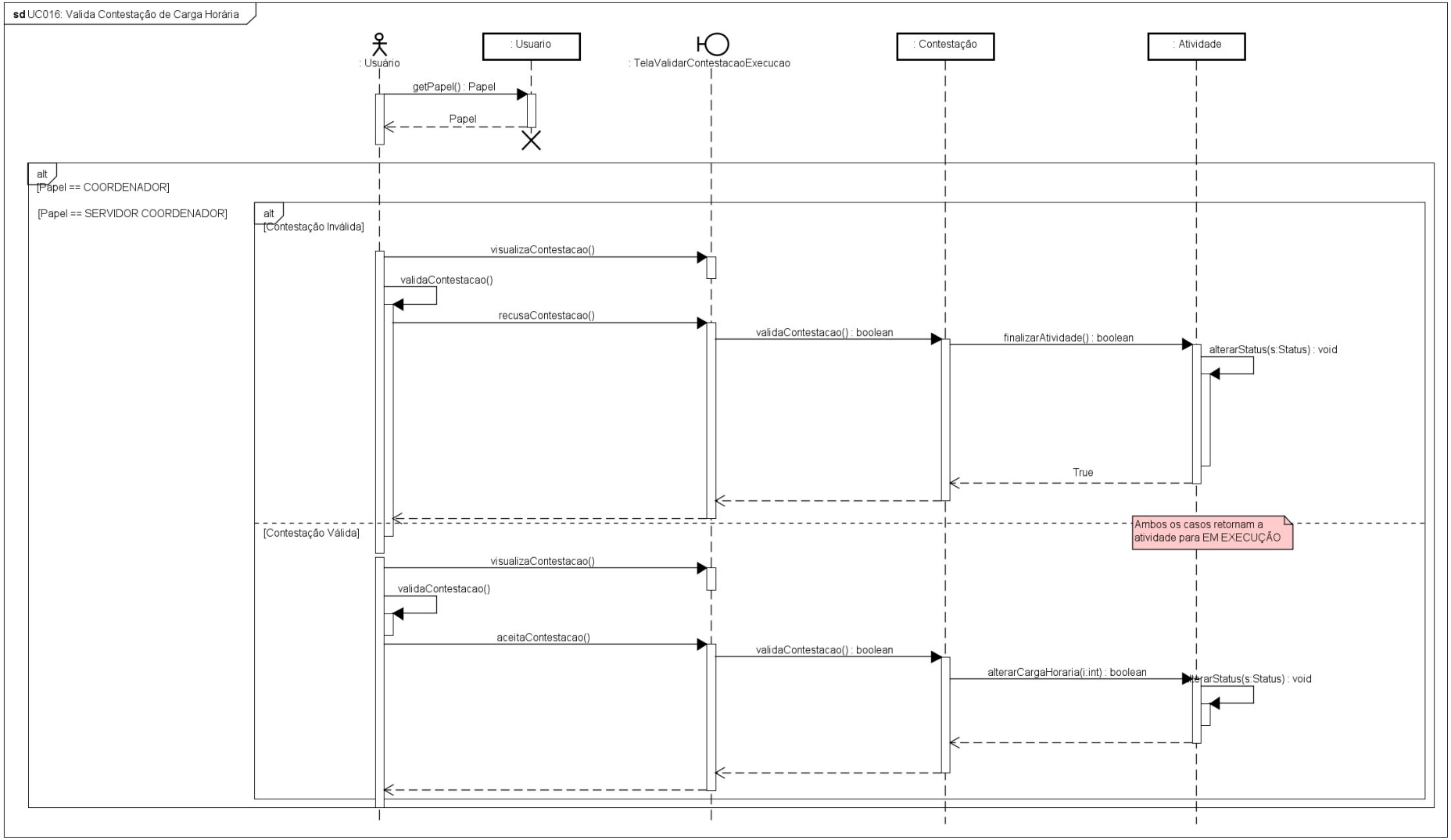
Fonte: Os Autores (2023)

FIGURA 19 – UC015: VALIDAR CONTESTAÇÃO DE ATIVIDADE



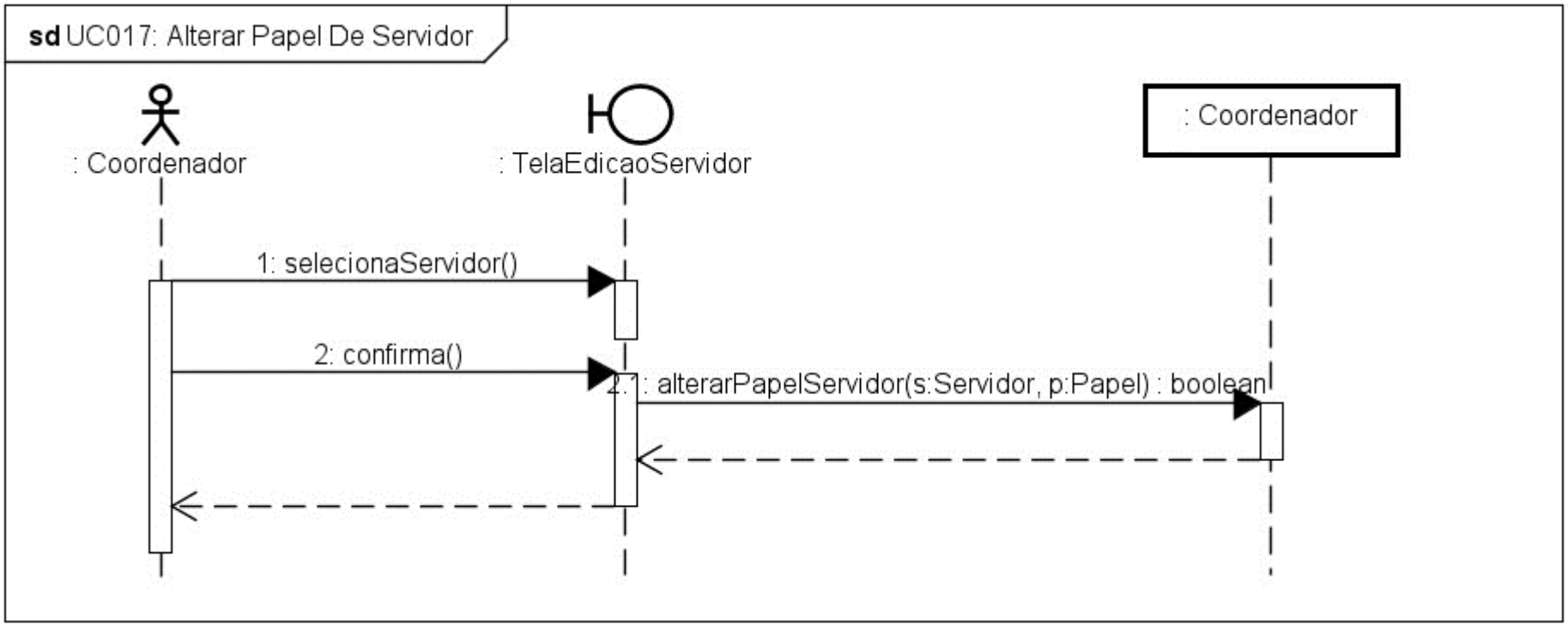
Fonte: Os Autores (2023)

FIGURA 20 – UC016: VALIDA CONTESTAÇÃO DE CARGA HORÁRIA



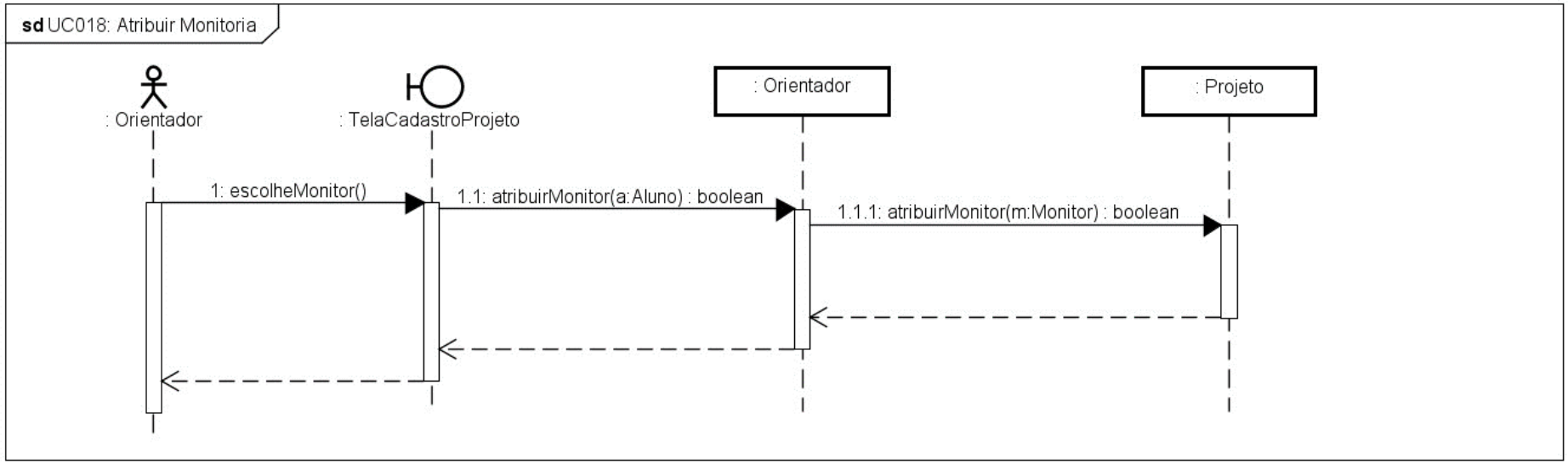
Fonte: Os Autores (2023).

FIGURA 21 – UC017: ALTERAR PAPEL DE SERVIDOR



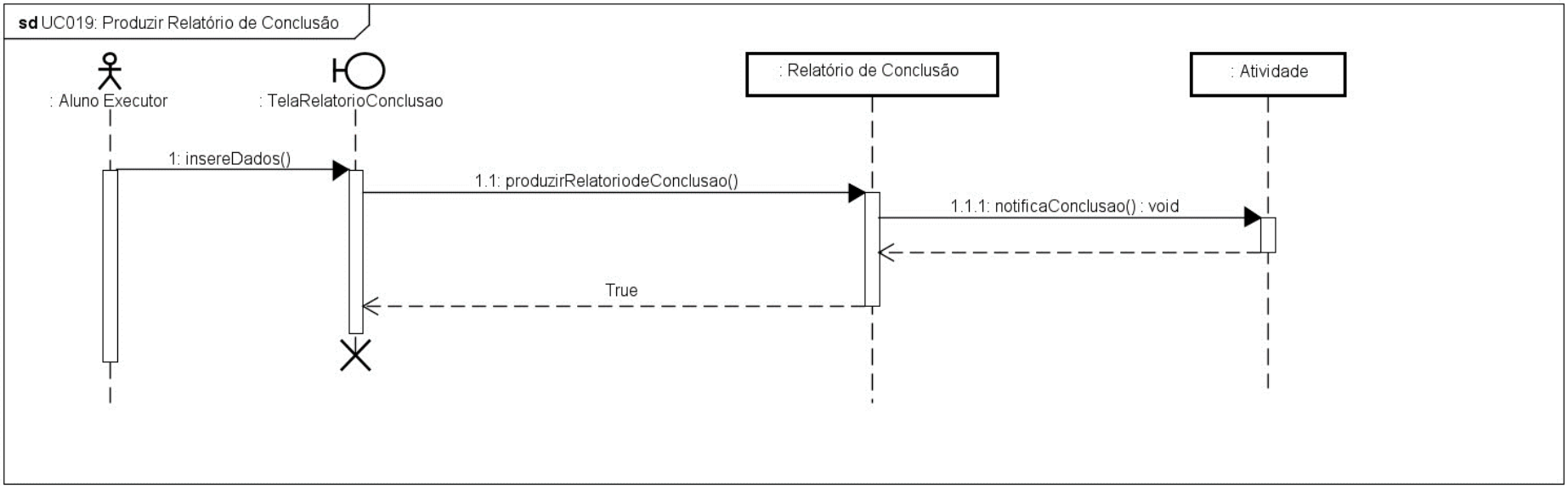
Fonte: Os Autores (2023).

FIGURA 22 – UC018: ATRIBUIR MONITORIA



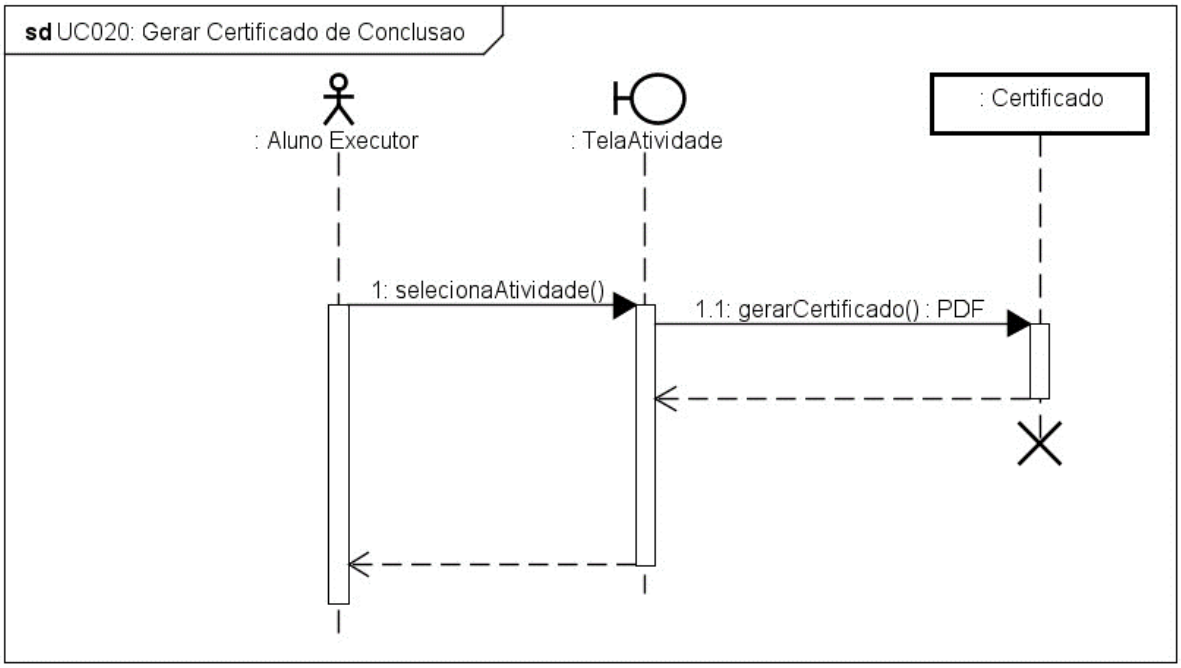
Fonte: Os Autores (2023).

FIGURA 23 – UC019: PRODUZIR RELATÓRIO DE CONCLUSÃO



Fonte: Os Autores (2023).

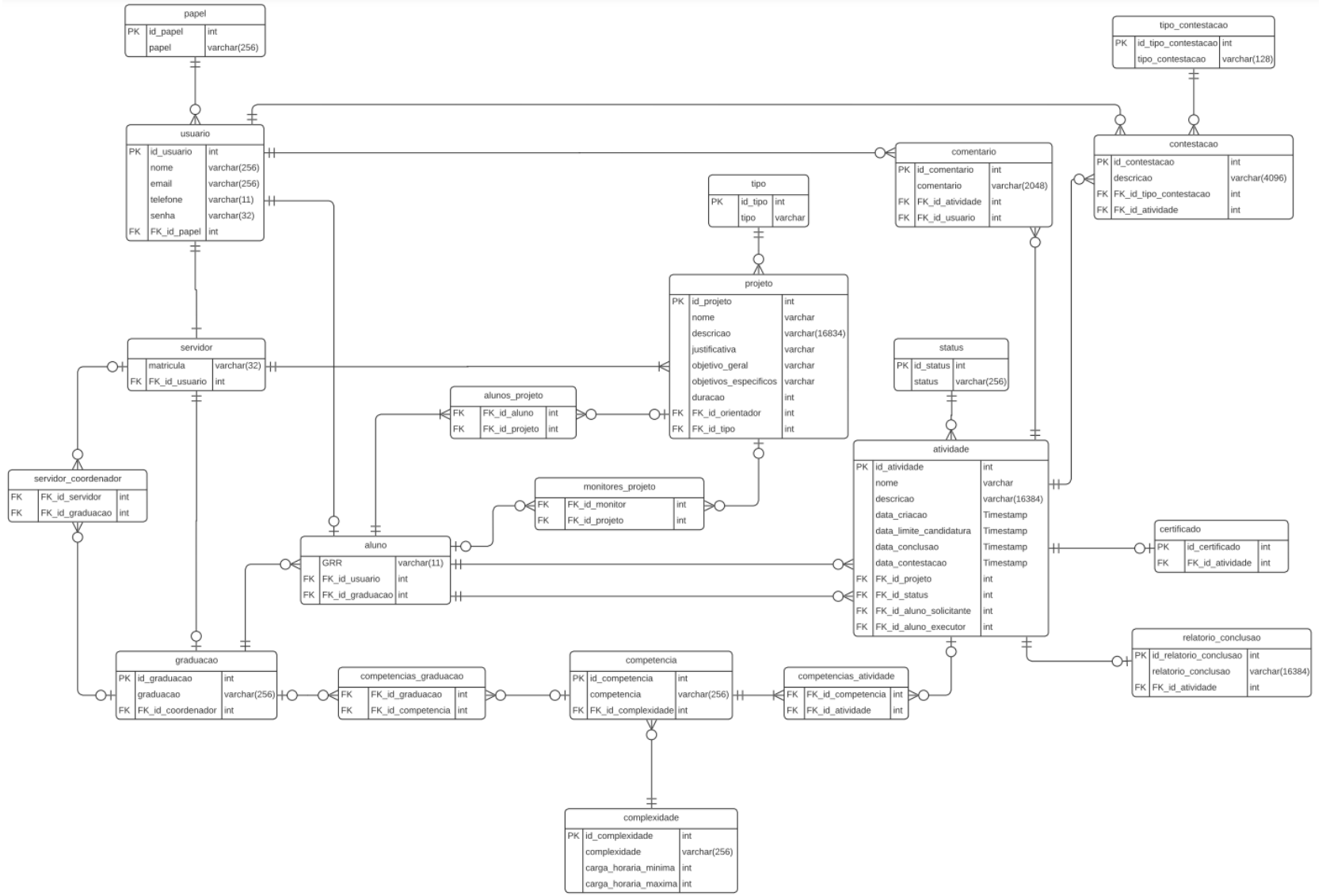
FIGURA 24 – UC020: GERAR CERTIFICADO DE CONCLUSÃO



Fonte: Os Autores (2023).

APÊNDICE F – DIAGRAMA FÍSICO DO BANCO DE DADOS

FIGURA 25 – DIAGRAMA ENTIDADE-RELACIONAMENTO (DER)



Fonte: Os Autores (2023).