UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE INFORMÁTICA

Automação do Processo de Mobilidade Erasmus *(Incoming)* no Sistema Fenix

Bernardo Salgado Andrade

Mestrado em Informática

Trabalho de Projeto orientado por:
Professora Doutora Maria Dulce Pedroso Domingos

2024

Aos meus Rs.

Agradecimentos

É curioso que este texto – escrito no final – seja colocado no início do documento. Talvez seja uma forma de mostrar que esta etapa, que agora culmina com este trabalho, não teria sequer começado sem todo o apoio que fui tendo, humanizando a obra e lembrando que, por trás de toda a dedicação e esforço, há um percurso coletivo de colaboração, sacrifício e inspiração. Milhares de horas investidas num objetivo. Uma conquista feita ao longo de anos, com muitas etapas, dúvidas e lágrimas, mas, acima de tudo, sustentada por apoio e generosidade, culminando num momento de orgulho e felicidade que agora partilho.

Escrever este texto é um exercício de reflexão sobre a jornada e uma forma de honrar todos os que dela fizeram parte. É, para mim, acima de tudo, um gesto de humildade e gratidão, que nunca estará à altura da generosidade que recebi ao longo do caminho.

Quero, por isso, começar por agradecer à minha professora e orientadora, professora Dulce Domingos, que me conheceu atrasado para a sua aula, com pouquíssimas bases e caído de "pára-quedas", mas com uma vontade imensa de aprender. Obrigado por acreditar em mim, por me dar a oportunidade de crescer consigo, de mostrar o meu valor e de concretizar este projeto no departamento que dirige enquanto Vice-Reitora. Muito obrigado, professora.

Obrigado também aos meus colegas de curso e de projeto, com quem partilhei o dia a dia, alguns dos quais se tornaram amigos e sem os quais nada teria sido o mesmo. A ti, Daniel, João e Fábio. A ti, Malvina e Marquinhos. A ti, Saúl e Gonçalo. A ti, Alice, Ivan e Tiago, meu companheiro de Fenix. O meu muito obrigado a cada um de vós! E claro, a ti, Catarina. Aprendi muito contigo e só te tenho a agradecer por tudo.

Um obrigado especial a ti, mestre Antão, que me apoiaste num dos piores momentos da minha vida e sem o qual, certamente, esta conquista não teria sido possível. Não me vou esquecer.

Wagner, meu amigo e companheiro de outras lutas, sempre determinado e profissional como poucos. Obrigado pelo teu exemplo. Seguimos juntos!

Um reconhecimento especial às minhas queridas tias Fi, Mocas e Mada, que me acompanham e ajudam há anos nas minhas batalhas. Fico muito feliz por vos continuar a deixar orgulhosas. Obrigado por tudo!

Obrigado também a ti, Mi, Rita e Sofia, por estarem aí.

Obrigado, Jimba e Marisa, meus amigos, meus irmãos. Amo-vos!

Aos meus irmãos Marcelo e Catarina, que apesar de distantes estão sempre ao meu lado e que são para mim exemplos de esforço, empenho, dedicação e superação, muito obrigado por tudo! Tenho muito orgulho em vocês. Pai, esta conquista é pela nossa lua, que assistiu ao início desta

jornada e que hoje a acompanha de um lugar tão distante, mas sempre à distância de um olhar. A falta que nos faz. Obrigado, Tia. Obrigado, Pai. Obrigado, Mãe.

Obrigado também a si, Avô, e a si, Avó. A sorte que tive em vos ter. Se aqui cheguei, foi também graças ao que sempre fizeram por mim.

Obrigado também a ti, meu mentor, que, sem me conheceres, me ajudaste generosamente desde que te desafiei a fazê-lo. Obrigado também a tantos outros que fizeram parte destes anos e que não mencionei, mas dos quais não me esqueço. Muito obrigado a todos.

Antes do último reconhecimento, uma palavra especial para si, Patrícia, que tem sido tão importante e me tem ajudado tanto. Muito obrigado.

Por fim, o reconhecimento mais importante. À minha família. A ti, Rio, e a ti, Rosairo. O que vivemos e onde chegámos nestes mais de 12 anos em que seguimos juntos. Em breve seremos mais, sempre juntos, para sempre. Nada disto faria sentido sem ti, e nada teria sido possível sem ti, sem o teu apoio incondicional e, sobretudo, sem o teu amor. Obrigado, meu amor.

Venha o próximo!

Resumo

O Fenix, um Sistema Integrado de Gestão Académica (SIGA) inicialmente criado pelo Instituto Superior Técnico (IST), é utilizado nas Escolas da Universidade de Lisboa (ULisboa), estando em constante desenvolvimento, evolução e melhoria. Este projeto dedica-se a estender o módulo de Mobilidade existente no Fenix, com o objetivo de desmaterializar e automatizar os processos de entrada de estudantes Erasmus. Ao alavancar a infraestrutura do Fenix, o objetivo é alinhar a gestão do programa de mobilidade com a iniciativa *Erasmus Without Paper* (EWP), melhorando os processos administrativos da Instituição para a mesma e para os estudantes da Universidade de Lisboa (ULisboa).

Apesar do processo de mobilidade ter uma estrutura macro definida pelo programa Erasmus e de esta ser aplicada na ULisboa, as Escolas que a compõe, possuem especificidades próprias. Assim, um dos principais desafios deste projeto é criar uma solução que respeite a estrutura geral do processo, mas permita personalizações de acordo com as necessidades particulares de cada instituição. Para atingir esse objetivo, utilizou-se a ferramenta de modelação de processos do Fenix, garantindo que o processo possa ser adaptado pelas próprias, por forma a refletir as suas exigências específicas, assegurando uma integração que respeite as necessidades locais.

Palavras-chave: Mobilidade, Erasmus, *Incoming*, Fenix, Automação

Abstract

Fenix, an Academic Management Integrated System initially crafted by Instituto Superior Técnico, is used across multiple Schools within the University of Lisbon (ULisboa), constantly being developed, evolving and improving. This project is dedicated to further extend the existing Mobility module in Fenix, with a focus on dematerializing and automating processes for incoming Erasmus students. By leveraging Fenix's infrastructure, the goal is to align mobility program management with Erasmus Without Paper (EWP) iniciative, enhancing the administrative processes of the Institution for it and the incoming students within the ULisboa.

Although the mobility process has a macro structure defined by the Erasmus program and is applied at ULisboa, its Schools have their own specificities. Thus, one of the main challenges of this project is to create a solution that respects the general structure of the process, but allows customization according to the particular needs of each institution. In order to achieve this goal, the Fenix process modeling tool was used, ensuring that the process can be adapted by the School itself in order to reflect their specific requirements, ensuring integration that respects local needs.

Keywords: Mobility, Erasmus, Incoming, Fenix, Automation

Conteúdo

Lista de Figuras **xvii**

1 Introdução **1**
1.1 Contexto . . . 1
1.2 Problema . . . 1
1.3 Motivação . . . 2
1.4 Objetivos . . . 2
1.5 Planeamento . . . 3
1.6 Estrutura . . . 4

2 Trabalho relacionado **7**
2.1 Sistema Integrado de Gestão Académica (SIGA) Fenix . . . 7
2.2 Contexto Histórico . . . 8
2.3 Arquitetura . . . 9
2.4 Ferramentas e Tecnologias . . . 10
2.5 *Frameworks* . . . 11
2.6 Linguagens de Modelação . . . 12
2.6.1 Linguagens de Modelação de Domínio . . . 12
2.6.2 Linguagem de Modelação da Camada de Apresentação . . . 15
2.7 *OmnisBar* . . . 17
2.8 *Workflows* . . . 17
2.9 Documentos *Online* . . . 19
2.10 Sumário . . . 20

3 Requisitos e Desenho **21**
3.1 Levantamento de Requisitos . . . 21
3.1.1 Conceitos-Chave sobre Mobilidade . . . 21
3.1.2 *Erasmus Without Paper* (EWP) . . . 22
3.1.3 Processo de Mobilidade Erasmus+ . . . 24
3.1.4 Processo de Mobilidade *Incoming* . . . 25
3.2 Análise de Requisitos . . . 26
3.2.1 Intervenientes . . . 26

3.2.2 Requisitos Funcionais . . . 26
3.2.3 Requisitos Não Funcionais . . . 28
3.3 Arquitetura . . . 28
3.4 Desenho . . . 29
3.4.1 Modelação do Processo de Mobilidade *Incoming*: BPMN . . . 30
3.4.2 Comunicações EWP para Nomeações . . . 31
3.4.3 Diagrama de Entidades e Associações (DEA) . . . 33
3.5 Sumário . . . 36

4 Implementação e Avaliação **37**
4.1 Implementação . . . 37
4.1.1 Ambiente de Desenvolvimento . . . 37
4.1.2 Domínio . . . 38
4.1.3 Interface . . . 41
4.1.4 Modelação do Processo de Mobilidade *Incoming*: *Workflow* Fenix . . . 47
4.1.5 Operações para Integração com o EWP . . . 53
4.1.6 Geração de Documentos . . . 55
4.1.7 Perfis de Acesso . . . 57
4.2 Avaliação . . . 60
4.2.1 Método de Avaliação . . . 60
4.2.2 Avaliação da Usabilidade . . . 61
4.2.3 Análise de Resultados . . . 61
4.3 Sumário . . . 67

5 Conclusão e Trabalho Futuro **69**

Abreviaturas **73**

Bibliografia **77**

Anexos **78**

A Macroprocesso de Mobilidade Erasmus **79**
A.1 Macroprocesso de Mobilidade Erasmus+ . . . 79

B Processo de Mobilidade Erasmus *Incoming* **81**
B.1 Faculdade de Ciências (FC) . . . 81
B.2 Instituto Superior de Economia e Gestão (ISEG) . . . 84
B.3 Faculdade de Letras (FL) . . . 87
B.4 Instituto Superior de Ciências Sociais e Políticas (ISCSP) . . . 90

C Mobilidade *Incoming* ULisboa **93**
C.1 Desenho do Processo de Mobilidade Erasmus *Incoming* ULisboa 93

D *Workflow* Fenix - Mobilidade *Incoming* **99**
D.1 Detalhe Workflow Fenix - Mobilidade *Incoming* 99
D.2 Documentação do Workflow Fenix - Mobilidade *Incoming* 105

E Modelação de Domínio **111**
E.1 Domínio do Módulo de Mobilidade Fenix 111
E.2 Mapeamento de Entidades e Atributos 113

F Carta de Aceitação **115**
F.1 Exemplo de proposta para *Carta de Aceitação* 115

Lista de Figuras

2.1 Excerto da DSL desenvolvida no âmbito do *bootcamp* realizado. 13
2.2 Eclipse IDE: Representação de excerto da DSL e da classe *Teacher* e dos respetivos métodos gerados após a compilação. 14
2.3 Excerto da PSL desenvolvida no âmbito do *bootcamp* realizado. 16
2.4 Conjunto de elementos da linguagem de modelação de *workflows* da *framework* Omnis [16]. 19

3.1 Diagrama de Rede EWP com a ULisboa - fornecido pelo NDS. 29
3.2 Interpretação das Comunicações de Nomeações definidas pelo EWP. 32
3.3 DEA Mobilidade *Incoming* - versão inicial. 34
3.4 DEA Mobilidade *Incoming* - versão atual. 36

4.1 Excerto da DSL do módulo de Mobilidade. 39
4.2 Excerto da Classe *UlMobInStudentNomination*. 40
4.3 Modelo de Classes de Domínio da Mobilidade *Incoming* - versão atual. 41
4.4 Vista Módulo Mobilidade: Menu *Incoming*. 42
4.5 Execuções de Nomeações: Detalhe do tipo *Estudos*. 42
4.6 Excerto da classe do ecrã *SearchScreen*: *Gestão de Nomeações*. 43
4.7 Excerto da PSL *Incoming Student Nomination*. 44
4.8 Exemplo de Operações para Aceitar ou Rejeitar nomeação, dentro de uma instância de *workflow*. 45
4.9 Exemplo de Operação para gerar documento online: Carta de Aceitação, dentro de uma instância de *workflow*. 46
4.10 Excerto dos separadores do processo de Mobilidade *Incoming*. 47
4.11 Início do *Workflow* de Mobilidade *Incoming*. 48
4.12 Detalhe do separador - *Monitorização de Processo*. 49
4.13 Detalhe do separador - *Dados EWP Nomeação Incoming*. 50
4.14 Detalhe do separador - *Dados do Incoming Student*. 51
4.15 Detalhe do separador - *Comentário da ID (via EWP)*. 52
4.16 Detalhe estado inicial do *workflow* - Validação de Nomeação. 52
4.17 Excerto da classe *UlMobInAcceptNominationOperation.java* relativa à operação de *Aceitar Nomeação Incoming*. 54

4.18 Excerto da classe *UlMobInUpdateRejectedNominationStatusOperation.java* relativa à operação de *Atualizar Nomeação Rejeitada*. 55
4.19 Menu Fenix *Editar Conteúdo da Template*: Carta de Aceitação. 56
4.20 Controlo de acessos. 58
4.21 Perfis de Acesso: Mobilidade. 59
4.22 Excerto do validador desenvolbido para o Aluno *Incoming*. 59
4.23 *Communication Logs* após importar nomeações do nó EWP. 62
4.24 *Communication Logs*: Exemplo de uma nomeação corretamente importada - ID 4334. 63
4.25 *Communication Logs*: *BODY* da nomeação corretamente importada - ID 4334. . 64
4.26 *Communication Logs*: *POST* de CNR - ID 4338. 65
4.27 *Communication Logs*: CNR de confirmação de nomeações *test1* e *test2* corretamente importadas - ID 4338. 65
4.28 *Communication Logs*: *POST* de CNR relativo à Aceitação da nomeação *test1* - ID 4377. 66
4.29 *Communication Logs*: *POST* de CNR relativo à Rejeição da nomeação *test2* - ID 4381. 67

A.1 Macroprocesso de Mobilidade Erasmus+ Parte 1. 79
A.2 Macroprocesso de Mobilidade Erasmus+ Parte 2. 80

B.1 Processo de Mobilidade Erasmus *Incoming* - FC. 83
B.2 Processo de Mobilidade Erasmus *Incoming* - ISEG. 86
B.3 Processo de Mobilidade Erasmus *Incoming* - FL. 89
B.4 Processo de Mobilidade Erasmus *Incoming* - ISCSP. 92

C.1 Proposta de Processo de Mobilidade Erasmus *Incoming* ULisboa - Parte 1. 93
C.2 Proposta de Processo de Mobilidade Erasmus *Incoming* ULisboa - Parte 2. 94
C.3 Proposta de Processo de Mobilidade Erasmus *Incoming* ULisboa - Parte 3. 95
C.4 Proposta de Processo de Mobilidade Erasmus *Incoming* ULisboa - Parte 4. 96
C.5 Proposta de Processo de Mobilidade Erasmus *Incoming* ULisboa - Parte 5. 97

D.1 *Workflow* de Mobilidade *Incoming* - Parte 1. 100
D.2 *Workflow* de Mobilidade *Incoming* - Parte 2. 101
D.3 *Workflow* de Mobilidade *Incoming* - Parte 3. 102
D.4 *Workflow* de Mobilidade *Incoming* - Parte 4. 103
D.5 *Workflow* de Mobilidade *Incoming* - Parte 5. 104
D.6 Excerto da Documentação do *Workflow* Fenix - Mobilidade *Incoming* - 1. 106
D.7 Excerto da Documentação do *Workflow* Fenix - Mobilidade *Incoming* - 2. 107
D.8 Excerto da Documentação do *Workflow* Fenix - Mobilidade *Incoming* - 3. 108
D.9 Excerto da Documentação do *Workflow* Fenix - Mobilidade *Incoming* - 4. 109

E.1 Modelo de Domínio do Módulo de Mobilidade Fenix - fornecido pelo NDS. . . . 111
E.2 Representação Mapeamento de Entidades e Atributos - API EWP *Incoming* V2. . 113
E.3 Representação do Mapeamento de Entidades e Atributos - API EWP *Outgoing* V3. 114

Capítulo 1

Introdução

Neste primeiro capítulo, apresentamos uma visão global do projeto, abordando o contexto em que se insere, o problema identificado e os objetivos a atingir. Exploramos a motivação subjacente a este trabalho, justificando a necessidade de enfrentar os desafios atuais nos processos de mobilidade, particularmente na mobilidade *incoming*, ou seja, no que respeita aos alunos que vêm para a Universidade de Lisboa (ULisboa). Adicionalmente, fornecemos uma visão geral do planeamento que guiará o desenvolvimento do projeto. Por último, descrevemos a estrutura do documento, oferecendo uma perspetiva geral do que será discutido em cada capítulo.

1.1 Contexto

O Sistema Integrado de Gestão Académica (SIGA) Fenix é uma plataforma para as dezoito Escolas da ULisboa, desde o ano letivo 2020/2021. Este sistema desempenha um papel na facilitação de diversos processos educativos e administrativos e será detalhado no próximo capítulo. Tal como as necessidades do dia-a-dia, é um sistema vivo e dinâmico em constante evolução, pelo que, existe a permanente necessidade de expandir as suas funcionalidades para atender às diversas exigências das Escolas.

Entre essas funcionalidades, destaca-se a gestão dos processos de mobilidade, que atualmente é ainda feita de forma manual ou através de fluxos de inscrições e candidaturas, não desenhados para o efeito. É nesse contexto que surge este projeto, que visa particularmente concentrar esforços na parte da mobilidade Erasmus *incoming*, mais especificamente, na gestão processual de acolhimento de alunos que vêm estudar ou estagiar nas Escolas da ULisboa, automatizando tanto quanto possível o processo e libertando os recursos humanos para outras tarefas.

1.2 Problema

No contexto dos processos de mobilidade na ULisboa, identificamos um desafio que impacta estudantes e recursos humanos administrativos responsáveis pela mobilidade *incoming*. O problema surge quando os estudantes, previamente nomeados por instituições parceiras das Escolas da ULisboa, ou os próprios núcleos de mobilidade dessas Escolas, são obrigados a introduzir manualmente

no SIGA Fenix informações que já foram fornecidas através da nomeação ou do *Learning Agreement* (LA). Esta duplicação de esforços não apenas frustra os estudantes, como também implica um desperdício significativo de tempo para os recursos humanos administrativos, que são obrigados a validar novamente dados já confirmados.

1.3 Motivação

Este projecto surge da necessidade de resolver o problema identificado que afeta diretamente as equipas operacionais e a experiência dos estudantes. No ano letivo de 2022/2023, mais de 700 alunos estiveram envolvidos em mobilidade *incoming* em apenas quatro das dezoito Escolas da ULisboa[1] e, diante deste volume considerável, é crucial melhorar os processos.

A nossa abordagem foca-se especificamente na automação dos processos de mobilidade *incoming* de alunos vindos de outras Universidades para a ULisboa - tanto no âmbito de estudos (*Student Mobility for Studies* (SMS)) como em estágios/projetos (*Student Mobility for Placements* (SMP)). Atualmente, estes processos, não decorrem em módulo próprio no entanto, deverão passar para o módulo da Mobilidade no SIGA Fenix e deverão permitir a ligação ao projeto *Erasmus Without Paper* (EWP). Ao integrar diretamente informações provenientes do EWP, quer de nomeações ou do próprio LA do aluno, procuramos melhorar o processo, reduzindo a necessidade de entrada e/ou validação manual de dados. Esta abordagem visa então automatizar os processos, eliminando a redundância na fonte e contribuindo para uma melhoria da gestão dos processos académicos, num volume anual considerável.

1.4 Objetivos

Nesta secção são identificados os objetivos gerais e específicos deste projeto de automação do processo de mobilidade Erasmus *incoming* no sistema Fenix.

Objetivos Gerais

Começando pelos objetivos gerais do projeto, temos dois a destacar:

1. Desenvolver o processo de mobilidade *incoming* do módulo de mobilidade no SIGA Fenix, tornando-o capaz de processar automaticamente informações de fontes como nomeações e/ou LA, recebidos através da rede EWP.

2. Criar estados Fenix[2] que permitam compor o processo de mobilidade Erasmus *incoming* no sistema Fenix, por forma a que as Escolas que compõe a ULisboa o possam usar, adaptando se necessário, à sua realidade.

[1] Dados obtidos nas reuniões de levantamento de requisitos.
[2] Detalhado na secção 2.8.

Objetivos Específicos

Quanto aos objectivos específicos deste projeto, identificamos os seguintes:

1. Desenhar e implementar uma integração com o projeto EWP, utilizando informações provenientes da sua *Application Programming Interface* (API).

2. Desenvolver mecanismos de automação para a inserção de dados do aluno a partir de informação recebida pelo EWP, reduzindo a necessidade de intervenção manual.

3. Utilizar a ferramenta de *workflows*[3] do SIGA Fenix para automatizar o fluxo de trabalho do processo de mobilidade *incoming*.

4. Priorizar a experiência do utilizador (aluno ou funcionário da ULisboa), garantindo um processo de mobilidade mais fluído, evitando a reintrodução e/ou validação manual de informações.

Em resumo, estes objetivos pretendem enfrentar os desafios identificados, a automação do processo atual, proporcionar flexibilidade para adaptações locais e integração com as iniciativas já existentes, nomeadamente o EWP.

1.5 Planeamento

O desenvolvimento deste projeto seguiu um plano organizado em seis etapas, cada uma com objetivos específicos, detalhados na lista a seguir:

1. **Integração na empresa e formação nas ferramentas de desenvolvimento e modelação do sistema Fenix:**

    - Formação interna com as várias equipas e núcleos do Departamento de Informática (DI) dos Serviços Centrais da Reitoria da Universidade de Lisboa (SCULisboa).
    - Realização de um *bootcamp* com a empresa parceira Quorum Born IT (qubIT) para aprender a utilizar as ferramentas[4] de desenvolvimento e modelação no contexto do SIGA Fenix.
    - Duração prevista: 7 semanas.

2. **Levantamento e análise de requisitos:**

    - Levantamento de requisitos para o cumprimento dos objetivos do projeto, junto de quatro Escolas selecionadas: Faculdade de Letras (FL), Faculdade de Ciências (FC), Instituto Superior de Economia e Gestão (ISEG) e Instituto Superior de Ciências Sociais e Políticas (ISCSP).

[3] Detalhado na secção 2.8.
[4] Os detalhes sobre essas ferramentas estão apresentados no capítulo 2 deste documento.

- Análise dos requisitos funcionais e não funcionais a serem considerados no desenvolvimento de um processo, transversal às Escolas da ULisboa.
- Duração prevista: 4 semanas.

3. **Desenho da solução:**
   - Com base na análise de requisitos, elaboração de uma solução alinhada com esses requisitos, transversal e adaptável às Escolas da ULisboa.
   - Duração prevista: 4 semanas.
4. **Desenvolvimento do módulo:**
   - Implementação das novas funcionalidades para o módulo de Mobilidade no SIGA Fenix, alinhadas com a solução proposta anteriormente.
   - Duração prevista: 10 semanas.
5. **Testes e validação:**
   - Realização de testes para verificar o funcionamento das novas funcionalidades desenvolvidas.
   - Após a conclusão dos testes, validação do trabalho realizado.
   - Duração prevista: 4 semanas.
6. **Entrada em produção:**
   - Disponibilização do módulo de Mobilidade com as funcionalidades recém-desenvolvidas, nas quatro Escolas iniciais, para entrada em produção.
   - Duração prevista: 3 semanas.

Apesar de não ter sido referida, está também contemplada uma etapa transversal a todo o projeto, cuja duração total prevista é de 4 semanas, e que é dedicada à escrita do relatório preliminar e da dissertação. Esta fase decorre em paralelo e é onde serão consolidadas as informações recolhidas até ao momento e delineado o plano detalhado para o desenvolvimento das próximas fases do projeto.

1.6 Estrutura

Este documento tem a seguinte estrutura:

1. **Introdução:**
   Apresenta a motivação do projeto, objetivos, contribuições, planeamento e a estrutura do documento.

2. **Trabalho Relacionado:**
   Aborda o sistema Fenix, começando com um contexto histórico, passando para a arquitetura, ferramentas e tecnologias utilizadas, *frameworks* existentes e, linguagens de modelação disponíveis tanto para o domínio, como para a camada de apresentação. De seguida apresenta a *OmnisBar* e depois os *workflows*, terminando com os documentos *online*. Em suma, é um capítulo que aborda as ferramentas relevantes para o desenvolvimento do projeto.

3. **Requisitos e Desenho:**
   Detalha o levantamento e análise dos requisitos funcionais e não funcionais com base na informação e conhecimento interno, incluindo as reuniões com quatro Escolas selecionadas. Em seguida, apresenta a arquitetura, a modelação do processo de mobilidade *incoming* em BPMN, as comunicações definidas pelo EWP e o Diagrama de Entidades e Associações (DEA).

4. **Implementação e Avaliação:**
   Descreve os procedimentos de implementação da solução para este projeto, com ênfase na interface e nas funcionalidades do sistema Fenix. A segunda parte deste capítulo detalha o método de avaliação, incluindo as limitações encontradas e os resultados obtidos.

5. **Conclusão e Trabalho Futuro**:
   Apresenta as conclusões do trabalho desenvolvido e uma síntese das atividades que poderão ser desenvolvidas como continuidade deste projeto.

Capítulo 2

Trabalho relacionado

Neste capítulo, iniciamos a exploração do SIGA Fenix, começando pelo seu contexto histórico e avançando para a sua arquitetura. De seguida, identificamos as ferramentas, tecnologias essenciais para o seu funcionamento, bem como as *frameworks* existentes. Durante esta análise, destacamos também as principais funcionalidades disponibilizadas pela *OmnisBar* e pela ferramenta de gestão de *workflows* da *framework* Omnis. Concluímos com um resumo das capacidades da ferramenta de criação de *templates* de documentos *online*. Este capítulo proporciona assim uma visão abrangente do ambiente em que o SIGA Fenix opera, destacando as ferramentas existentes e fundamentais para o seu bom desempenho.

2.1 Sistema Integrado de Gestão Académica (SIGA) Fenix

O SIGA Fenix é uma plataforma integrada que dá suporte aos processos académicos, e que implementa 3 principais sistemas de gestão: um Sistema de Gestão de Conteúdos (SGC), que pode ser utilizado a nível de curso, grau, departamento ou instituição, um Sistema de Gestão de Estudantes (SGE) focado nos alunos, e um Sistema de Gestão de Aprendizagem (SGA). O Fenix integra ainda os componentes exigidos por uma plataforma de gestão académica padrão, nomeadamente a gestão e o apoio de tarefas académicas, como por exemplo, os processos de candidatura e admissão de estudantes *online*, a matrícula e o registo *online*, a avaliação e os registos de notas, o planeamento, a conceção e a aprovação de diplomas, entre outros.

Este sistema fornece também um importante apoio aos serviços administrativos e de gestão académica, incluindo os *workflows* exigidos pela maioria dos processos. Nestes incluem-se, por exemplo, a conceção, o planeamento e a aprovação de cursos e disciplinas (incluindo conteúdos, bibliografia, estrutura e planeamento), a validação do Sistema Europeu de Transferência e Acumulação de Créditos (ECTS), a aprovação formal pelo conselho científico de disciplinas, cursos e pessoal docente, a emissão de diplomas, certificados e comprovativos, entre outros[1].

[1] Disponível em: `https://docplayer.net/6319893-The-fenixedu-project-an-open-source-academic-information-platform.html` [1]

2.2 Contexto Histórico

O projeto Fénix teve início em 2002 no IST, em Lisboa, com o objetivo de desenvolver um sistema avançado de informação académica para instituições de ensino superior que permitisse a automatização de processos relacionados com a gestão académica e a centralização da informação gerada por estes, numa base de dados [16]. O sistema foi concebido desde o início para ser totalmente baseado na Web, a fim de proporcionar uma ampla disponibilidade e uma fácil interação com o utilizador, independentemente do *software* do cliente e dos sistemas operativos, assegurando simultaneamente os elevados padrões de segurança, um rigoroso controlo do acesso e das acções de registo exigidos por um sistema de informação crítico [1].

Depois da fusão da Universidade Técnica de Lisboa com a Universidade de Lisboa em 2012/2013, que deu origem à ULisboa [8], e com vista à homogeneização de procedimentos e adotar estratégias partilhadas para a resolução de determinados problemas comuns a todas as Escolas, o SIGA começou a ser estendido em 2015, estando actualmente implementado nos SCULisboa e nas dezoito Escolas que compõe a ULisboa.

Face à complexidade, dimensão e necessidades específicas de cada Escola, o processo de criação e desenvolvimento de instâncias personalizadas para estas, foi faseado.

- **Fase Inicial:** Identificadas e selecionadas quatro Escolas que partilhavam os mesmos requisitos essenciais para o desenvolvimento da primeira versão do SIGA Fenix. Essa versão permitiu no ano lectivo 2015/2016, a entrada em produção das unidades SIGA Fenix nas Faculdade de Letras (FL), de Farmácia (FF), de Medicina Veterinária (FMV) e de Medicina Dentária (FMD), assim como, na Reitoria da ULisboa.

- **Fase Intermédia:** No ano letivo seguinte, 2016/2017, foram selecionadas nove Escolas adicionais e, realizado o reconhecimento das suas necessidades específicas, foram criadas as suas novas instâncias. Esta fase estendeu assim o SIGA Fenix às Faculdades de Belas-Artes (FBA), de Ciências (FC), de Direito (FD), de Medicina (FM), de Motricidade Humana (FMH) e de Psicologia (FP), e aos Institutos de Ciências Sociais (ICS), de Educação (IE) e de Geografia e Ordenamento do Território (IGOT).

- **Fase Final:** Começa com a entrada em produção das instâncias das Faculdade de Arquitetura (FA) e do Instituto Superior de Agronomia (ISA) e culmina no ano letivo de 2020/2021, com a entrada em produção das últimas instâncias Fenix dos Institutos Superiores de Ciências Sociais e Políticas (ISCSP) e de Economia e Gestão (ISEG), ficando assim todas as Escolas da ULisboa abrangidas por este SIGA [16].

2.3 Arquitetura

O SIGA Fenix apresenta uma arquitetura baseada no modelo *Model-View-Controller* (MVC) [16] - um padrão de *design* de *software* bastante comum entre programadores - que se caracteriza por estar dividido em três camadas distintas que separam as características da aplicação, no entanto interligadas entre si. Neste padrão, o primeiro nível *Model* é utilizado para comunicar com a base de dados e definir as regras de negócio. O segundo nível, *View*, é utilizado para obter dados do modelo e apresentá-los ao utilizador. Por último, o *Controller*, gere a interação do utilizador e atua como intermediário entre o *Model* e a *View*, isto é, recebe determinada entrada do utilizador, processa-a, e atualiza as outras camadas conforme necessário [13].

No SIGA Fenix, estas três camadas são designadas por **camada de persistência** (*Model*) onde são geridos os dados da aplicação, **camada de apresentação** (*View*) responsável por apresentar de forma visual os dados da aplicação ao cliente e, por último, **camada de aplicação** (*Controller*) que atua como elo de ligação entre as camadas anteriores, comunicando os pedidos gerados na camada de apresentação pelo cliente (*View*), à camada de persistência (*Model*), e vice-versa [16]. No entanto, uma diferença para o modelo MVC, é que, no SIGA Fenix, as regras de negócio estão aplicadas na camada aplicação (*Controller*), e não na camada de persistência *Model*.

- **Camada de Persistência:** Esta camada é assegurada pela *framework* Fenix, cujo modelo de domínio é expresso recorrendo a entidades, e é responsável por gerir os dados do SIGA Fenix sendo que todas as mudanças feitas sobre as entidades, são escritas numa base de dados relacional MariaDB, mantendo a integridade e coerência dos dados ao nível transacional.

- **Camada de Apresentação:** Esta camada é assegurada pela linguagem de modelação da *framework Omnis*, concretamente a *Presentation Specific Language* (PSL), e é aqui que são definidas e desenvolvidas as interfaces que permitem a interação dos clientes com o SIGA Fenix.

- **Camada de Aplicação:** É responsável pela comunicação entre as camadas anteriores e é utilizada para responder aos pedidos do utilizador, servindo de intermediário entre estas. Nesta camada, são aplicadas, pelos programadores, as regras de negócio às classes compiladas pela *Domain Modeling Language* (DML), ou seja, é nesta camada que são definidas as regras de negócio associadas à aplicação.

Nota: As *frameworks* Fenix e Omnis referidas, serão detalhadas na secção 2.5. As linguagens de modelação PSL e DML, na secção 2.6.

2.4 Ferramentas e Tecnologias

Por forma a disponibilizar as várias funcionalidades de gestão académica, o SIGA Fenix recorre a algumas ferramentas e tecnologias, nomeadamente:

- **Java:** É a linguagem utilizada pela equipa de desenvolvimento do Fenix, sendo, naturalmente, utilizada no desenvolvimento deste projecto. É uma linguagem de programação baseada em classes, orientada a objetos, e independente de plataforma. É uma das linguagens de programação mais utilizadas, o que, só por si, traz uma grande vantagem no desenvolvimento de software, na medida em que existe uma vasta comunidade de utilizadores e abundante documentação, acessível a todos [7]. Para além disso, uma outra grande vantagem desta linguagem é a sua portabilidade. O Java funciona em qualquer máquina, independentemente do *hardware* e sistema operativo em que é executado, uma vez que a execução do *bytecode*, gerado na compilação do código, pode ser realizada em qualquer *Java Virtual Machine* (JVM) [17].

- **Eclipse:** o Eclipse é um *Integrated Development Environment* (IDE) que proporciona um ambiente de desenvolvimento unificado para diversas linguagens de programação, famoso pelo seu IDE de Java. O Eclipse tem ainda um *Marketplace* que permite a sua personalização e extensão [10].

- **Docker:** É uma plataforma de *software* que permite a criação, o teste e a implantação de aplicações de forma rápida. Este gera pacotes de *software* em unidades padronizadas chamadas de contentores (*Containers*) que têm tudo o que um determinado *software* precisa para ser executado, nomeadamente bibliotecas, ferramentas de sistema, código e *runtime*. Ao usar esta ferramenta, é possível implantar e escalar rapidamente aplicações em qualquer ambiente e ter a certeza de que o seu código será executado [21].

- **Git:** Git é um sistema de controlo de versões distribuído, amplamente utilizado para gerir o desenvolvimento de *software*, em projetos de qualquer dimensão. Este permite rastrear alterações no código-fonte ao longo do tempo, bem como a colaboração entre programadores, na medida em que facilita a integração de código de diferentes colaboradores, permitindo assim que vários programadores trabalhem em conjunto num desenvolvimento não linear [22].

- **MariaDB:** É um sistema de gestão de bases de dados relacional, derivado do *MySQL* que tem a garantia de se manter como código aberto. Como base de dados relacional, fornece uma interface SQL (*Structured Query Language*) para aceder aos dados. O *MariaDB* é

rápido, escalável, robusto e muito versátil [11].

- **TomCat:** É um servidor *web* Java que permite executar instâncias *web* locais derivadas dos ambientes de desenvolvimento [23]. No âmbito deste projeto é utilizado como servidor *web* Java para correr o ambiente de desenvolvimento do Fenix.

- **Maven:** É uma ferramenta de automação de compilação que pode ser utilizada para construir e gerir projetos baseados em Java, simplificando o processo de construção, documentação e gestão de dependências do projeto Java [20].

2.5 *Frameworks*

O sistema Fenix utiliza três *frameworks*: a Fenix, a Bennu e a OMNIS [2], detalhados de seguida:

- **Fenix:** A Fenix é a mais antiga *framework* do sistema e constitui a sua base fundamental. Concebida para simplificar a representação de modelos de domínio em Java convencional, oferece um modelo de programação natural aos desenvolvedores. Destaca-se pela introdução da DML, uma linguagem de domínio própria - abordada na secção 2.6 - que facilita a definição de tipos de entidades e associações no modelo de domínio. Na DML, são descritas entidades, atributos e relações de multiplicidade. Essa descrição na DML é então compilada em classes Java correspondentes pela *framework* Fenix, proporcionando uma abordagem simplificada para a definição de comportamentos específicos em Java na *framework* Fenix [1]. Além disso, esta *framework* é responsável pelo modelo transacional e pelo modelo de persistência, assegurando a integridade e coerência dos dados [2].

- **Bennu**: A *framework* Bennu, desenvolvida a partir do projeto FenixEdu no IST, surge como uma solução para gerir o crescimento significativo de código na aplicação FenixEdu. Focando-se em autenticação, escalonamento e desenvolvimento de aplicações web, esta *framework* estabelece fronteiras claras para as interações entre objetos, reduzindo a complexidade e fomentando a reutilização. Ao conferir ao Fenix funcionalidades como autenticação de utilizadores, pesquisa e indexação de informações, criação de grupos de acesso e tarefas periódicas, a *framework* Bennu contribui para a robustez e eficiência do sistema Fenix [1].

- **OMNIS**: é a mais recente *framework* para o desenvolvimento de módulos e funcionalidades no sistema Fenix. Esta *framework* permite o desenvolvimento da camada de domínio e da camada de apresentação através de duas linguagens: a *Domain Specific Language* (DSL) e a *Presentation Specific Language* (PSL), respetivamente. Como veremos de seguida e em mais detalhe, a DSL é responsável pela modelação do domínio das entidades, dos atributos

dos mesmos e das relações com outros. A PSL é responsável pela modelação dos ecrãs de apresentação, baseando-se no padrão de desenho *Create, Read, Update, Delete* (CRUD) para a gestão dos ecrãs no sistema. É ainda possível determinar ações e eventos dos ecrãs criados para uma melhor gestão dos objetos de domínio. É através desta *framework* que os ecrãs de procura, criação, visualização e edição dos objetos, são criados automaticamente durante a compilação do ficheiro da PSL [2].

2.6 Linguagens de Modelação

Como mencionado anteriormente, no desenvolvimento em Fenix são utilizadas três linguagens de modelação: duas para a modelação de entidades do domínio e que são a DSL (*framework* OMNIS) e a DML (*framework* Fenix), e uma terceira, para a modelação de ecrãs ao nível da Camada de Apresentação, a PSL (*framework* OMNIS).

2.6.1 Linguagens de Modelação de Domínio

No Fenix, a modelação das entidades do domínio é baseada na linguagem DSL (*Domain Specific Language*) da *framework* Omnis e na DML (*Domain Modeling Language*) da *framework* Fenix. Essa dualidade de linguagens é motivada pela necessidade de compatibilidade, sendo que a criação da DSL surgiu como uma solução estratégica para superar desafios específicos durante o desenvolvimento da *framework* Omnis, por parte da qubIT em colaboração com a Reitoria da Universidade de Lisboa (RUL) [16]. Aquando do desenvolvimento da *framework* Omnis, o principal desafio tinha que ver com a evolução da PSL (*Presentation Specific Language*), que é uma linguagem específica para apresentação na *framework* Omnis. A PSL dependia da introdução de novas funcionalidades na DML, especialmente na criação de métodos para obter metadados dos atributos das entidades. No entanto, implementar essas funcionalidades na DML exigiria um esforço considerável de desenvolvimento. Para minorar esse desafio e tornar o processo mais eficiente, optaram por criar então a DSL como linguagem de domínio própria, que já incluía todas as funcionalidades necessárias para o correto funcionamento da PSL. Isto permite que a DSL seja utilizada em futuros desenvolvimentos, mantendo inalteradas as definições de domínio feitas com a DML [16].

Quanto ao processo de desenvolvimento, este inicia-se com a redação da DSL pelo programador - após a definição inicial do modelo de domínio - onde são definidas as entidades com base nos atributos e nas relações com outras entidades, conforme representado no excerto da DSL criada no âmbito do *bootcamp*, e ilustrada na figura 2.1.

bootcamp_academic.dsl

```
(...)
entity Person channels (WebJava) {
   required String name;
   unique required String username;
   unique String email;
   DateTime dateOfBirth;

   manyToOne DomainRoot domainRoot;
   }

 entity Teacher extends Person channels (WebJava){
    unique required Integer teacherNumber;

    oneToMany Lesson lesson mappedBy teacher;
    manyToMany (TeacherBelongsToFaculty) Faculty faculty mappedBy teacher;
    }

 entity Student extends Person channels (WebJava){
    unique required Integer studentNumber;

    manyToMany (StudentEnrolledInDegree) Degree degree mappedBy student;
    manyToMany (StudentEnrollesInCourse) Course course mappedBy student;
    manyToMany (StudentEnrollesInLesson) Lesson lesson mappedBy student;
    }

 entity Degree channels (WebJava){
     required Integer requiredEcts;
    unique required LocalizedString name;

    manyToOne DomainRoot domainRoot;
    manyToMany (StudentEnrolledInDegree) Student student;
    manyToMany (DegreeHasCourses) Course course mappedBy degree;
 }
(...)
```

Figura 2.1: Excerto da DSL desenvolvida no âmbito do *bootcamp* realizado.

Na compilação da DSL, o compilador da *framework* Omnis gera automaticamente a DML. Na DML, as entidades são representadas como classes, preservando todos os atributos da DSL e transpondo as relações para uma representação que destaca a multiplicidade associada. Essa abordagem permite aproveitar as funcionalidades da *framework* Fenix por meio da DML e garantir o correto funcionamento da PSL.

A figura 2.2 ilustra um exemplo da sequência de classes geradas pelo excerto da DSL (exibido anteriormente na figura 2.1), no IDE utilizado: *Eclipse*.

Abaixo, são explicados os pontos numerados na figura 2.2:

1. **Excerto da DSL com a Classe *Teacher.java***: A classe `Teacher` estende `Person`, estabelecendo a base para a estrutura da entidade.

2. **Todas as classes enumeradas na DSL**: Esta etapa mostra as classes geradas a partir da definição da DSL.

3. **Classe *Teacher.java***: É a classe principal, onde o programador pode definir regras de negócio e comportamentos específicos.

4. **Classe *Teacher_Base.java***: Esta é a classe base gerada automaticamente, que contém o construtor da classe `Teacher`, bem como os métodos *getters* e *setters* para cada atributo e relação definidos.

5. **Lista de métodos gerados automaticamente pela classe *Teacher_Base.java***: Aqui, são exibidos os métodos *getters* e *setters* gerados automaticamente para os atributos e relações da classe `Teacher`.

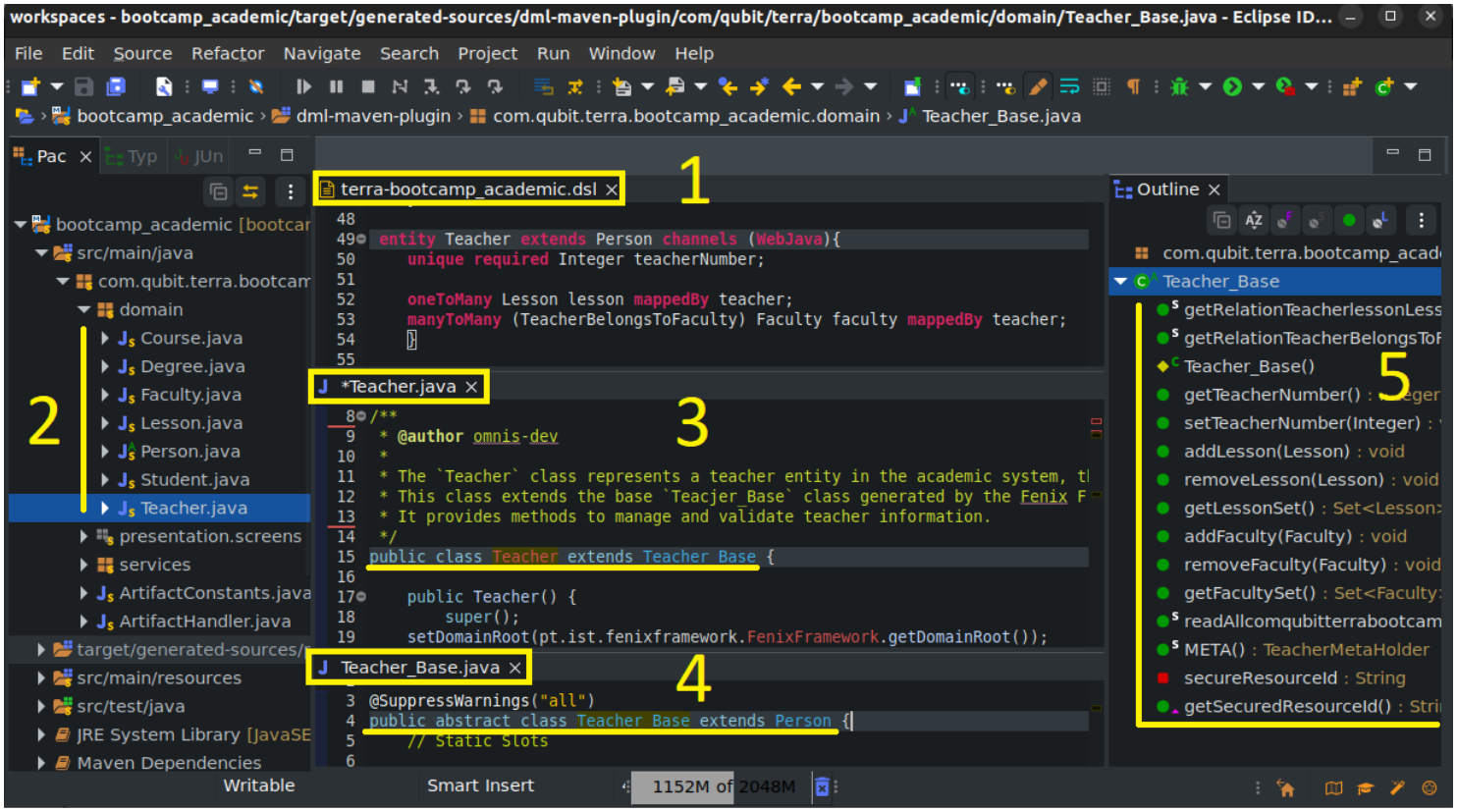


Figura 2.2: Eclipse IDE: Representação de excerto da DSL e da classe *Teacher* e dos respetivos métodos gerados após a compilação.

Em suma, após a definição da DSL (1), o compilador da *framework* Fenix cria as classes Java correspondentes à DML (2). Cada entidade definida na DML resulta em duas classes Java: uma classe base (3) e outra que estende a base (4) e que contem os métodos gerados automaticamente (5). As classes base, como `Teacher_Base.java`, não devem ser alteradas diretamente, pois são geradas a cada compilação para refletir eventuais alterações na DSL e na DML. Assim, a cada compilação, as classes base são "esmagadas". Por outro lado, as classes que estendem a base, como `Teacher.java`, são geradas apenas se ainda não existirem, garantindo que as personalizações feitas pelo programador não sejam substituídas pelo compilador, a menos que as classes sejam eliminadas manualmente antes de uma recompilação.

2.6.2 Linguagem de Modelação da Camada de Apresentação

A PSL (*Presentation Specific Language*) é a linguagem de modelação da camada de apresentação na *framework* OMNIS, do sistema Fenix. Essa linguagem permite aos programadores criar e personalizar ecrãs e fluxos de apresentação para visualizar objetos gerados pela DSL, sem a necessidade de compreender profundamente a camada interna de apresentação.

Por oferecer a conveniência de organizar os ecrãs de apresentação num único ficheiro, seguindo uma estrutura única, é na PSL que são criados os diversos *flows*, nos quais são definidos os vários ecrãs. Nessa estrutura, semelhante à DSL, é possível definir a localização dos ecrãs nas pastas do projeto, especificar o nome do ecrã, o tipo, os fluxos de apresentação, o tipo de objeto a ser utilizado no ecrã, os atributos dos objetos a serem exibidos, assim como os nomes de operações e eventos associados. Essa abordagem simplificada e estruturada permite uma maior flexibilidade no desenvolvimento da camada de apresentação no Fenix [2].

Há 4 tipos de ecrãs previamente modelados ao dispor do programador e que são do tipo: *Create*, *Read*, *Update* e *Search*. Como é possível inferir pelos nomes, estes são utilizados, respetivamente, para: criação de entidades, leitura de valores associados a determinadas entidades, alteração de valores de alguns atributos e, para a pesquisa com apresentação da informação de forma tabular. Para além destes 4, é também possível a definição de ecrãs genéricos do tipo *Screen*, nos quais o programador tem uma maior flexibilidade e liberdade na utilização de componentes para a modulação do ecrã de acordo com as necessidades evidenciadas [16].

Para definir os fluxos entre ecrãs, são definidos eventos e ações que realizarão uma mudança no fluxo de ecrãs. Ao definir eventos, o programa cria botões para o propósito do evento, enquanto que ao definir ações, este cria um texto clicável que permitirá progredir nos fluxos. Primeiro são definidos os eventos e depois as ações, seguindo-se alguns exemplos destes [2].

Eventos:

- *CreateEvent*:
  Para navegar para o ecrã de criação de objetos;

- *UpdateEvent*:
  Para navegar para o ecrã de edição de objetos;

- *DeleteEvent*:
  Para apagar um objeto que está a ser visualizado no ecrã de leitura;

- *BackEvent*:
  Para retroceder para o ecrã precedente de navegação;

- *CancelEvent*:
  Para cancelar a criação ou edição de objetos;

- *Event*:
  Para criar novos eventos;
- *SelectMultipleEvent*:
  Para selecionar vários objetos e realizar o comportamento desejado nos objetos selecionados.

Ações:

- *ViewAction*:
  Para visualizar os detalhes de um objeto;
- *DeleteAction*:
  Para apagar obejcto associado ao evento;
- *Action*:
  Para criar novas Ações.

A figura 2.3 contêm um excerto da PSL realizada no âmbito do *bootcamp*, e ilustra o *flow* para gestão de docentes - `manageTeachers` - bem como os tipos de ecrãs e de eventos, anteriormente referidos.

bootcamp_academic.psl

```
(...)
//Gestão de Professores
flow manageTeachers channel(WebJava) {

  searchScreen searchTeacher [Teacher]
(fields name username teacherNumber email dateOfBirth)
   (searchFields name username teacherNumber email) {
    backEvent hidden
    createEvent -> createTeacher
    viewAction -> readTeacher
    deleteAction -> searchTeacher
  }

  createScreen createTeacher [Teacher] (fields name username teacherNumber email dateOfBirth) {
    createEvent -> searchTeacher
    cancelEvent -> searchTeacher
  }

  readScreen readTeacher [Teacher] (fields name username teacherNumber email dateOfBirth lesson faculty) {
    backEvent -> searchTeacher
    updateEvent -> updateTeacher
    deleteEvent -> searchTeacher

    dialog event assignTeacherToLesson -> searchAssignTeacherToLesson
     dialog event unassignTeacherToLesson -> searchUnassignTeacherToLesson
  }

  updateScreen updateTeacher [Teacher] (fields name username teacherNumber email dateOfBirth) {
    updateEvent -> readTeacher
    cancelEvent -> readTeacher
  }
(...)
```

Figura 2.3: Excerto da PSL desenvolvida no âmbito do *bootcamp* realizado.

Por último, referir que, assim como na DML, por cada ecrã definido na PSL, são geradas duas classes Java: uma classe base e outra que a estende. O programador pode ajustar o ecrã conforme necessário, na classe que estende a classe base. A geração destas classes é realizada pelo compilador da *framework* Omnis [16].

2.7 *OmnisBar*

A *OmnisBar* proporciona diversas funcionalidades que simplificam o desenvolvimento de recursos no sistema Fenix, oferecendo opções aos programadores. Esta ferramenta está localizada na parte inferior dos ecrãs e possui quatro menus: *OmnisBar*, Introspecção, Desenvolvimento e Localização.

Através da *OmnisBar* é possível aceder a atributos específicos dos ecrãs de apresentação. Esta funcionalidade permite a quem está a programar poder, por exemplo, ativar ou desativar a opção de *debug* em tempo real e efetuar alterações nas etiquetas de texto dos ecrãs, evitando assim ter que recompilar o código. Assim, ao ter a capacidade de recarregar ecrãs em tempo real, o programador pode efetuar alterações nas classes dos ecrãs e atualizar essas classes no servidor sem recompilar o código ou reiniciar o servidor *web*. Contudo, importa salientar, que isto é apenas possível em classes da camada de apresentação ou nos serviços do sistema, uma vez que, as alterações em classes de domínio ou nas classes da DSL e PSL, requerem sempre a recompilação do código, e o consequente reinício do servidor *web*, para que as alterações no código sejam refletidas.

Outra mais-valia é a gestão de etiquetas de texto nos ecrãs de apresentação que permite a edição das etiquetas no servidor *web* e a sua gravação num ficheiro. Assim, existindo alterações nos ecrãs ou na base de dados, deixa de ser necessário alterar as etiquetas individualmente, passando a ser suficiente carregar o ficheiro para o servidor, o que é bastante útil, por exemplo, para assegurar e validar, as corretas etiquetas em Português e Inglês (PT/EN).

Constatamos assim, que as funcionalidades trazidas pelas *OmnisBar* - e acima identificadas - agilizam o ciclo de desenvolvimento, reduzindo o tempo entre a implementação das alterações no código e a sua visualização nos ecrãs, melhorando o processo de desenvolvimento.

2.8 *Workflows*

Um *workflow* é um conjunto de passos predefinidos que devem ser realizados por um ou mais intervenientes, fazendo com que o processo percorra diversas fases. Através deste conjunto de passos e ações os intervenientes pretendem alcançar um determinado objetivo [16].

O SIGA Fenix incorpora um motor de *workflow*, que tem um comportamento semelhante a uma máquina de estados e que permite modelar vários fluxos de processos [2].

Para modelar um *workflows* importa conhecer os seguintes conceitos:

- **Estado**: Um estado é definido como uma fase do processo, possuindo um significado específico para a entidade que o define. O estado designado *Inicial* marca o início do *workflow*, e o *Final*, o seu fim.

- **Separador**: Os separadores estão associados aos estados e são os componentes onde os utilizadores podem consultar e/ou inserir dados. É possível definir permissões para os separadores com base nos perfis dos utilizadores. Um mesmo separador pode estar em estados diferentes com permissões distintas.

- **Transições**: As transições são desencadeadas automaticamente ou através da execução de determinadas operações manuais - geralmente materializadas por botões nas interfaces - e representam a passagem de um estado, para o seguinte.

No contexto do Fenix, os estados são classificados em três tipos: automáticos, manuais e condicionais.

- **Estados automáticos:** Executam automaticamente a operação associada, sem a intervenção do utilizador.

- **Estados manuais:** Requerem a intervenção do utilizador para transitar para o próximo estado.

- **Estados condicionais:** Permitem definir sequências alternativas com base na avaliação de uma condição, desencadeando comportamentos diferentes na aplicação.

O estados manuais podem incluir três tipos de separadores: dinâmicos, personalizáveis e de documentos.

- **Separadores dinâmicos:** Geralmente contêm formulários a serem preenchidos durante a execução do processo.

- **Separadores personalizáveis:** Concedem aos programadores maior liberdade na transmissão de informações aos utilizadores.

- **Separadores de documentos:** Permitem a submissão e consulta de documentos necessários durante o processo.

Outra componente fundamental em qualquer processo é a temporal. Para ajudar nesse campo, o Fenix possui temporizadores que são operações especiais que permitem definir períodos temporais e desencadear a execução de outras operações associadas. Para além disso, é também possível enviar notificações a determinados perfis de utilizadores, podendo ser feito na entrada ou saída de um estado ou mediante solicitação, ou seja, manualmente.

Outro componente no Fenix são os eventos, que desencadeiam um comportamento específico no *workflow* com base em certos acontecimentos. Este componente pode ser usado, por exemplo, para transições de estados após o pagamento de emolumentos.

É também possível lançar excepções e informar os utilizadores sobre algo relacionado ao processo, com base na verificação de uma determinada condição.

Por fim, é possível lançar subprocessos dentro de um determinado processo, simplificando a modelação ao dividí-lo em subprocessos mais simples.

A criação de *workflows* no Fenix é realizada através do *Process Modeler da framework* Omnis, que permite a modelação gráfica e a configuração das várias componentes que compõem o *workflow* [16]. A figura 2.4 apresenta os diferentes símbolos disponíveis na criação de *workflows* no Fenix.

Figura 2.4: Conjunto de elementos da linguagem de modelação de *workflows* da *framework* Omnis [16].

2.9 Documentos *Online*

A *framework* Omnis disponibiliza a funcionalidade de criar *templates* para a produção de documentos *online*. Esta ferramenta possibilita a personalização completa de um *template* para gerar documentos durante um *workflow* e, no âmbito da Mobilidade, poderá vir a ser utilizada, por exemplo, para a produção de documentos como a Carta de Aceitação, Declarações de Chegada/Partida e/ou o *Transcript of Records* (ToR).

O acesso aos Documentos *Online*, ou seja, à ferramenta de criação de *templates*, está disponível em qualquer instância Fenix, permitindo a cópia de *templates* entre instâncias. Esta funcionalidade simplifica o processo de criação de documentos, uma vez que, na maioria dos casos, os *templates* são idênticos entre instâncias, sendo apenas necessário ajustar os logótipos da instituição e outros dados específicos [16]. O que também simplifica a sua utilização, é o facto desta se assemelhar aos editores de texto comuns, permitindo a inserção de imagens, tabelas e formatação de texto.

Por último, referir que cada *template* pode estar associado a várias fontes de dados, facilitando o preenchimento automático de informações no documento. No entanto, para esse preenchimento, são utilizados os valores dos atributos das entidades associadas à instância do *workflow* onde o documento é gerado, o que, no âmbito da Mobilidade, poderá trazer desafios acrescidos.

2.10 Sumário

Neste capítulo, introduzimos o SIGA Fenix, começando pelo seu contexto histórico, avançando para a sua arquitetura e identificando de seguida as ferramentas, tecnologias e *frameworks* utilizadas. Na parte final, resumimos as principais funcionalidades disponibilizadas pela OmnisBar e pela ferramenta de gestão de *workflows* da *framework* Omnis, terminando com o resumo da ferramenta de criação de *templates* de documentos - documentos *online*.

No capítulo seguinte iremos detalhar o levantamento de requisitos onde serão clarificados alguns conceitos-chave e o EWP, e onde serão partilhados os processos de mobilidade *incoming* existentes em algumas Escolas. De seguida faremos uma análise tanto dos requisitos funcionais como dos não funcionais e avançaremos para a arquitetura. Com base nestes pontos, partilharemos o desenho da nossa proposta para o Módulo de Mobilidade *Incoming* e o Diagrama de Entidades e Associações (DEA) resultante. Será esta primeira proposta que servirá de base ao trabalho a ser desenvolvido no decorrer deste projeto.

Capítulo 3

Requisitos e Desenho

Neste capítulo, detalhamos o trabalho realizado até à fase de implementação, começando pelo levantamento e análise de requisitos, essenciais para estabelecer os conceitos e desafios do projeto de mobilidade. Em seguida, abordamos a arquitetura proposta e o desenho da solução, representado pela modelação do processo em BPMN, a análise das comunicações EWP, e o desenvolvimento do Diagrama de Entidades e Associações (DEA), que servirão de base para o desenvolvimento subsequente.

3.1 Levantamento de Requisitos

No levantamento de requisitos, começaremos por identificar conceitos-chave sobre a mobilidade. De seguida, faremos uma contextualização do EWP e depois do processo de mobilidade, abordando tanto o seu funcionamento global como, em particular, o processo *Incoming* apurado junto das Escolas.

3.1.1 Conceitos-Chave sobre Mobilidade

São apresentados de seguida alguns conceitos-chave da mobilidade académica identificados no levantamento de requisitos e que fornecem uma compreensão fundamental para o desenvolvimento deste projecto.

- **Erasmus+**: É um programa da União Europeia que oferece oportunidades de mobilidade para estudantes e pessoal nas áreas de educação, formação, juventude e desporto.
- **Mobilidade Erasmus+**: Refere-se aos programas de mobilidade oferecidos no âmbito do programa Erasmus+, que possibilita a troca de estudantes e pessoal docente e não docente entre instituições de ensino superior europeias.
- **NM - Núcleo de Mobilidade**: Refere-se à estrutura administrativa dentro de cada Escola que lida com os assuntos relacionados com a mobilidade da sua comunidade. Este núcleo é responsável por coordenar e facilitar os processos de mobilidade, incluindo a gestão de documentação, a comunicação com os estudantes e a colaboração com instituições parceiras. Estas estruturas podem ter diferentes designações em cada Escola, como por exemplo,

Gabinete de Mobilidade (GM), Divisão de Relações Externas e Internacionais (DREI), entre outras.

- **EWP - *Erasmus Without Paper***: É uma iniciativa digital que visa conectar sistemas de gestão de mobilidade Erasmus+ para facilitar a gestão de estudantes em mobilidade de forma eficiente e sem papel.
- ***Incoming***: O termo *Incoming* é utilizado para descrever os participantes que chegam a determinada instituição de ensino para realizar um período de mobilidade.
- ***IIA - Inter-institutional Agreements***: Os Acordos Interinstitucionais referem-se a acordos estabelecidos entre instituições de ensino superior para promover a mobilidade académica e a colaboração.
- **SMS - *Studies Mobility***: Refere-se à mobilidade académica relacionada com estudos, onde os estudantes realizam UCs numa instituição de ensino superior diferente da sua instituição de origem.
- **SMP - *Traineeships Mobility***: Diz respeito à mobilidade relacionada com estágios/projetos, em que os estudantes têm a oportunidade de realizar períodos de formação prática numa instituição ou empresa estrangeira.
- **Fluxos**: Os Fluxos representam o volume de trocas acordado entre partes, dos diferentes tipos de mobilidade (SMS ou SMP), medido em vagas por período temporal (Semestre/Ano).
- **Nomeação**: Processo formal em que uma instituição de ensino superior de origem indica oficialmente um estudante para participar num programa de mobilidade a uma instituição de ensino superior de destino. Essa indicação é realizada tendo em conta os fluxos acordados e em vigor, entre ambas as instituições.
- **LA - *Learning Agreements***: São documentos normalizados que estabelecem o plano de estudos acordado entre o estudante e as instituições envolvidas na mobilidade.
- **ToR - *Transcript of Records***: É um documento que detalha o desempenho académico de um estudante durante um determinado período.
- **Declaração de Chegada e Declaração de Saída**: As Declarações de Chegada e Saída são documentos que confirmam oficialmente o início e o fim do período de mobilidade de um estudante numa instituição.

3.1.2 *Erasmus Without Paper* (EWP)

O projeto EWP é uma iniciativa desenvolvida para simplificar e digitalizar os processos administrativos relacionados com o programa Erasmus na área da educação. Este projeto faz parte

da Iniciativa Cartão Europeu de Estudante, que é uma iniciativa destinada a ajudar os estudantes e as instituições de ensino superior nos intercâmbios Erasmus+, simplificando os processos administrativos e reforçando a digitalização.

De seguida veremos de forma breve os seus principais objectivos, os benefícios que traz, as funcionalidades que tem e de que forma podem as instituições de ensino superior aderir.

Objectivos: Os principais objectivos do EWP são eliminar o uso de papel e melhorar os processos, facilitando a troca de informações entre as instituições de ensino superior participantes. Para tal, o EWP proporciona uma plataforma eletrónica que permite às instituições gerir de forma mais eficaz a mobilidade de estudantes, docentes e funcionários no âmbito do programa Erasmus. Através desta plataforma, é possível realizar tarefas como o registo de candidaturas, nomeações, emissão de documentos e a partilha de informações relevantes de forma digital. Assim, ao eliminar a necessidade de papel e simplificar os processos, o EWP pretende melhorar a experiência dos participantes no programa Erasmus, tornando a gestão administrativa mais rápida, transparente e acessível. Este projeto representa um esforço conjunto de diversas instituições de ensino superior na Europa para modernizar e agilizar os procedimentos associados à mobilidade académica no âmbito do programa Erasmus [6].

Benefícios: Ao gerir as mobilidades dos seus estudantes com o EWP, a instituição de ensino superior pode usufruir dos seguintes benefícios[3]:

1. **Substituir o Processo em Papel:** Eliminar o uso de papel ao adotar um fluxo de trabalho digital no âmbito do programa Erasmus.

2. **Reduzir a Carga Administrativa:** Simplificar e agilizar o processo de troca de estudantes, aliviando o trabalho tanto dos estudantes quanto do pessoal de apoio.

3. **Aproveitar Soluções Técnicas Eficientes:** Utilizar e integrar soluções técnicas existentes de forma eficiente, promovendo a conectividade e facilitando a mobilidade estudantil.

4. **Contribuir para uma Infraestrutura Pública Gratuita:** Participar e beneficiar de uma infraestrutura pública gratuita, promovendo a colaboração entre Instituições de Ensino Superior na União Europeia.

Principais Funcionalidades: De acordo com os responsáveis do projeto, o princípio fundamental do EWP, é que as instituições de ensino superior mantenham o seu sistema atual de gestão da mobilidade dos estudantes, mas que os conecte à rede EWP [4]. É esta conectividade entre instituições de ensino superior que tratá os benefícios descritos fornecendo às instituições de ensino superior as seguintes funcionalidades:

- Aprovar documentos *online*;

- Enviar documentos através do próprio sistema de gestão da mobilidade da instituição, à instituição parceira;
- Solicitar à sua instituição parceira que os aprove digitalmente.

Adesão: Uma instituição do ensino superior pode aderir à rede EWP, caso tenha uma Carta Erasmus para o Ensino Superior (CEES), através de três possíveis opções [5]:

- ***Software* para gestão de mobilidade:** Existem fornecedores de *software* que conectam o *software* existente nas instituições de ensino superior, à rede EWP.
- ***Software* interno ou que é usado por outras instituições:** Ao integrar o seu *software* interno de gestão de mobilidade à rede EWP, é possível estabelecer comunicação com outros sistemas. Para tal, a sua equipa de desenvolvimento de *software* da instituição deverá desenvolver as APIs necessárias para a integração efetiva com a rede EWP.
- ***Dashboard* EWP:** Uma ferramenta gratuita que permite gerir os processos básicos da mobilidade estudantil nas instituições que não utilizam nenhuma solução digital ou não tenham os meios e os recursos para desenvolver/adaptar uma solução. A *dashboard* está ligada à rede EWP, permitindo a troca de dados entre instituições parceiras, independentemente da solução de *software* utilizada. Também se conecta à aplicação móvel Erasmus+ da Comissão Europeia, permitindo a comunicação direta com estudantes em mobilidade *Incoming* ou *Outgoing*.

3.1.3 Processo de Mobilidade Erasmus+

O processo de Erasmus+ entre duas instituições de ensino superior funciona, em termos gerais, da seguinte forma[4]:

1. É assinado um acordo institucional Erasmus+ entre duas instituições de ensino superior onde são estabelecidos os fluxos e a duração do acordo;
2. A instituição de envio nomeia estudantes para a instituição de acolhimento;
3. Um *Learning Agreement* (LA) é estabelecido e assinado pelas três partes: instituição de envio, instituição de acolhimento e o aluno;
4. Confirmação da data de chegada do aluno por parte da instituição de acolhimento;
5. Em caso de alteração do LA, o mesmo terá de ser assinado novamente pela instituição de envio, instituição de acolhimento e o aluno;
6. Quando termina a mobilidade e o aluno parte da instituição de acolhimento, esta tem de confirmar a data de partida;
7. A instituição de acolhimento envia os registos para a instituição de envio, nomeadamente *Transcript of Records* (ToR) e declarações de chegada/partida.

Em cada uma destas etapas, quando é necessária a comunicação entre a instituição de acolhimento e de envio, será utilizada a rede EWP, através das suas APIs que visam facilitar a comunicação entre os sistemas. Isto permite que os processos sejam geridos nos próprios sistemas das instituições, recorrendo à rede EWP quando é necessária confirmação, assinatura ou outro tipo de interação por parte da instituição parceira.

A representação do macroprocesso de mobilidade Erasmus+, desde a iniciativa de estabelecer um *Inter-Institutional Agreement* (IIA), até ao fim da mobilidade de um aluno, foi elaborada em *Business Process Model and Notation* (BPMN) e pode ser consultada nas figuras A.1 e A.2, do anexo A.1.

3.1.4 Processo de Mobilidade *Incoming*

No âmbito da mobilidade, o foco deste projeto é a Mobilidade Erasmus *Incoming* e, como tal, importa conhecer os processos existentes atualmente. Para isso tivemos reuniões com os responsáveis dos Núcleos de Mobilidade das quatro Escolas selecionadas: FC, ISEG, FL e ISCSP.

Com base nessas reuniões foi-nos possível estabelecer os processos que, após terem sido validados juntos dos NM das Escolas, foram posteriormente representados em BPMN. Tanto o detalhe dos processos como as suas respectivas representações podem ser vistas em detalhe nos anexos:

- **FC**: Anexo B.1;
- **ISEG**: Anexo B.2;
- **FL**: Anexo B.3;
- **ISCSP**: Anexo B.4.

A partir das reuniões de levantamento de requisitos, foi possível identificar um conjunto de pontos em comum e podemos considerar que, em termos gerais, o macroprocesso da mobilidade *Incoming* tem a seguinte forma:

1. Instituição de Destino/Acolhimento (ID) recebe a nomeação do aluno em mobilidade;
2. ID valida a nomeação e os fluxos em vigor;
3. ID (ou o Aluno) formaliza a inscrição/matrícula do aluno;
4. Início da mobilidade - emissão/envio da declaração de chegada;
5. A partir do início de mobilidade, os alunos têm 15 dias para alterações ao LA;
6. ID cria registo de Mobilidade *Incoming* no Fenix;
7. Decorre período lectivo;
8. Fim da mobilidade com emissão de ToR e declaração de saída.

Com base neste macroprocesso, vamos de seguida analisar os requisitos.

3.2 Análise de Requisitos

Nesta secção, realizaremos uma análise detalhada dos requisitos do sistema, identificando os principais intervenientes, os requisitos funcionais e os requisitos não funcionais, terminando com um resumo dos principais desafios. Esta análise fornecerá uma base para o desenvolvimento de soluções alinhadas com as necessidades da ULisboa e das suas Escolas.

3.2.1 Intervenientes

Os intervenientes relevantes que desempenham um papel no processo de mobilidade *Incoming*, são os seguintes:

- **Aluno**: É o interveniente principal que participa no processo de mobilidade. A sua responsabilidade inclui seguir os procedimentos definidos, preencher a documentação necessária e cumprir os requisitos académicos da sua experiência de mobilidade.
- **Instituição de Destino - Núcleo de Mobilidade (NM)**: O NM da Instituição de Destino/Acolhimento (ID) desempenha um papel crucial na receção e integração dos alunos em mobilidade. Gere os processos de mobilidade e a documentação associada, fornecendo suporte aos alunos durante a sua mobilidade. A designação de NM pode ser diferente em cada Escola.
- **Instituição de Destino - Coordenador de Erasmus**: Quando existe um Coordenador de Erasmus na Instituição de Destino, este é responsável por supervisionar os programas de mobilidade Erasmus+. As suas funções podem incluir a coordenação de atividades académicas relacionadas com a mobilidade, o acompanhamento dos processos de mobilidade, a comunicação com outras instituições parceiras, a aprovação dos LA, etc.

3.2.2 Requisitos Funcionais

Aqui exploraremos os requisitos funcionais, que definem as funcionalidades específicas que o sistema deve oferecer aos intervenientes. Estas funcionalidades serão detalhadas na forma das *User Stories* seguintes:

- **Como Gabinete/Núcleo de Mobilidade (NM)** da Instituição de Destino/Acolhimento (ID), quero receber e validar nomeações de alunos vindos de outras instituições via EWP.

  Cenário: O NM recebe uma ou mais nomeações de alunos através do EWP, valida as informações e verifica se estão de acordo com os acordos interinstitucionais existentes.

- **Como NM da ID**, quero emitir uma carta de aceitação da ULisboa após a aceitar a nomeação de um aluno.

  Cenário: O NM emite uma carta de aceitação da ULisboa após aceitar a mobilidade.

- **Como NM da ID**, quero emitir uma declaração de chegada da ULisboa após o início das aulas, para registrar a presença do aluno (esta funcionalidade será objecto de análise com vista à sua automação).

  Cenário: O NM emite uma declaração de chegada da ULisboa após o início das aulas.

- **Como NM da ID**, quero assinar declarações de chegada diferentes, solicitadas pelos alunos, para atender às suas necessidades específicas.

  Cenário: O NM assina as declarações solicitadas pelos alunos, que podem variar de acordo com as necessidades individuais.

- **Como NM da ID**, quero validar alterações no plano de estudos do aluno e poder assinar novo LA, se necessário.

  Cenário: O NM valida as alterações no plano de estudos, atualizando os registros de mobilidade conforme necessário.

- **Como NM da ID**, quero criar um registo de Mobilidade *Incoming* no Fenix RAIDES, para fins de relatórios e estatísticas (esta funcionalidade será objecto de análise com vista à sua automação).

  Cenário: O NM cria um registo no Fenix RAIDES durante o período adequado.

- **Como NM da ID**, quero emitir um ToR no Fenix no final da mobilidade, para registrar o desempenho académico do aluno (esta funcionalidade será objecto de análise com vista à sua automação).

  Cenário: O sistema emite um ToR no Fenix no final da mobilidade.

- **Como NM da ID**, quero emitir uma declaração de saída/estada após o lançamento das notas, para oficializar o término da mobilidade (esta funcionalidade será objecto de análise com vista à sua automação).

  Cenário: O NM emite uma declaração de saída/estada após o lançamento das notas.

- **Como Coordenador de Erasmus**, desejo validar e autorizar alterações nos planos de estudos dos alunos em mobilidade, garantindo que estão alinhados com os requisitos académicos.

  Cenário: O coordenador de Erasmus valida que as alterações propostas nos planos de estudos, estão em conformidade.

- **Como aluno**, quero ter acesso a uma conta no Fenix para aceder a informações e serviços relacionados à mobilidade.

  Cenário: O aluno acede ao sistema Fenix com a conta criada para o efeito.

- **Como aluno**, quero poder obter diferentes declarações (chegada, saída, entre outras) da ULisboa, conforme necessário.

Cenário: O aluno acede ao sistema Fenix, localiza o seu processo de mobilidade e obtem as declarações oficiais da ULisboa.

- **Como aluno**, quero solicitar a assinatura de declarações especificas da minha ID(chegada, saída, entre outras) ao Núcleo de Mobilidade, conforme necessário.

  Cenário: O aluno acede ao sistema Fenix, localiza o seu processo de mobilidade e faz os pedidos necessários.

- **Como aluno**, quero ter acesso ao meu ToR no Fenix, no final da mobilidade.

  Cenário: O aluno acede ao sistema Fenix e encontra o seu ToR disponível para visualização e *download*.

- **Como aluno**, quero poder realizar alterações ao meu plano de estudos durante um período específico, caso seja necessário e solicitar nova assinatura do LA.

  Cenário: O aluno acede ao sistema Fenix e submete novo LA para assinatura, se necessário.

- **Como aluno**, quero receber notificações automáticas do sistema Fenix sobre alterações de estado do meu processo.

  Cenário: O aluno configura as suas preferências de notificação no sistema Fenix e recebe alertas definidos pela equipa de mobilidade.

3.2.3 Requisitos Não Funcionais

Os requisitos não funcionais abrangem um conjunto de características que são, na sua maioria, asseguradas pelo SIGA Fenix e que, no âmbito deste projeto de mobilidade, são as seguintes:

- **Segurança:** O projeto deve considerar os aspetos de segurança do SIGA Fenix, permitindo que somente intervenientes autorizados possam intervir no processo de mobilidade.
- **Usabilidade:** O resultado final do projeto deve ser semelhante aos outros módulos do SIGA Fenix, por forma a manter a usabilidade que os utilizadores finais estão habituados.
- **Flexibilidade:** O sistema deve ser desenhado para possibilitar a sua modificação em caso de alteração aos processos de mobilidade.
- **Multilinguismo:** O sistema deve ser bilingue, permitindo o uso de inglês e português.

3.3 Arquitetura

Após a conclusão do levantamento e análise de requisitos, avançamos para a definição da arquitetura e do design do módulo no contexto do SIGA Fenix.

Começando pela arquitetura do sistema, esta é ilustrada pela figura 3.1 e oferece uma visão abrangente da comunicação entre as instâncias que desempenham papéis fundamentais no processo de mobilidade *incoming* do projeto em desenvolvimento.

Esta representação ilustra os fluxos de comunicação essenciais do processo de mobilidade, definidos no âmbito do projecto EWP pelo Núcleo de Desenvolvimento de *Software* (NDS) dos Serviços Centrais da Reitoria da Universidade de Lisboa (SCULisboa), no seguimento de trabalho anteriormente já realizado, nomeadamente, a mobilidade *Outgoing*.

Um aspeto fundamental da arquitetura definida, é a necessidade de autenticação do utilizador, seja este da Reitoria da ULisboa, ou de uma das suas Escolas (excepto IST). Relativamente às Escolas, estas devem ser capazes de comunicar com a Reitoria da ULisboa, que, por sua vez, estabelece comunicação com o nó EWP das instituições de ensino superior parceiras de origem. As Escolas da ULisboa - à excepção do IST - não comunicarão diretamente com as instituições de ensino superior parceiras de origem via nó EWP, uma vez que, todas as comunicações com o nó EWP, deverão passar pela instância da Reitoria da ULisboa.

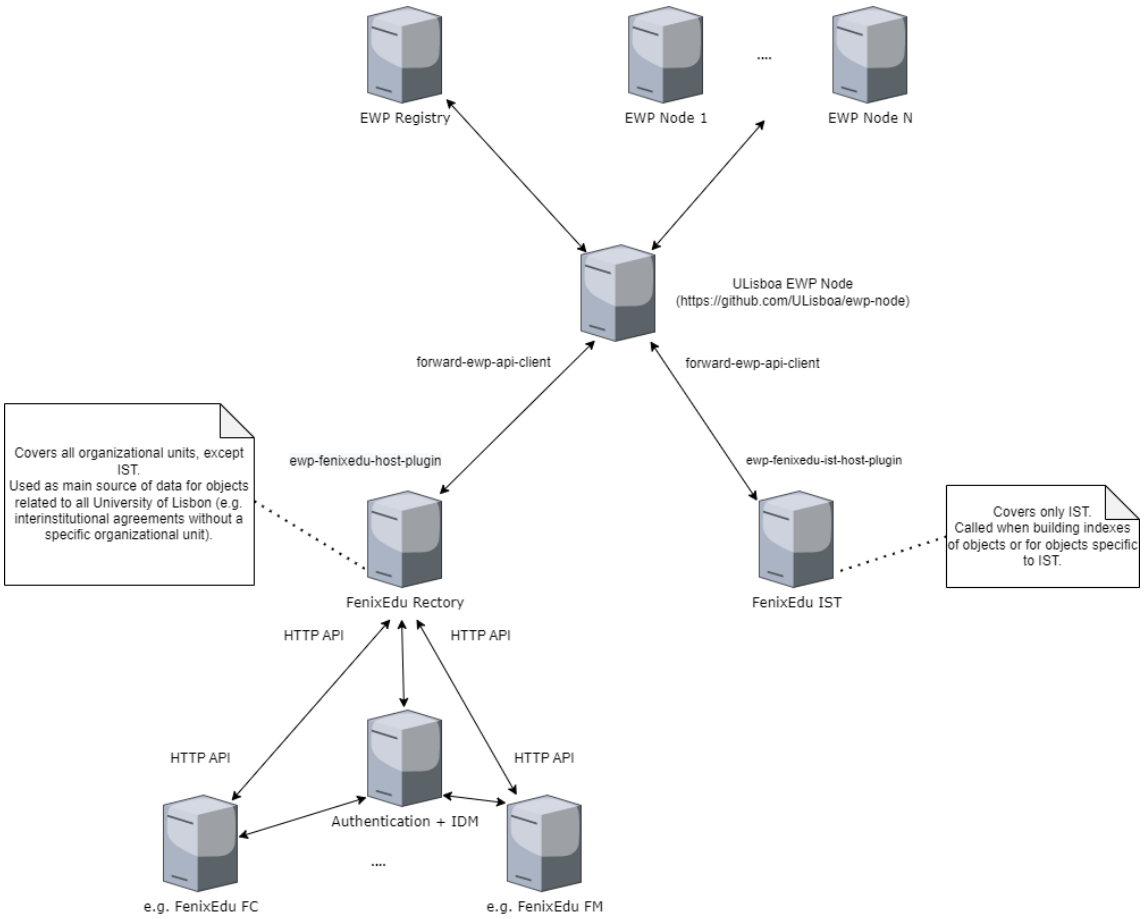


Figura 3.1: Diagrama de Rede EWP com a ULisboa - fornecido pelo NDS.

3.4 Desenho

Depois de estabelecida a arquitetura, avançámos para o desenho do processo, que será representado em três níveis principais: a modelação do processo em BPMN, a análise das comunicações definidas pelo EWP e, finalmente, o desenvolvimento do Diagrama de Entidades e Associações

(DEA). Este desenho servirá como base para a modelação do *workflow* na *framework* OMNIS e para a implementação do sistema.

3.4.1 Modelação do Processo de Mobilidade *Incoming*: BPMN

Com base no levantamento realizado, em que foram analisados os processos de mobilidade *incoming* de quatro Escolas, identificámos os pontos principais e comuns aos processos. Isto permitiu-nos definir as várias fases do processo e, de seguida, modelá-lo em BPMN — utilizando uma metodologia semelhante à adotada no levantamento de requisitos junto das Escolas.

O desenho resultante encontra-se representado em BPMN nas figuras C.1 a C.5 no anexo C.1. Importa ressaltar que, para simplificar a representação no diagrama BPMN referido, foram usadas apenas três *pools*[1]: Instituição de Origem, EWP e Instituição de Destino. Desta forma, o diagrama não ilustra o fluxo completo de encaminhamento visível na figura 3.1, em que a nomeação é inicialmente recebida pelo nó EWP na Reitoria e só depois redirecionada para a Escola correspondente, e vice-versa. No entanto, esta simplificação não compromete a leitura e interpretação do processo e dos seus vários estados.

De forma resumida, propomos que o processo de mobilidade Erasmus *Incoming* da ULisboa tenha por base a estrutura enumerada abaixo. Esta estrutura alinha-se com o fluxo de atividades descrito no diagrama BPMN, representando as etapas do processo de forma sequencial e correspondente.

1. A Instituição de Ensino Superior de Origem (IO) nomeia um aluno no âmbito do programa de mobilidade Erasmus, através do EWP, para uma Escola da ULisboa, designada Instituição de Destino (ID).

2. A nomeação é recebida através do nó EWP da ULisboa, para a instância FenixEdu da Reitoria que, por sua vez, encaminhará para a instância FenixEdu da ID (excepto IST)..

3. O Gabinete/Núcleo de Mobilidade (NM) da ID recebe a nomeação do aluno em mobilidade:

   (a) Nomeação realizada através de EWP pela IO.

   (b) Prazos propostos: Até 30 de Abril 1º ano/1º Semestre; 1 a 30 de Setembro, 2º Semestre.

4. O NM valida a nomeação conforme os acordos interinstitucionais e respectivos fluxos em vigor existentes, entre a IO e a ID.

5. Estando válido, o NM aceita a nomeação e é criada a conta de aluno.

6. A IO submete LA para validação/aprovação da ID:

   (a) A submissão do LA deverá ser feita através do EWP (pressupõe assinatura/aprovação prévia do aluno).

[1] Em BPMN, uma *pool* representa um participante no processo, podendo corresponder a uma entidade, organização ou sistema. Cada *pool* define um espaço de trabalho independente para processos que interagem entre si [12].

7. O LA é validado e aprovado pela ID.
8. É gerada automaticamente a Carta de Aceitação com os dados do aluno.
9. O aluno formaliza a matrícula e inscreve-se às UCs e aos horários pretendidos.
10. Início da mobilidade com início das aulas.
11. Emissão automática pelo Fenix (Documento Online) de uma declaração de chegada modelo, da ULisboa:
    (a) Algumas IO tem os seus próprios modelos de declarações, pelo que, o aluno também deverá poder solicitar a assinatura de outra declaração ao NM.
    (b) Caso o aluno assim o solicite, o NM assina as declarações das IO, submetidas pelo aluno.
12. A partir do início das aulas, os alunos têm cerca de 15 dias para alterações ao plano de estudos:
    (a) Caso exista alteração ao plano de estudos, seja por iniciativa do aluno, seja por limitação de vagas, seja por não abertura de UCs, etc., o aluno apenas tem que se inscrever/alterar UCs pretendidas de acordo com os processos administrativos existentes na ID e submeter posteriormente, o LA atualizado através do EWP.
    (b) Ao receber o LA através do EWP, o NM valida se as UCs às quais o aluno está efetivamente inscrito, correspondem ao que está no LA.
13. Criar registo de Mobilidade *Incoming* no Fenix (Registo de Alunos Inscritos e Diplomados do Ensino Superior (RAIDES)).
14. Decorre o período lectivo (1 ano ou 1 semestre).
15. Fim da mobilidade:
    (a) Emissão do ToR no Fenix - documento gerado automaticamente.
    (b) Emissão da declaração de saída/estada modelo da ULisboa após lançamento das notas - documento gerado automaticamente.
    (c) Tal como nas declarações de chegada, se o aluno solicitar a assinatura de uma declaração diferente, também aqui o NM deverá assinar a declaração submetida pelo aluno.
    (d) Dá-se por concluído o processo de mobilidade *Incoming*.

3.4.2 Comunicações EWP para Nomeações

Com base na análise detalhada dos repositórios do EWP, foi possível compreender a sequência de comunicações definida para a troca de dados de nomeações entre Instituições de Ensino Superior (IES). Esta sequência, ilustrada na figura 3.2, estabelece o fluxo de troca de informações

entre a Instituição de Origem (IO), a Instituição de Destino (ID) e o nó EWP, garantindo que as atualizações de dados de mobilidade sejam propagadas entre IES.

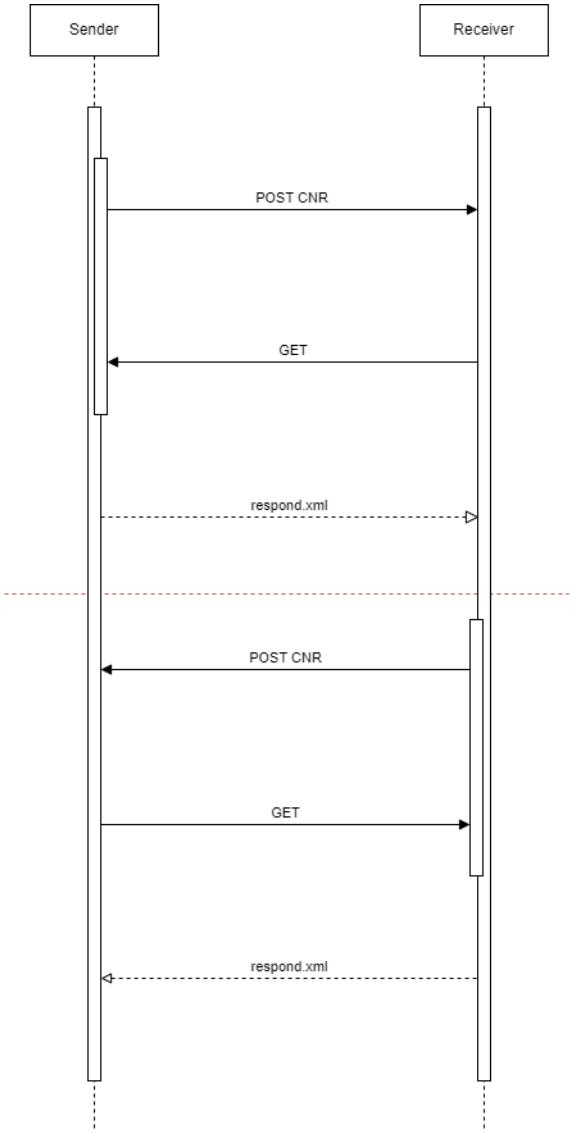


Figura 3.2: Interpretação das Comunicações de Nomeações definidas pelo EWP.

De referir que, à semelhança da simplificação aplicada na modelação em BPMN, a figura 3.2 não reflete integralmente a arquitetura definida para a ULisboa. A figura ilustra apenas o fluxo direto entre o remetente (*sender*) que, no caso, é o nó EWP, e o destinatário das comunicações (ID/*receiver*), assumindo este último como sendo a instância da Reitoria da ULisboa. A comunicação interna entre a instância da Reitoria e as instâncias das Escolas, isto é, o redirecionamento das nomeações pelo sistema Fenix da Reitoria para a Escola a que corresponde a nomeação, não está representada.

Abaixo, descreve-se o fluxo das principais etapas de comunicação:

1. A Instituição de Destino/Acolhimento (ID) recebe o *POST* do CNR[2] enviado pela IO. De momento, não é possível replicar esta comunicação, uma vez que esta se encontra fora do âmbito deste projeto e ainda não foi implementada;

2. Após receber o CNR, a ID realiza uma chamada *GET* ao nó do EWP - atualmente simulada através do separador importar - e obtém os dados das nomeações;

3. A ID envia um *POST* de um CNR para confirmar à IO a receção das nomeações;

4. A IO realiza um *GET* para ler os dados - sendo que, atualmente, o comportamento da IO não pode ser simulado;

5. A ID altera uma das nomeações (aceita, rejeita ou atualiza), e cada uma dessas ações despoleta um *POST* de um CNR para informar que foram realizadas alterações nomeações;

6. A IO realiza um *GET* para atualizar as informações - mais uma vez, o comportamento da IO não pode ser simulado nesta fase;

Com a análise das comunicações definidas pelo EWP, foi possível clarificar as ações que cada entidade (Instituição de Origem, EWP e Instituição de Destino) deve realizar em cada fase. Esta compreensão foi essencial para assegurar que o *workflow* desenvolvido em Fenix, seja totalmente compatível com o EWP, suportando as operações e trocas de dados necessárias para a interoperabilidade.

3.4.3 Diagrama de Entidades e Associações (DEA)

Após a modelação do processo de mobilidade e a análise das comunicações definidas pelo EWP, foi necessário projetar o modelo de domínio, representado pelo DEA. Este diagrama serve como base estrutural para definir as entidades e as interações essenciais no sistema, facilitando a integração com o fluxo de comunicações estabelecido pelo EWP.

Nesta fase, já dispúnhamos do modelo de processo em BPMN e da estrutura existente do domínio do módulo de mobilidade no sistema Fenix, fornecida pelo NDS e representada na figura E.1 do anexo E.1. No entanto, para a criação do DEA, foi necessário realizar uma análise detalhada dos repositórios do EWP, em particular das APIs de Nomeações *Incoming*[3] e *Outgoing*[4]. Isto foi fundamental, uma vez que as nomeações *incoming* na nossa instituição (Instituição de Destino) correspondem às *outgoing* da Instituição de Origem, tornando essencial compreender que dados seriam disponibilizados e quais as entidades, atributos e relações a preservar no nosso domínio.

[2] O CNR (*Change Notification Receiver*) é uma API concebida para notificar outros *Hosts* EWP quando algo é alterado (inserido, atualizado ou eliminado), em vez de exigir que os outros *Hosts* consultem periodicamente. Baseia-se no padrão de publicação-subscrição, que se integra bem com a arquitetura e a terminologia do EWP [18].

[3] Repositório disponível em: `https://github.com/erasmus-without-paper/ewp-specs-api-imobilities`

[4] Repositório disponível em: `https://github.com/erasmus-without-paper/ewp-specs-api-omobilities`

Devido à escassez de documentação até maio de 2024[5], foi necessária uma análise extensiva dos ficheiros `.xsd` e `.xml` nesses repositórios, identificando as entidades, atributos e associações para o nosso sistema. Esse processo revelou-se complexo, exigindo um considerável investimento de tempo para o levantamento completo.

Com base na informação estruturada obtida a partir dos repositórios, foi elaborado um conjunto de ficheiros de mapeamento, que resume a informação relevante para as APIs *Incoming* e *Outgoing* (ver figuras E.2 e E.3 no anexo E.2). A partir destes mapeamentos, foi possível elaborar uma primeira versão do DEA, representada na figura 3.3, onde estão identificadas as principais entidades e relações.

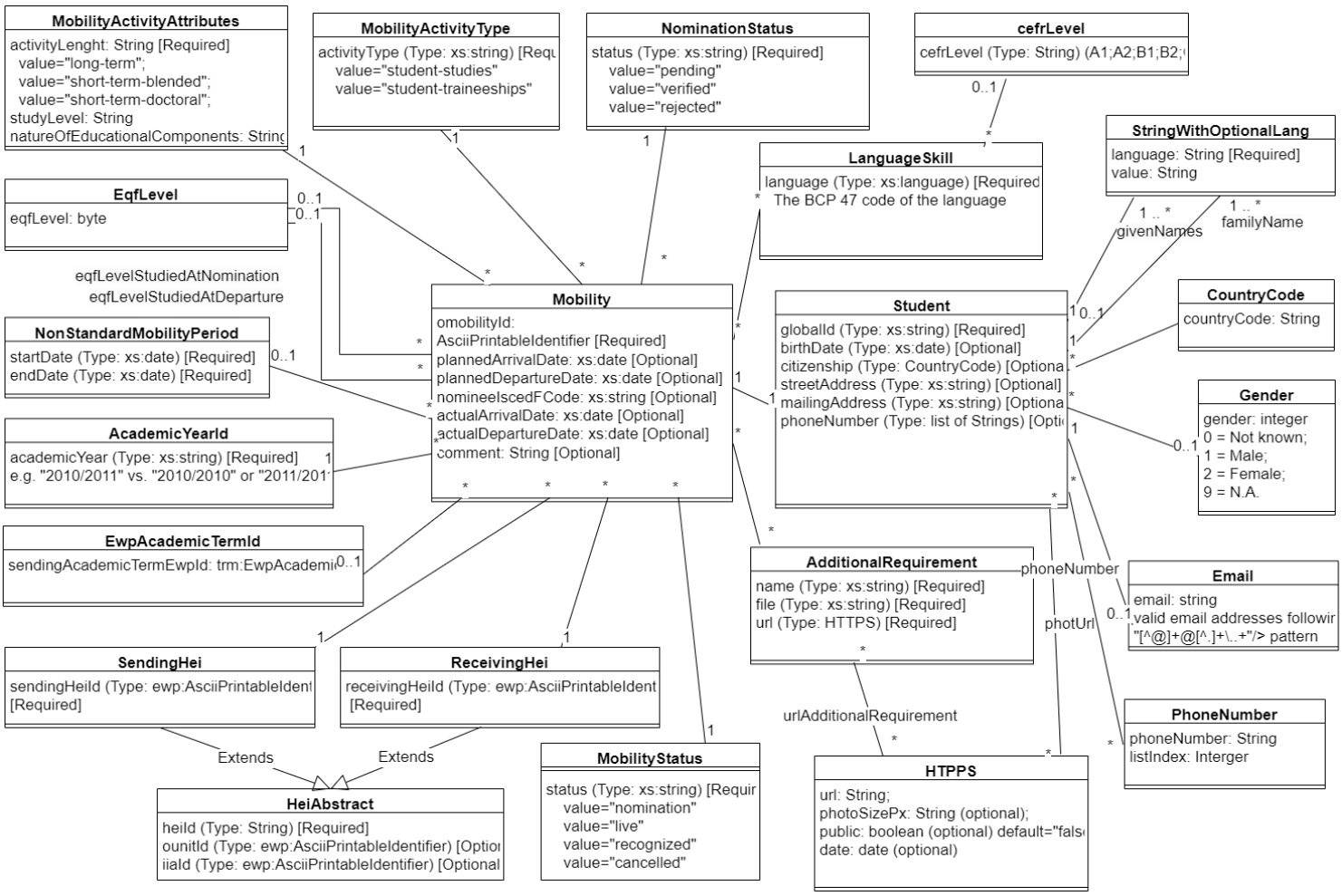


Figura 3.3: DEA Mobilidade *Incoming* - versão inicial.

[5]Commit d88243b no repositório GitHub *ewp-specs-api-omobilities*, disponível em `https://github.com/erasmus-without-paper/ewp-specs-api-omobilities`

A fase seguinte consistiu em cruzar este modelo com o domínio do módulo de mobilidade já existente no Fenix disponibilizado pelo NDS (figura E.1 do anexo E.1), com o objetivo de identificar as entidades que poderiam ser reutilizadas, minimizando redundâncias e assegurando a compatibilidade com as novas funcionalidades. Neste ponto, foi possível identificar quais as entidades que necessitavam de adaptações, quais poderiam ser reutilizadas e quais deveriam ser criadas de raiz.

Ao longo do desenvolvimento, o DEA passou por múltiplas iterações, sendo ajustado e refinado para refletir as necessidades emergentes e as mudanças na lógica de negócio. Uma das principais razões para a simplificação do modelo foi a evolução dos requisitos definidos pelos repositórios do EWP. As atualizações de versão, em particular a transição de V1 para V2 no *imobilities* e de V2 para V3 no *omobilities*, levaram à remoção de várias entidades e relações anteriormente obrigatórias. Essa mudança resultou numa simplificação significativa, reduzindo o número de entidades essenciais, sem comprometer os requisitos funcionais necessários.

O modelo final, representado na figura 3.4, reflete essa evolução, com um foco nas entidades principais da mobilidade *Incoming*: *Incoming Student*, *Incoming Student Nomination*, *Student Nomination Type* e *Higher Education Institution*. Embora mais simples do que a versão inicial, o modelo final mantém a conformidade com o EWP e atende a todas as necessidades funcionais e de interoperabilidade identificadas.

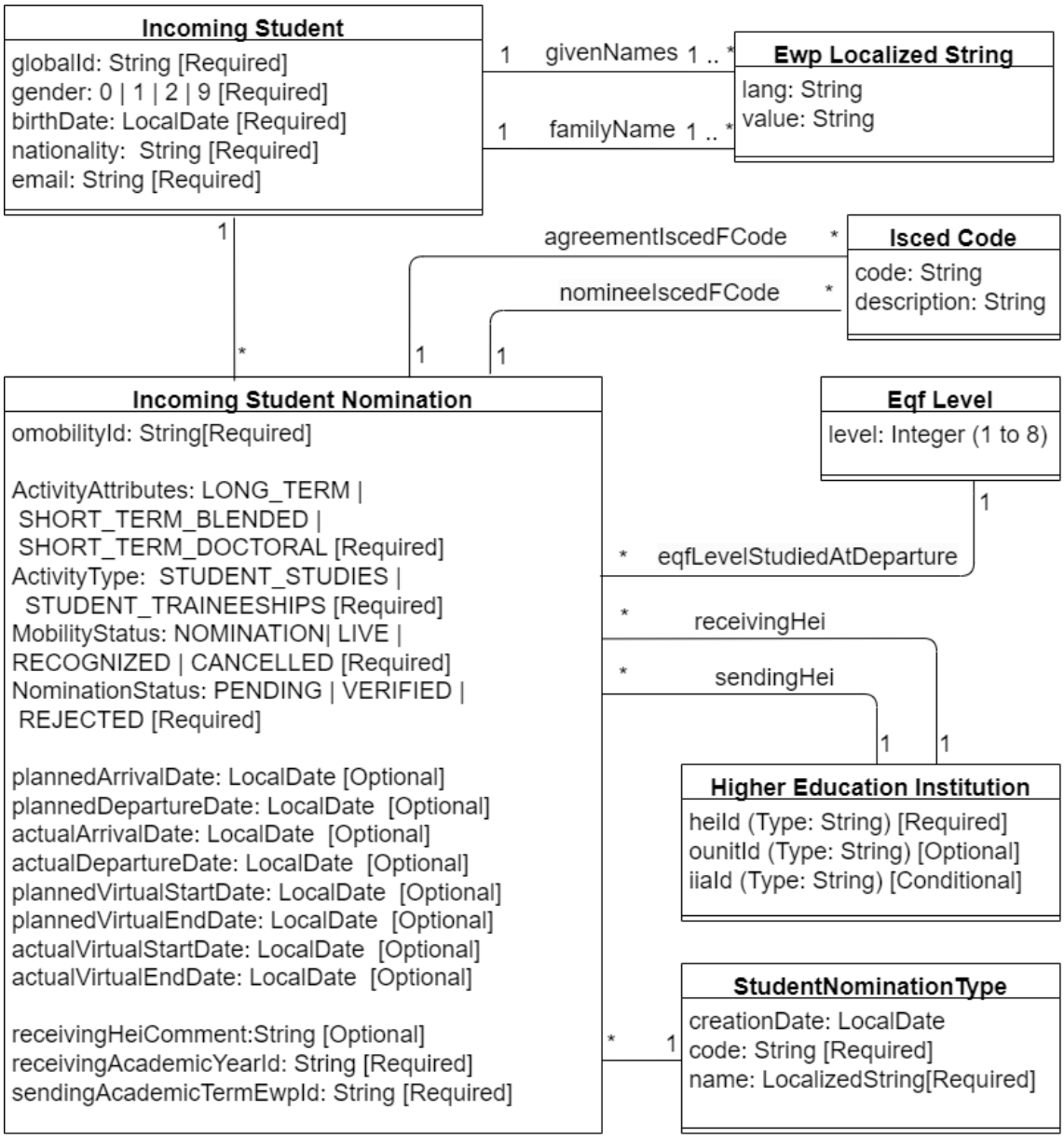


Figura 3.4: DEA Mobilidade *Incoming* - versão atual.

3.5 Sumário

O terceiro capítulo abrange o levantamento de requisitos e o desenho do projeto. Iniciámos com os conceitos essenciais sobre mobilidade e o EWP, explorando os processos e requisitos funcionais e não funcionais identificados. Foram mapeados os principais intervenientes e desafios a considerar.

Definimos a arquitetura do sistema, seguida do desenho do processo de mobilidade *incoming* em BPMN. Analisámos também as comunicações entre IES conforme as especificações do EWP, e concluímos com o DEA, que fornece a base para a modelação do domínio e o desenvolvimento do *workflow* na *framework* OMNIS.

O próximo capítulo apresentará a implementação do sistema e a sua avaliação.

Capítulo 4

Implementação e Avaliação

Neste capítulo, detalhamos os desenvolvimentos realizados e a avaliação da solução proposta. Na primeira parte, abordamos os principais componentes implementados, desde a configuração do ambiente de desenvolvimento, a modelação do domínio, a estruturação da interface, a modelação do *workflow*, até à geração de documentos automáticos, definição dos perfis de acesso e implementação das operações de aceitação, rejeição e atualização de nomeações rejeitadas. Na segunda parte, apresentamos a metodologia de avaliação e os resultados obtidos, destacando os avanços alcançados e as limitações identificadas.

4.1 Implementação

Descrevemos aqui os desenvolvimentos realizados, começando pela configuração do ambiente de desenvolvimento e pelas extensões ao domínio aplicacional. Em seguida, detalhamos a interface, a modelação do processo de mobilidade *Incoming* através do *workflow* no sistema Fenix, e as operações desenvolvidas para suportar as transições entre estados e as comunicações com o EWP. Finalizamos com a integração da geração de documentos automáticos e a definição dos perfis de acesso, garantindo que as permissões refletem as necessidades do projeto.

4.1.1 Ambiente de Desenvolvimento

O desenvolvimento do módulo de mobilidade foi realizado num ambiente configurado para garantir a compatibilidade com o sistema Fenix. Para isso, utilizou-se uma máquina virtual com o sistema operativo *Ubuntu*, onde foi instalada uma versão personalizada do IDE Eclipse, devido à sua integração com as ferramentas da *framework* OMNIS. Para além do Eclipse, foram configuradas as tecnologias *Git*, *Maven*, *Docker* e *Tomcat*, descritas anteriormente no ponto 2.4.

Após a configuração do ambiente, seguindo as indicações da equipa de desenvolvimento, clonaram-se dois repositórios: um do projeto específico do módulo de mobilidade onde são adicionados os desenvolvimentos, e o outro da *WebApp*, referente à instância do sistema Fenix da Reitoria da ULisboa e utilizado para testar localmente as funcionalidades à medida que eram desenvolvidas. Para isso, foi também adicionada a dependência do projeto de mobilidade à POM do projeto da *WebApp*, permitindo assim a integração das novas funcionalidades com o resto do

sistema. Esta configuração permitiu uma rápida iteração e validação das funcionalidades implementadas à medida que iam sendo desenvolvidas, garantindo que a funcionalidade do sistema Fenix, ia sendo assegurada.

4.1.2 Domínio

Com o DEA concluído e o ambiente de desenvolvimento configurado, o próximo passo consistiu em traduzir as entidades e relações identificadas para a DSL do domínio do sistema, com base no modelo descrito na secção 3.4.3. Dado que o módulo de mobilidade já possuía uma DSL pré-existente, este foi complementado com novas entidades e relações necessárias para a funcionalidade de mobilidade *Incoming*. Este processo foi iterativo, refletindo ajustes às alterações de requisitos e domínio.

A figura 4.1 apresenta um excerto da DSL, que inclui as principais entidades do módulo de Mobilidade *Incoming*: *UlMobInStudentNomination*, *UlMobInStudent* e *UlMobInHei*, juntamente com os respetivos atributos e relações.

Após a atualização da DSL, o projeto foi recompilado, gerando novas classes que foram posteriormente enriquecidas com métodos para implementar as regras de negócio. A figura 4.2 exemplifica a classe *UlMobInStudentNomination*, incluindo o construtor e métodos *create* para criar instâncias, quer através de atributos específicos, quer utilizando dados recebidos via DTO.

No âmbito do projeto, as nomeações são criadas exclusivamente através de dados recebidos do nó EWP, impedindo instâncias manuais via interface. Esta restrição é aplicada através de métodos como *createFromDTO*, garantindo a conformidade com os requisitos de integração. Adicionalmente, a ausência de funcionalidades no *frontend* para criação manual reforça esta abordagem, como será detalhado na secção 4.1.3.

Foram ainda desenvolvidos métodos para gestão e manipulação de objetos, tais como *delete()*, *findByHeiId(String)*, *findByOmobilityId(String)* e *edit(args)*. Para operações envolvendo DTOs, foram implementados métodos específicos, incluindo *convertActivityAttributesFromDTO*, *convertMobilityStatusFromDTO* e *editFromDTO*, assegurando a consistência entre os dados recebidos e os do domínio.

fenixedu-ulisboa-student-mobility.dsl

```
(...)
//Incoming Nomination
entity mobility.incoming.UlMobInStudentNomination channels (WebJava) {
    //Attributes
    required String omobilityId;
    required String sendingAcademicTermEwpId;
    required String receivingAcademicYearId;
    required UlMobInActivityAttributes activityAttributes;
    required UlMobInActivityType activityType;
    required UlMobInMobilityStatus mobilityStatus;
    required UlMobInNominationStatus nominationStatus;
    LocalDate plannedArrivalDate;
    LocalDate plannedDepartureDate;
    LocalDate actualArrivalDate;
    LocalDate actualDepartureDate;
    LocalDate plannedVirtualStartDate;
    LocalDate plannedVirtualEndDate;
    LocalDate actualVirtualStartDate;
    LocalDate actualVirtualEndDate;
    String receivingHeiComment;
    Boolean previouslyRejected;
    //Relationships
    oneToOne WorkflowInstance workflowInstance mappedBy ulMobInStudentNomination;
    required manyToOne UlMobInStudentNominationType studentNominationType;
    required manyToOne UlMobInHei sendingHei;
    required manyToOne UlMobInHei receivingHei;
    required manyToOne UlMobInStudent incomingStudent;
    required manyToOne UlMobIscedCode nomineeIscedFCode;
    required manyToOne UlMobIscedCode agreementIscedFCode;
    required manyToOne UlMobEqfLevel eqfLevelStudiedAtDeparture;
    manyToOne DomainRoot domainRoot;
}
//Incoming Student
entity mobility.incoming.student.UlMobInStudent channels (WebJava) {
    //Attributes
    required String globalId;
    required LocalDate birthDate;
    required String nationality;
    required String email;
    required String gender;
    //Relationships
    oneToMany UlMobInStudentNomination ulMobInStudentNominations mappedBy incomingStudent;
    oneToMany UlMobEwpLocalizedString givenNames mappedBy givenNames;
    oneToMany UlMobEwpLocalizedString familyName mappedBy familyName;
    oneToOne Person person mappedBy UlMobInStudent;
    manyToOne DomainRoot domainRoot;
}
//HEIs
entity mobility.incoming.hei.UlMobInHei channels (WebJava) {
    //Attributes
    required String heiId;
    String iiaId;
    String ounitId;
    //Relationships
    oneToMany UlMobInStudentNomination sentNomination mappedBy sendingHei;
    oneToMany UlMobInStudentNomination receivedNomination mappedBy receivingHei;
    manyToOne DomainRoot domainRoot;
}
(...)
```

Figura 4.1: Excerto da DSL do módulo de Mobilidade.

UlMobInStudentNomination.java

```java
/**
 * This class represents an incoming student nomination for mobility.
 * It contains methods to manage and manipulate the nomination data.
 */
public class UlMobInStudentNomination extends UlMobInStudentNomination_Base implements InvariantCheck {

    public UlMobInStudentNomination() {
        setDomainRoot(pt.ist.fenixframework.FenixFramework.getDomainRoot());
    }
    /**
     * Constructor with parameters to create an UlMobInStudentNomination instance.
     */
    protected UlMobInStudentNomination(final String omobilityId, final UlMobInStudent ulMobInStudent,
            final UlMobInActivityAttributes ulMobInActivityAttributes, final UlMobInActivityType ulMobInActivityType,
            final UlMobInMobilityStatus ulMobInMobilityStatus, final UlMobIscedCode agreementIscedFCode,
            final UlMobIscedCode nomineeIscedFCodeCode, final UlMobEqfLevel eqfLevelStudiedAtDepartureLevel,
            final UlMobInStudentNominationType ulMobInStudentNominationType, final ApplicationUser responsible,
            final UlMobInHei sendingHei, final UlMobInHei receivingHei, final String sendingAcademicTermEwpId,
            final String receivingAcademicYearId) {

            this();
            setOmobilityId(omobilityId);
            setIncomingStudent(ulMobInStudent);
            setActivityAttributes(ulMobInActivityAttributes);
            setActivityType(ulMobInActivityType);
            setMobilityStatus(ulMobInMobilityStatus);
            setNominationStatus(UlMobInNominationStatus.PENDING);
            setPreviouslyRejected(false);
            setNomineeIscedFCode(nomineeIscedFCodeCode);
            setEqfLevelStudiedAtDeparture(eqfLevelStudiedAtDepartureLevel);
            setStudentNominationType(ulMobInStudentNominationType);
            setWorkflowInstance(ulMobInStudentNominationType.createWorkflowInstance(responsible));
            initWorkflowRuntimeConfiguration(responsible, getCurrentExecutionYear());
            setSendingHei(sendingHei);
            setReceivingHei(receivingHei);
            setSendingAcademicTermEwpId(sendingAcademicTermEwpId);
            setReceivingAcademicYearId(receivingAcademicYearId);
    }
    /**
     * Creates a new UlMobInStudentNomination instance.
     */
    @Atomic
    public static UlMobInStudentNomination create(final String omobilityId, final UlMobInStudent ulMobInStudent,
            final UlMobInActivityAttributes ulMobInActivityAttributes, final UlMobInActivityType ulMobInActivityType,
            final UlMobInMobilityStatus ulMobInMobilityStatus, final UlMobIscedCode agreementIscedFCode,
            final UlMobIscedCode nomineeIscedFCodeCode, final UlMobEqfLevel eqfLevelStudiedAtDepartureLevel,
            final UlMobInStudentNominationType ulMobInStudentNominationType, final ApplicationUser responsible,
            final UlMobInHei sendingHei, final UlMobInHei receivingHei, final String sendingAcademicTermEwpId,
            final String receivingAcademicYearId) {

        return new UlMobInStudentNomination(omobilityId, ulMobInStudent, ulMobInActivityAttributes, ulMobInActivityType,
                ulMobInMobilityStatus, agreementIscedFCode, nomineeIscedFCodeCode, eqfLevelStudiedAtDepartureLevel,
                ulMobInStudentNominationType, responsible, sendingHei, receivingHei, sendingAcademicTermEwpId,
                receivingAcademicYearId);
    }
    /**
     * Creates a new UlMobInStudentNomination instance from a DTO.
     */
    public static UlMobInStudentNomination createFromDTO(final UlMobInStudentNominationDTO incomingNominationDTO) {
        Optional<UlMobIscedCode> nomineeIscedFCodeCodeOptional = Optional.empty();
        if (incomingNominationDTO.getNomineeIscedFCode() != null) {
            nomineeIscedFCodeCodeOptional = UlMobIscedCode.findByCode(incomingNominationDTO.getNomineeIscedFCode());
        }
        Optional<UlMobEqfLevel> eqfLevelStudiedAtDepartureLevelOptional = Optional.empty();
        if (incomingNominationDTO.getNomineeIscedFCode() != null) {
            eqfLevelStudiedAtDepartureLevelOptional =
                UlMobEqfLevel.findByLevel(incomingNominationDTO.getEqfLevelStudiedAtDeparture().intValue());
        }
        Optional<UlMobInStudentNominationType> nominationTypeOptional = Optional.empty();
        nominationTypeOptional = UlMobInStudentNominationType.findNominationTypeFromDTO(incomingNominationDTO);

        return UlMobInStudentNomination.create(incomingNominationDTO.getOmobilityId(),
                UlMobInStudent.handleEwpIncomingStudent(incomingNominationDTO.getIncomingStudent()),
                convertActivityAttributesFromDTO(incomingNominationDTO), convertActivityTypeFromDTO(incomingNominationDTO),
                convertMobilityStatusFromDTO(incomingNominationDTO), null,
                nomineeIscedFCodeCodeOptional.isEmpty() ? null : nomineeIscedFCodeCodeOptional.get(),
                eqfLevelStudiedAtDepartureLevelOptional.isEmpty() ? null : eqfLevelStudiedAtDepartureLevelOptional.get(),
                nominationTypeOptional.isEmpty() ? null : nominationTypeOptional.get(),
                ApplicationUser.getCurrentApplicationUser(), UlMobInHei.handleEwpHei(incomingNominationDTO.getSendingHei()),
                UlMobInHei.handleEwpHei(incomingNominationDTO.getReceivingHei()),
                incomingNominationDTO.getSendingAcademicTermEwpId(), incomingNominationDTO.getReceivingAcademicYearId());
    }
```

Figura 4.2: Excerto da Classe *UlMobInStudentNomination*.

Por último, a figura 4.3 apresenta o modelo de classes atual baseado no DEA ilustrado na figura 3.4, destacando, a cinzento, as entidades principais do domínio de mobilidade *Incoming* e, a azul, as entidades existentes no Fenix que foram ajustadas para incluir relações com as novas classes.

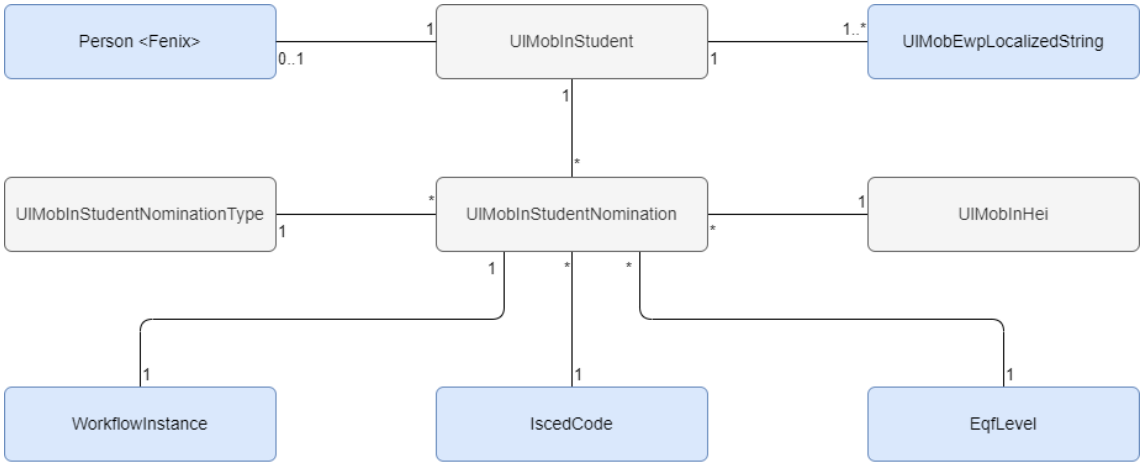


Figura 4.3: Modelo de Classes de Domínio da Mobilidade *Incoming* - versão atual.

4.1.3 Interface

Após a criação das classes do domínio e do *workflow*, foi necessário integrá-las na interface. Por uma questão de consistência de desenvolvimento, as instâncias Fenix seguem um padrão uniforme: na horizontal, no topo da página, encontram-se os módulos, e na vertical, no lado esquerdo, estão dispostos os Menus de cada Módulo, com os respetivos Submenus.

A figura 4.4 ilustra esta disposição, mostrando o módulo de mobilidade no canto superior direito e o submenu específico do projeto, *Incoming*, no lado esquerdo. Este menu possui quatro submenus: *Gestão de Alunos*, *Gestão de Nomeações*, *Tipos de Nomeação* e *Execuções de Nomeações*.

Os dois primeiros referem-se à gestão de alunos e nomeações *incoming*. O submenu *Tipos de Nomeação* permite definir os tipos de nomeação a receber pelo EWP (atualmente SMS e SMP), enquanto o submenu *Execuções de Nomeações* associa tipos de nomeação aos *workflows* e aos períodos de execução desejados (exemplificado na figura 4.5), permitindo receber nomeações via EWP.

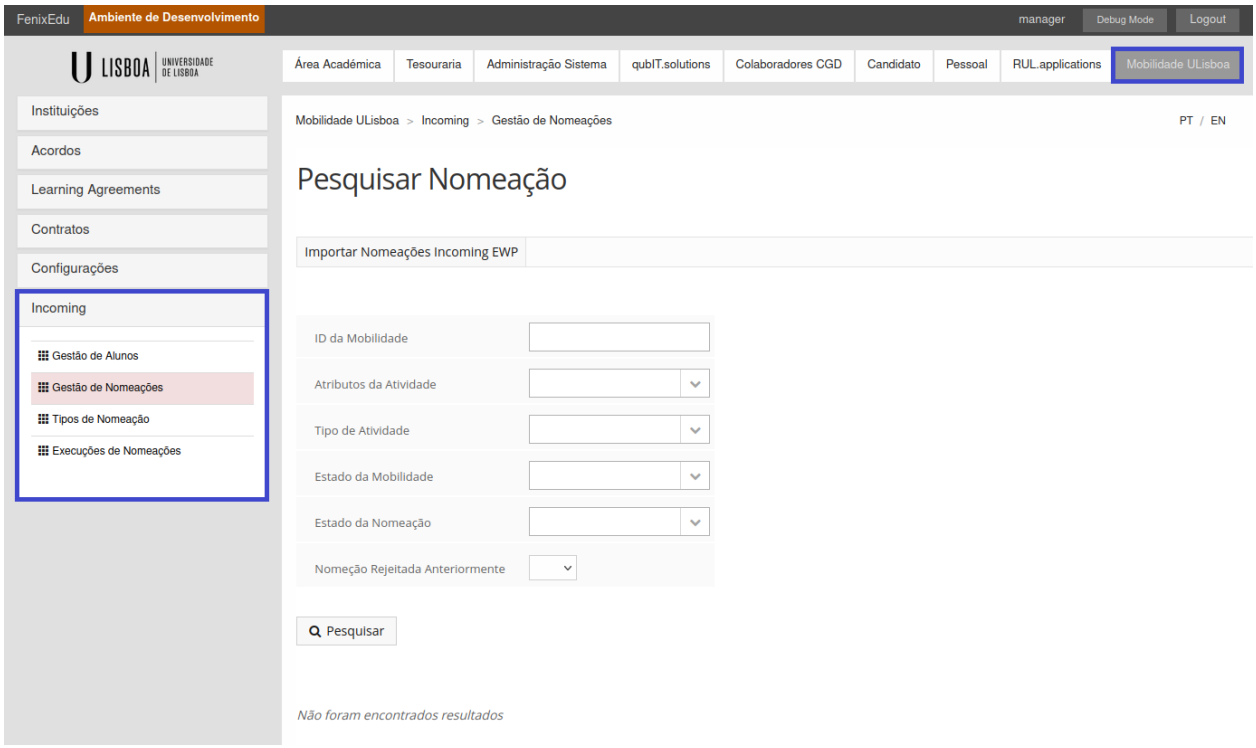


Figura 4.4: Vista Módulo Mobilidade: Menu *Incoming*.

De salientar que o módulo de Mobilidade já existia, e apenas o menu *Incoming* e os seus sub-menus foram criados e desenvolvidos no âmbito deste projecto, e adicionados ao módulo existente.

Figura 4.5: Execuções de Nomeações: Detalhe do tipo *Estudos*.

Todos os ecrãs que foram desenvolvidos para *interface* são do tipo *ReadScreen*, *UpdateScreen*, *SearchScreen* ou *CreateScreen*, e as suas bases foram obtidas pela definição e compilação da classe

da PSL, conforme descrito anteriormente no ponto 2.6.2.

Para o melhor ilustrar, a figura 4.6 exibe um excerto de uma das classes gerada pela PSL e que foi posteriormente adaptada conforme necessário. Neste caso particular, trata-se de um ecrã do tipo *SearchScreen* referente ao ecrã do submenu *Gestão de Nomeações*, onde é possível observar, por exemplo, os métodos que determinam se o utilizador tem permissões para importar ou apagar nomeações, ou que resultados são apresentados com o método *getSearchUniverse()*. A figura 4.7 contém um excerto da PSL.

SearchIncomingStudentNomination.java

```
(...)

@ScreenInfo(
        url = "com.qubit.solution.fenixedu.module.fenixeduUlisboaStudentMobility.presentation.
        screens.mobility.incoming.student.nomination.SearchIncomingStudentNomination")
public class SearchIncomingStudentNomination extends SearchIncomingStudentNomination_Base {
    @Inject
    private AccessControlService accessControlService;

    @Override
    protected void processDeleteAction(final UlMobInStudentNomination object) {
        object.delete();
    }

    @Override
    protected boolean isSearchActiveOnStart() {
        return true;
    }

    @Override
    protected void buildUI() {
        super.buildUI();

        getDeleteAction()
                .visible(accessControlService.hasPermission(Permissions.ADMIN_MOBILITY_INCOMING_NOMINATION_PERMISSION_CODE));

        getImportEwpIncomingNominationsEvent()
                .visible(accessControlService.hasPermission(Permissions.BACKOFFICE_MOBILITY_INCOMING_NOMINATION_PERMISSION_CODE)
                        || accessControlService.hasPermission(Permissions.ADMIN_MOBILITY_INCOMING_NOMINATION_PERMISSION_CODE));

        getSearchForm().setVisible(
                accessControlService.hasPermission(Permissions.BACKOFFICE_MOBILITY_INCOMING_NOMINATION_PERMISSION_CODE)
                        || accessControlService.hasPermission(Permissions.ADMIN_MOBILITY_INCOMING_NOMINATION_PERMISSION_CODE));
    }

     @Override
    protected Set<UlMobInStudentNomination> getSearchUniverse() {
        final Person authenticatedUsed = Authenticate.getUser().getPerson();

        if (accessControlService.hasPermission(Permissions.BACKOFFICE_MOBILITY_INCOMING_NOMINATION_PERMISSION_CODE)
                || accessControlService.hasPermission(Permissions.ADMIN_MOBILITY_INCOMING_NOMINATION_PERMISSION_CODE)) {
            return UlMobInStudentNomination.findAll().stream().collect(Collectors.toSet());
        }

        return UlMobInStudentNomination.findAll().stream().filter(r -> r.getWorkflowInstance().getActiveState()
                .getMembersWithAccess().contains(BaseUtil.toAppUser(authenticatedUsed))).collect(Collectors.toSet());
    }

    (...)
```

Figura 4.6: Excerto da classe do ecrã *SearchScreen*: *Gestão de Nomeações*.

incoming-student-nomination.psl

```
(...)
//Gestão de Nomeações
flow mobility.incoming.student.nomination channel (WebJava) {

    searchScreen searchIncomingStudentNomination  [UlMobInStudentNomination]
    (fields omobilityId activityAttributes activityType mobilityStatus nominationStatus previouslyRejected)
    (searchFields omobilityId activityAttributes activityType mobilityStatus nominationStatus previouslyRejected) {
        dialog event importEwpIncomingNominations -> importEwpIncomingNominations
        viewAction
        deleteAction -> searchIncomingStudentNomination
    }

    createScreen importEwpIncomingNominations [UlMobInHei] (fields heiId iiaId ounitId) {
        createEvent -> searchIncomingStudentNomination
        cancelEvent -> searchIncomingStudentNomination
    }
(...)
```

Figura 4.7: Excerto da PSL *Incoming Student Nomination*.

Conforme referido em 1.4, um dos principais objetivos deste projeto de Automação do Processo de Mobilidade Erasmus *Incoming* no Sistema Fenix é tornar o Fenix capaz de processar automaticamente informações de nomeações recebidas pela rede EWP. Assim, o objetivo é que o Fenix receba nomeações através do nó EWP e instancie automaticamente processos de nomeação, de acordo com os dados recebidos.

Por conseguinte, as nomeações apenas devem ser instanciadas a partir dessa informação e, no âmbito deste projecto, não deverá ser possível criá-las manualmente. Assim, nos menus de gestão de Alunos e de Nomeações, as entradas são criadas automaticamente com base nas informações recebidas do nó do EWP, não havendo qualquer botão que permita ao utilizador iniciar uma nomeação *incoming* manualmente, o que garante o cumprimento deste requisito e salvaguarda o modelo do EWP.

No caso dos alunos, estes são automaticamente criados ao receber uma nomeação, caso ainda não existam no sistema, como no caso de não terem sido previamente nomeados para outra mobilidade.

No caso das nomeações, e especificamente nesta fase de desenvolvimento, existe o separador *Importar Nomeações Incoming EWP* que permite a sua importação manual (visível na figura 4.4). Isto deve-se ao facto de, atualmente, não ser possível simular comunicações completas, uma vez que, à data, estas ainda não se encontram desenvolvidas. Apenas conseguimos simular uma chamada (GET) ao nó, que devolve dados pré-configurados manualmente por nós, permitindo simular a receção de nomeações *incoming* e testar o *workflow* e os desenvolvimentos realizados – essa chamada é simulada no separador *Importar Nomeações Incoming EWP*. No entanto, o objetivo é que, assim que estejam concluídos os desenvolvimentos complementares necessários (fora

do âmbito deste projecto), as nomeações passem a ser atualizadas automaticamente, dispensando a importação manual.

Assim, após receber uma ou mais nomeações novas, são criadas entradas no ecrã, com cada uma correspondendo a uma instância de *workflow*. Note-se que, à semelhança do que ocorre com os alunos, do ponto de vista do *frontend*, caso uma nomeação já existente seja recebida, a entrada não será duplicada; antes será atualizada. Isto é, naturalmente, assegurado nas classes de domínio, de acordo com as regras de negócio.

Uma vez criadas as novas nomeações recebidas e instanciados os respetivos *workflows*, este menu permite aceder às mesmas e executar as ações necessárias em função do estado do processo, como por exemplo, aprovar ou rejeitar nomeações (figura 4.8), solicitar documentos e emitir documentos (figura 4.9) sendo que este último veremos em detalhe de seguida, na secção 4.1.6.

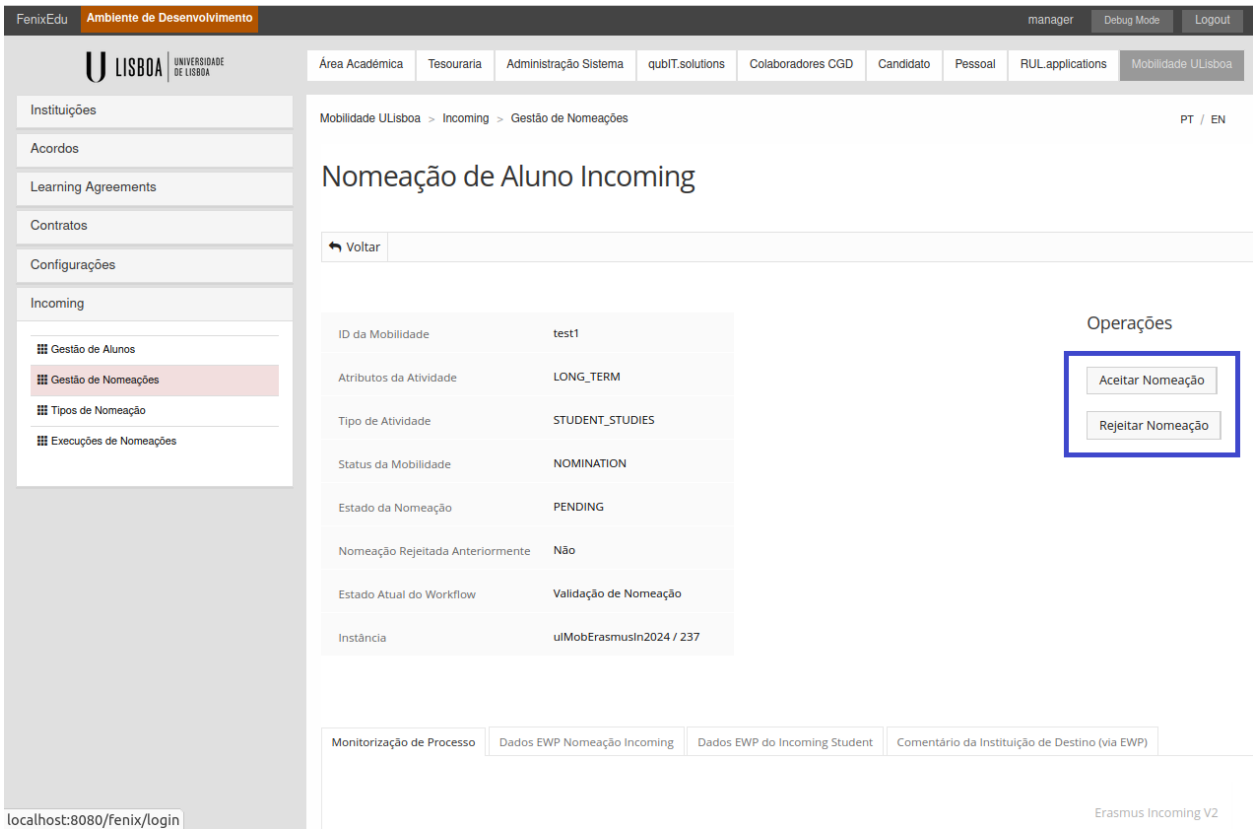


Figura 4.8: Exemplo de Operações para Aceitar ou Rejeitar nomeação, dentro de uma instância de *workflow*.

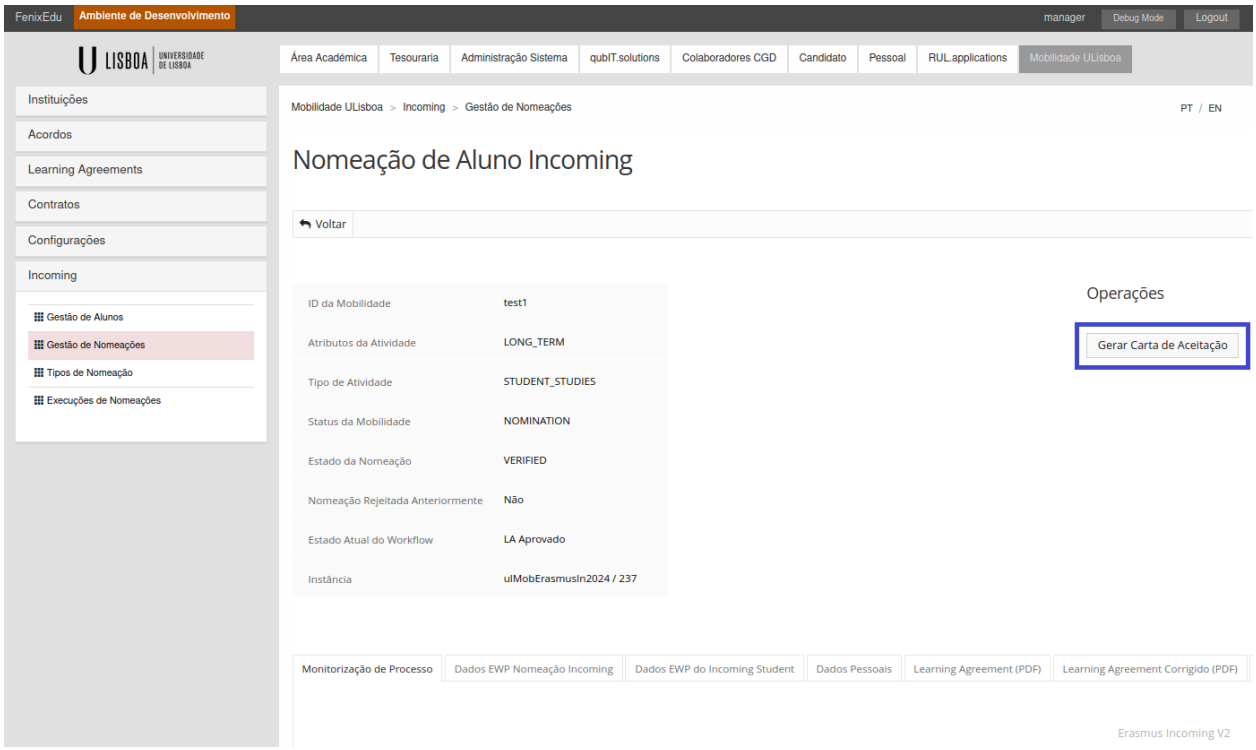


Figura 4.9: Exemplo de Operação para gerar documento online: Carta de Aceitação, dentro de uma instância de *workflow*.

Além destes menus, foram também desenvolvidos ecrãs adicionais no contexto do *workflow* desenvolvido, apresentados na forma de separadores e enumerados na figura 4.10. Neste caso, apenas os separadores de tipo *customizado* foram criados através da PSL e desenvolvidas as respetivas classes. Para os separadores de tipo *documento*, essa etapa de desenvolvimento não é necessária, uma vez que se trata de um tipo específico disponibilizado pela plataforma. Neste caso, estes separadores vão estando visíveis ou não, em função não apenas do estado do processo, como também do utilizador e do seu perfil de acesso – ponto que aprofundaremos de seguida em 4.1.7.

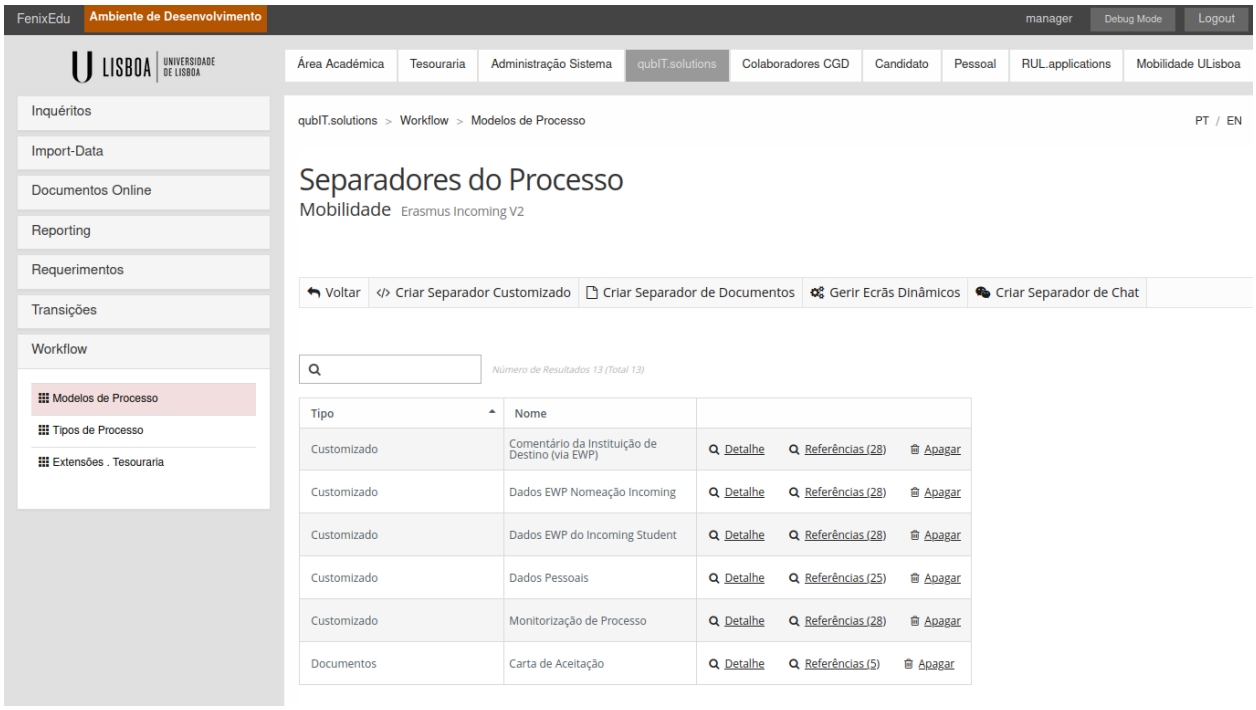


Figura 4.10: Excerto dos separadores do processo de Mobilidade *Incoming*.

Com a interface definida e implementada, podemos então avançar para a modelação do *workflow* do processo de mobilidade *incoming*.

4.1.4 Modelação do Processo de Mobilidade *Incoming*: *Workflow* Fenix

Com a interface implementada e os menus necessários acessíveis, o próximo passo foi transpor a modelação do processo de mobilidade *incoming*, previamente definida em BPMN (anexo C.1), para o *workflow* da *Framework Omnis* através da ferramenta de modelação de processos descrita no ponto 2.8, para que, tendo o processo modelado num *workflow*, seja possível avançar para a configuração e desenvolvimento dos estados, separadores e operações que o compõem.

A construção do *workflow* foi um processo iterativo, desenvolvido em colaboração com a equipa funcional, até se alcançar uma estrutura que reflete de forma satisfatória as necessidades do processo de Mobilidade *Incoming*. A figura 4.11 ilustra a parte inicial deste *workflow*, enquanto a representação completa encontra-se disponível no anexo D.1, com detalhe distribuído pelas figuras D.1 a D.5.

Como em qualquer processo, há um ponto inicial que, neste caso, é representado por um estado manual "Validação de Nomeação". Este é o estado inicial do nosso *workflow* e que representa a recepção de uma Nomeação *Incoming*, através do nó EWP para a ULisboa.

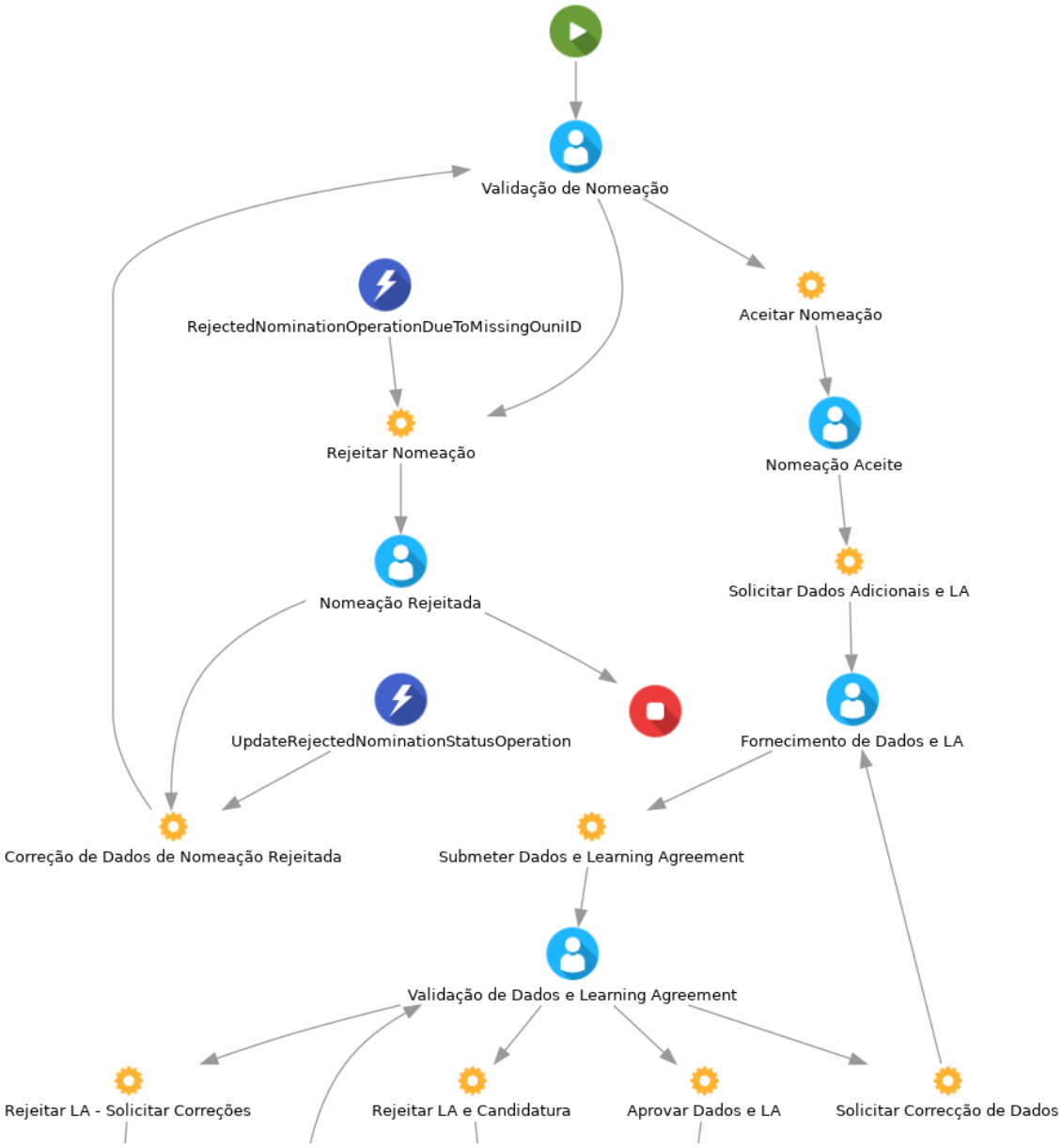


Figura 4.11: Início do *Workflow* de Mobilidade *Incoming*.

Como referido anteriormente, os estados manuais do *workflow* são compostos por separadores que contêm informações e, neste caso, este estado inicial é composto por 4 separadores:

- **Monitorização de Processo**: Este é o primeiro separador e permite aos intervenientes ter sempre presente uma representação visual do estado do processo. É um separador que estará presente e visível, ao longo de todo o *workflow* (Figura 4.12).

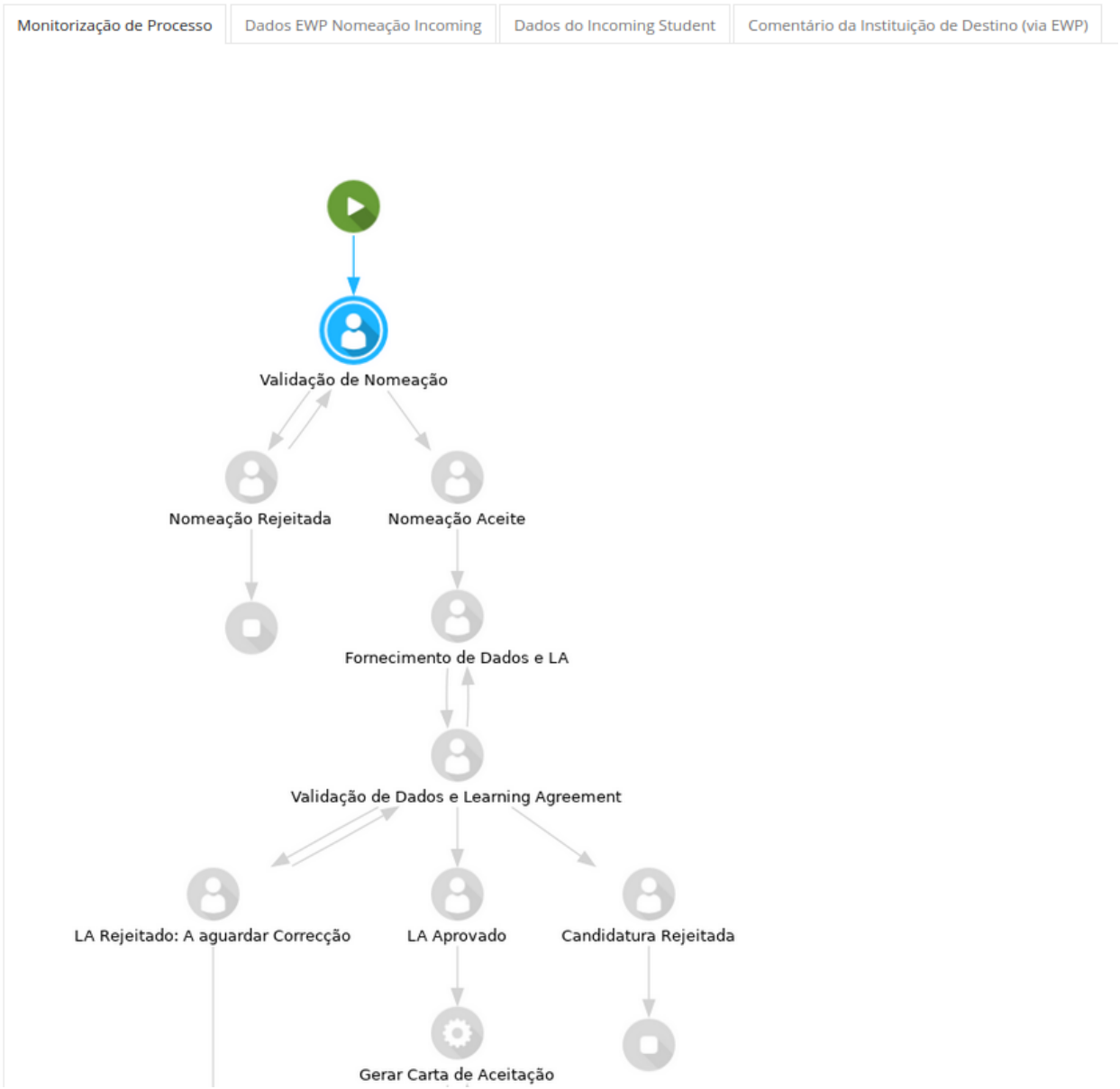


Figura 4.12: Detalhe do separador - *Monitorização de Processo*.

- **Dados EWP Nomeação Incoming**: Neste separador (Figura 4.13), estão refletidas as informações relativas à nomeação em questão e que foram recebidas via EWP, excepto as informações do aluno, que estarão no separador seguinte.

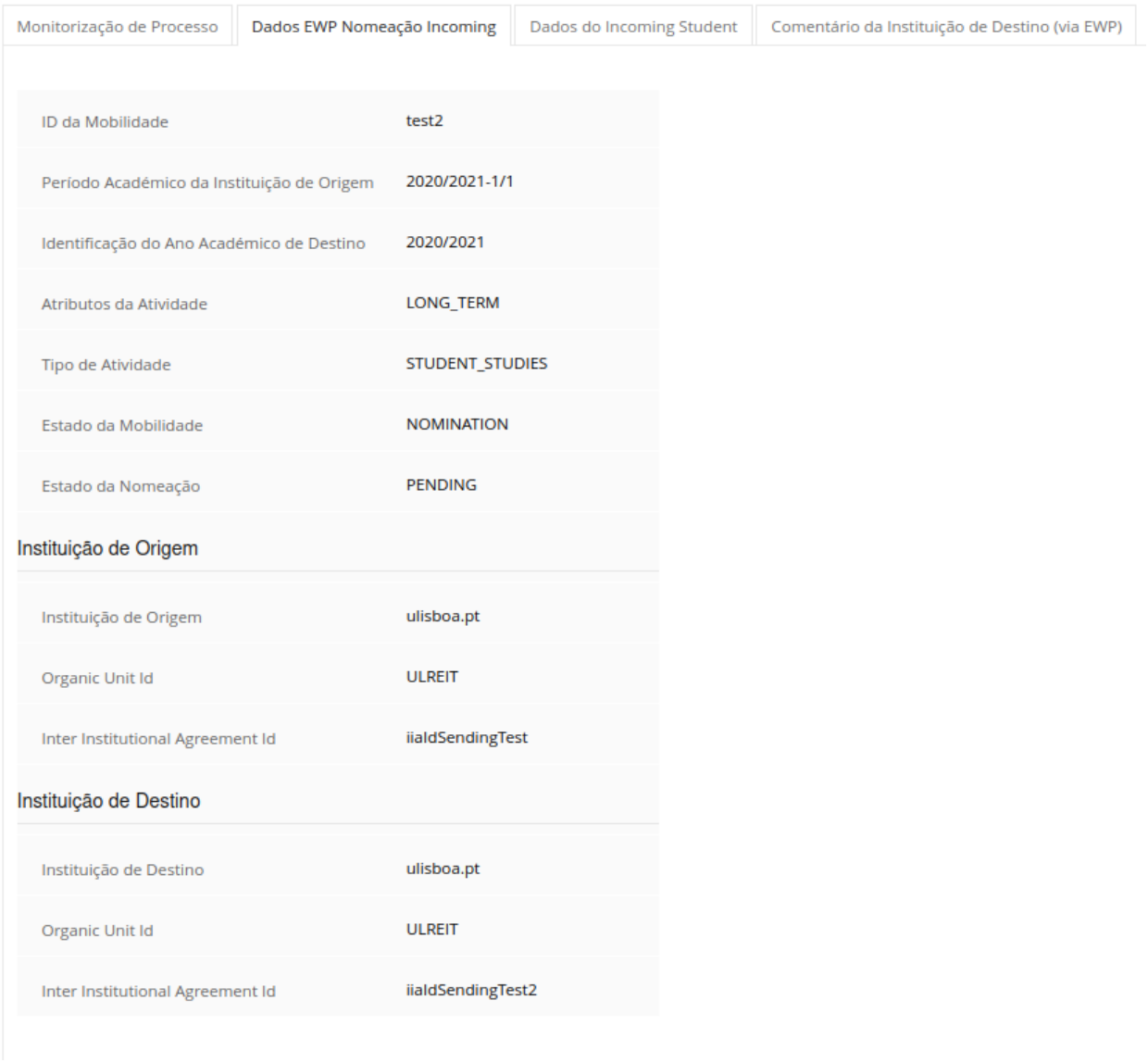


Figura 4.13: Detalhe do separador - *Dados EWP Nomeação Incoming*.

- **Dados do *Incoming Student***: Como o nome indica, é neste separador (Figura 4.14) que estão reunidas as informações do aluno *Incoming* recebidas até ao momento, via EWP.

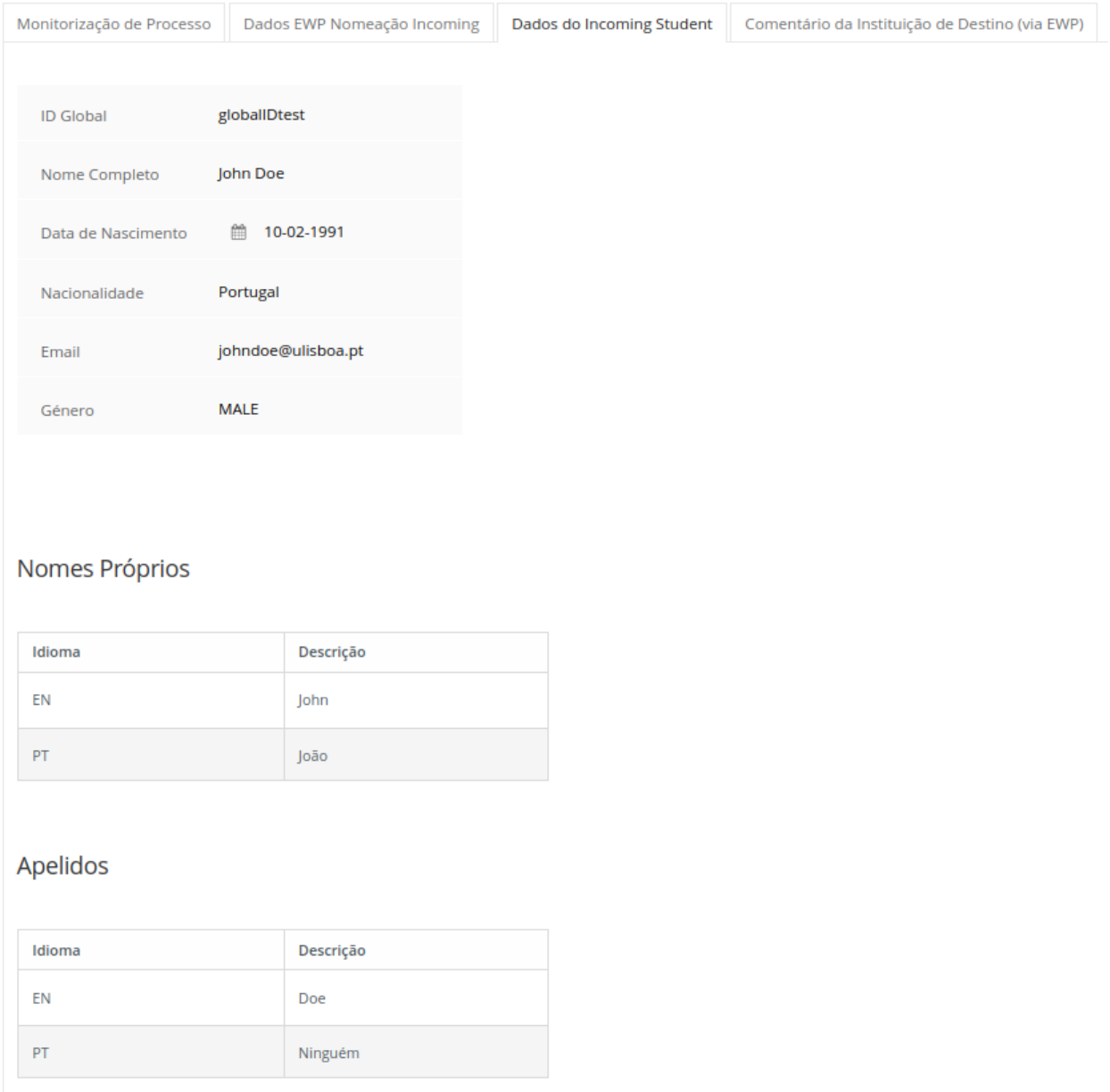


Figura 4.14: Detalhe do separador - *Dados do Incoming Student*.

- **Comentário da ID**: Por último temos um separador (Figura: 4.15) que permite a escrita de um comentário por parte do NM da ID. Caso a nomeação seja rejeitada, este campo é de preenchimento obrigatório, devendo ser aqui indicado o motivo que levou à rejeição da nomeação, para que este possa ser transmitido pela rede EWP, à IO, responsável pela nomeação em questão.

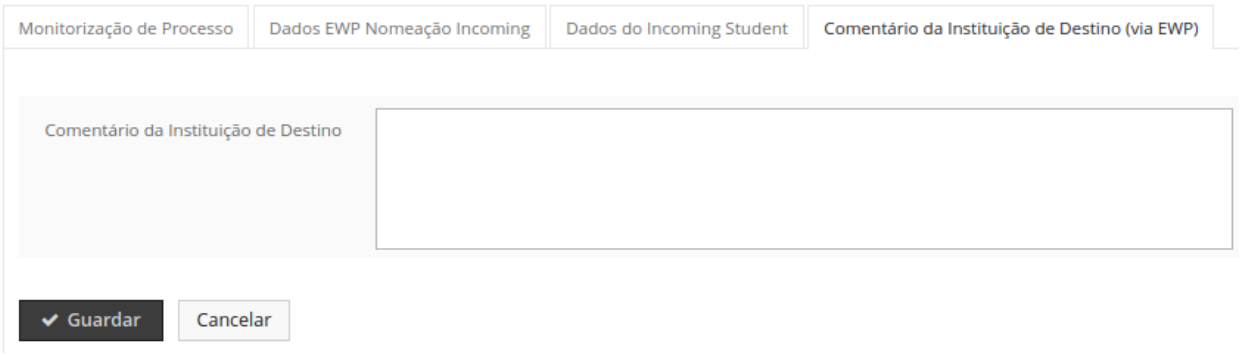


Figura 4.15: Detalhe do separador - *Comentário da ID (via EWP)*.

Para além dos separadores, este estado inicial tem também operações representadas por 2 botões e que permitem *Aceitar* ou *Rejeitar* a nomeação. Estas operações não só levarão a estados diferentes do processo, como também despoletarão comunicações para a rede EWP, por forma a informar a IO, acerca da decisão da ID relativamente à nomeação em causa. O detalhe deste estado inicial, com os separadores e operações referidos, encontra-se abaixo ilustrado na figura 4.16.

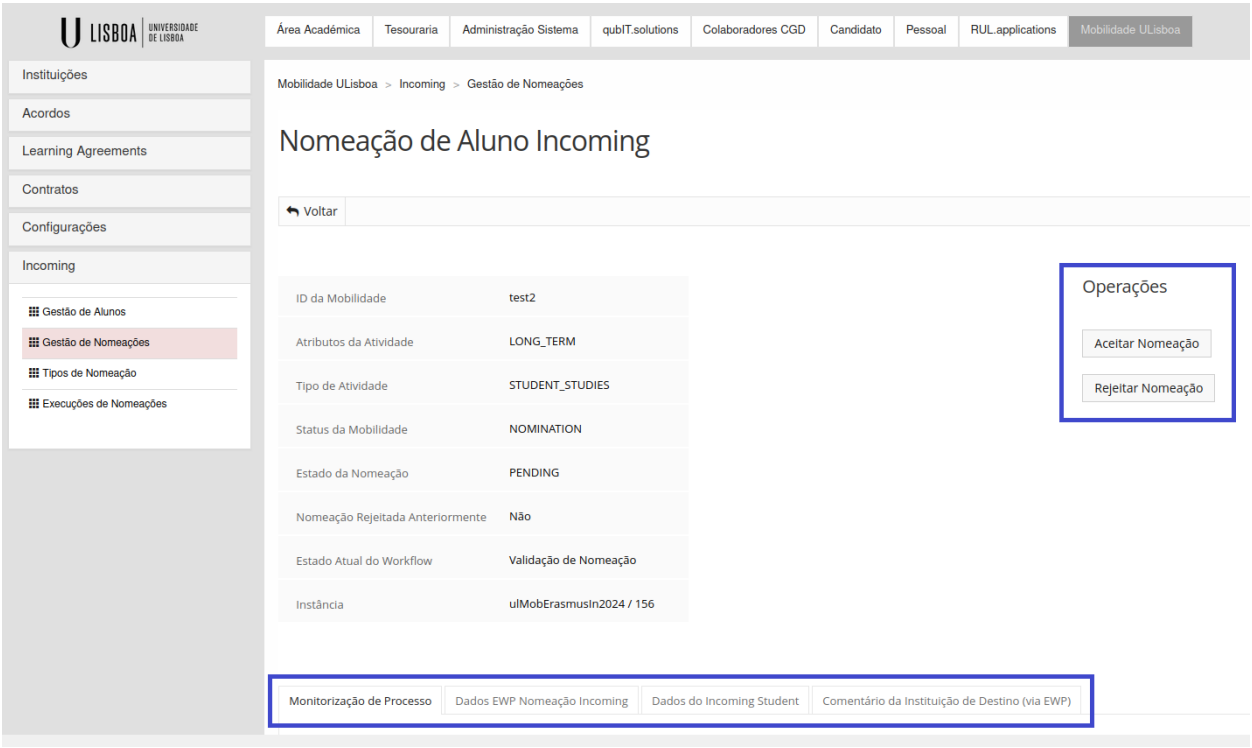


Figura 4.16: Detalhe estado inicial do *workflow* - Validação de Nomeação.

À medida que o processo vai avançando, os separadores vão sendo atualizados por forma a refletir a informação existente e relevante a cada momento. Neste caso, se a nomeação for rejeitada, não serão acrescentados separadores. Porém, caso a nomeação seja aceite, serão necessários dados e documentos adicionais do Aluno *Incoming*, como por exemplo o LA, e que levarão à disponibilização de novos separadores para que as informações necessárias possam ser fornecidas

ou disponibilizadas, como no caso da "Carta de Aceitação".

O detalhe desta informação foi sumarizado numa tabela ao longo do processo de desenvolvimento, para que seja facilmente possível saber que separadores e informação deverão existir em cada estado, que utilizadores deverão ter que tipo de permissões (leitura/escrita), que operações são possíveis executar a partir desse estado, e onde nos conduzirão entre outras. Para além disso, o ficheiro serve também como documentação para atuais e futuros utilizadores internos. Alguns excertos desta tabela estão disponíveis no Anexo D.2.

À data de conclusão deste documento, o *workflow* de mobilidade *Incoming* encontra-se integralmente modelado, abrangendo todas as fases do processo, desde a receção de uma nomeação até à conclusão da mobilidade do aluno. Os desenvolvimentos realizados nesta fase cobrem o percurso desde a receção da nomeação até ao estado de geração e assinatura da Carta de Aceitação, incluindo a implementação de estados, separadores e operações necessários para suportar estas etapas.

4.1.5 Operações para Integração com o EWP

Após a modelação do *workflow* e o desenvolvimento dos estados e separadores, foi necessário implementar operações que permitissem a transição entre estados e a execução de ações essenciais no âmbito das nomeações *incoming*. Embora a *Framework Omnis* já disponibilize operações genéricas, como a de *"Transitar de Estado"* ou a de *"Gerar Documento Online"* e cujos nomes já evidenciam a sua finalidade, o projeto requereu a criação de operações específicas que integrassem comunicações com o nó EWP. Este desenvolvimento foi fundamental para assegurar a automação das nomeações e a conformidade com os requisitos do projecto.

As operações desenvolvidas — *UlMobInAcceptNominationOperation*, *UlMobInRejectNominationOperation* e *UlMobInUpdateRejectedNominationStatusOperation* — foram projetadas para lidar com os casos de aceitação, rejeição e atualização de nomeações rejeitadas.

Estas operações garantem não apenas a transição entre estados no *workflow*, mas também a comunicação com o EWP, realizando um *POST CNR* para notificar a IO sobre alterações realizadas à nomeação, conforme descrito em 3.4.2. No entanto, é importante destacar que os desenvolvimentos realizados nesta fase do projeto assumem, temporariamente, uma comunicação direta entre a instância da Escola e o nó do EWP. Adicionalmente, assume-se que a comunicação entre a instância da Reitoria e as instâncias das Escolas, utilizará o mesmo protocolo definido pelo EWP, garantindo uniformidade e consistência na troca de informações.

Este modelo direto, embora funcional, não reflete a arquitetura final prevista em 3.3 e ilustrada na figura 3.1, onde as Escolas comunicarão exclusivamente com a instância da Reitoria, que será responsável por gerir a comunicação com o nó do EWP. Porém, este modelo temporário permite desenvolver e validar corretamente as operações necessárias, assim como as transições de estado no *workflow*. Esta abordagem foi adoptada considerando que as comunicações entre instâncias serão desenvolvidas numa fase posterior, para ir de encontro à arquitetura prevista.

Aceitar Nomeação: A operação *UlMobInAcceptNominationOperation* é acionada através de um botão visível na interface, quando a ID decide aceitar uma nomeação. Um excerto dessa classe está ilustrado na figura 4.17 e, ao ser executada, esta operação:

- Atualiza o estado da nomeação para `VERIFIED`.
- Envia um *POST CNR* ao nó EWP, notificando a IO de uma alteração às informações da mobilidade em causa.
- Cria uma conta no sistema para o estudante, caso este ainda não esteja registado.
- Transita para o próximo estado no *workflow*, permitindo a continuação do processo.

UlMobInAcceptNominationOperation.java

```
(...)
@WorkflowComponent(type = "ULISBOA_STUDENT_MOBILITY")
public class UlMobInAcceptNominationOperation extends AbstractWorkflowOperation {
    private static final long serialVersionUID = 1L;
    @Override
    public void execute() {
        ServiceProvider.getService(TransactionalManager.class).executeInWriteContext(getClass().getName(), () -> {
            final UlMobInStudentNomination incomingStudentNomination =
                    getState().getWorkflowInstance().getUlMobInStudentNomination();
            // Atualiza o nomination status para VERIFIED
            incomingStudentNomination.setNominationStatus(UlMobInNominationStatus.VERIFIED);
            //POST CNR para comunicar alteração do Status
            InstanceCommunicationUtils.sendIncomingMobilityChangeNotificationToEWPNode(incomingStudentNomination);
            // Cria a Person se necessário e associa-a ao Student
            incomingStudentNomination.getIncomingStudent()
                    .setPerson(createPersonIfDoesNotAlreadyExist(incomingStudentNomination.getIncomingStudent()));
            // Transita de estado
            getOperation().applyDefaultTransition(getState(), getResponsible());
        });
    }
(...)
```

Figura 4.17: Excerto da classe *UlMobInAcceptNominationOperation.java* relativa à operação de *Aceitar Nomeação Incoming*.

Rejeitar Nomeação: A operação *UlMobInRejectNominationOperation* é também acionada através de um botão na interface, quando a ID decide rejeitar uma nomeação. Esta operação:

- Valida a existência de um comentário de rejeição, que é obrigatório.
- Atualiza o estado da nomeação para `REJECTED`.
- Envia um *POST CNR* ao nó EWP, notificando a IO da rejeição.
- Transita para o estado correspondente no *workflow*.

Atualizar Nomeação Rejeitada: A operação *UlMobInUpdateRejectedNominationStatusOperation* difere das anteriores, pois não pode ser acionada manualmente. Esta operação é despoletada automaticamente pelo sistema ao receber uma nomeação cujo *OmobilityID* já exista no domínio e tenha sido previamente rejeitada. Ao ser executada, esta operação:

- Define o atributo `previouslyRejected` para `true`.
- Atualiza o estado da nomeação para `PENDING`.
- Envia um *POST CNR* ao nó EWP, notificando a IO da atualização.
- Retorna ao estado inicial do *workflow* (*"Validação de Nomeação"*) para reiniciar o processo de avaliação.

Esta restrição, para que a operação de atualização seja exclusivamente automática, assegura a coerência dos dados, eliminando o risco de intervenção manual e garantindo que o sistema reflete corretamente as atualizações recebidas do EWP. A Figura 4.18 ilustra um excerto da Classe desta operação.

UlMobInUpdateRejectedNominationStatusOperation .java

```
(...)
@WorkflowComponent(type = "ULISBOA_STUDENT_MOBILITY")
public class UlMobInUpdateRejectedNominationStatusOperation extends AbstractWorkflowOperation {
    private static final long serialVersionUID = 1L;
    @Override
    public void execute() {
        ServiceProvider.getService(TransactionalManager.class).executeInWriteContext(getClass().getName(), () -> {
            final UlMobInStudentNomination incomingStudentNomination =
                    getState().getWorkflowInstance().getUlMobInStudentNomination();
            //Set the Boolean previouslyRejected attribute to true;
            incomingStudentNomination.setPreviouslyRejected(true);
            //Atualiza o nomination Status para PENDING
            incomingStudentNomination.setNominationStatus(UlMobInNominationStatus.PENDING);
            //POST CNR a comunicar alteração do Status
            InstanceCommunicationUtils.sendIncomingMobilityChangeNotificationToEWPNode(incomingStudentNomination);
            //Transita de Estado
            getOperation().applyDefaultTransition(getState(), getResponsible());
        });
    }
(...)
```

Figura 4.18: Excerto da classe *UlMobInUpdateRejectedNominationStatusOperation.java* relativa à operação de *Atualizar Nomeação Rejeitada*.

Estas operações são cruciais para o funcionamento do *workflow* e para a integração entre o sistema Fenix e o nó EWP. Ao conectar diretamente os processos internos com a rede EWP, elas não só garantem a transição fluída entre estados no *workflow*, como também habilitam uma comunicação das informações consistente com as IOs.

4.1.6 Geração de Documentos

No âmbito do processo de mobilidade, identificou-se a necessidade de gerar automaticamente determinados documentos, em momentos específicos, com base na informação já existente no domínio do Fénix. Entre estes documentos incluem-se a *Carta de Aceitação*, a *Declaração de Chegada* e a *Declaração de Estada*.

Para satisfazer esta necessidade, utilizámos a funcionalidade de Documentos *Online* disponibilizada pelo Fénix, conforme descrito no ponto 2.9. Nesta fase, desenvolvemos a Carta de

Aceitação, cujo *template* flexível se encontra parcialmente exemplificado na figura 4.19 (documento completo no Anexo F.1), com base num modelo atualmente utilizado pela FL. Este documento é totalmente configurável, permitindo que as diferentes Escolas adaptem o conteúdo às suas necessidades. Além disso, para suportar este e outros documentos planeados, como as Declarações de Chegada e Estada, criámos fontes de dados específicas, acessíveis às Escolas para uso nos seus próprios *templates*.

A figura 4.19 ilustra esta flexibilidade, mostrando ao centro a interface para configuração do conteúdo do documento e, à direita, algumas das fontes de dados criadas e disponibilizadas para utilização nos *templates*.

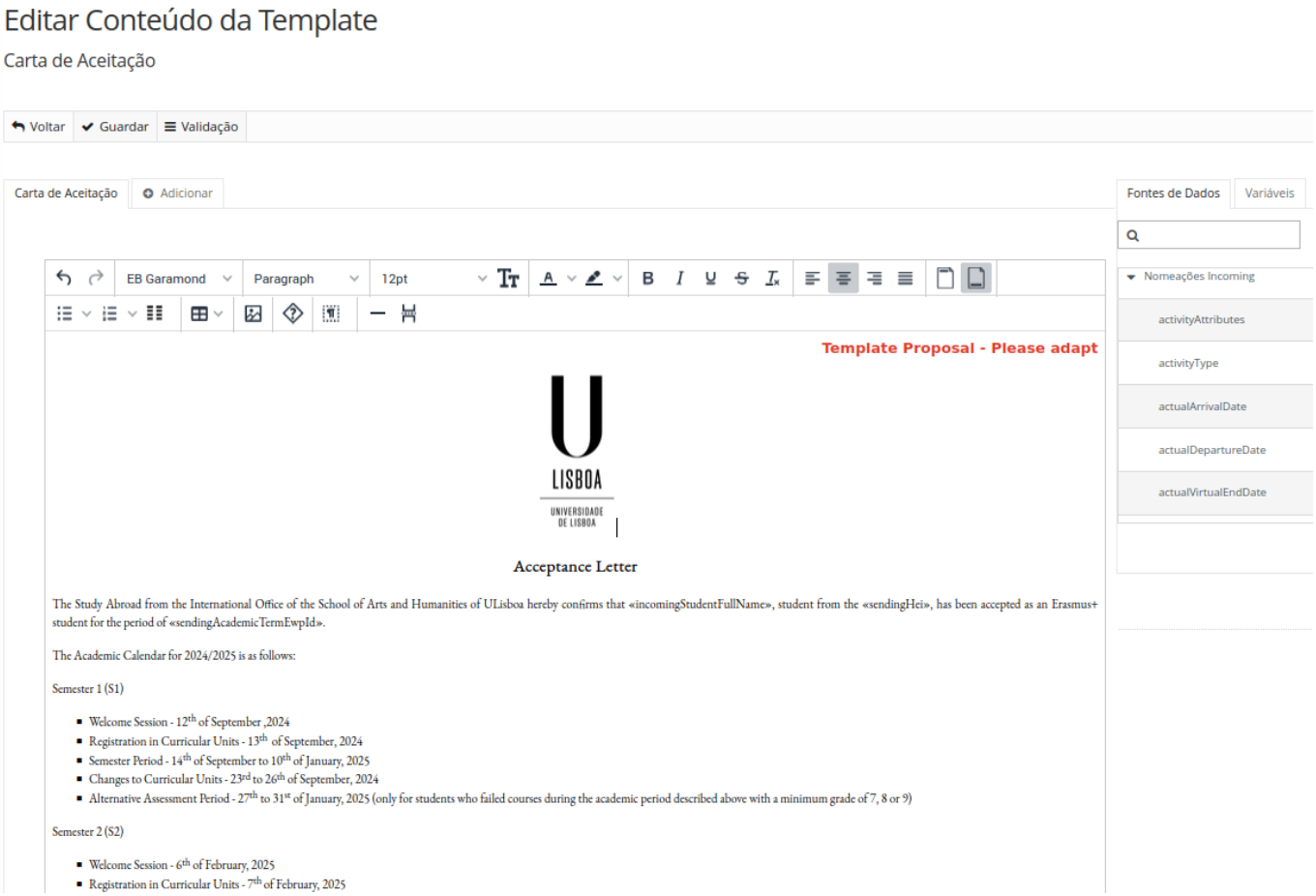


Figura 4.19: Menu Fenix *Editar Conteúdo da Template*: Carta de Aceitação.

Estas fontes de dados consistem em classes definidas, com métodos que garantem acesso direto e estruturado às informações necessárias para o preenchimento dinâmico dos documentos. Entre os métodos desenvolvidos, destacam-se, por exemplo:

- `getIncomingStudentFullName()`: para obter o nome completo do aluno *incoming*;
- `getIncomingStudentGlobalID()`: para obter o ID do aluno *incoming*;
- `getReceivingHeiComment()`: para extrair o comentário da ID;

- `getPlannedArrivalDate()`: para aceder à data de chegada planeada;
- `getActualArrivalDate()`: para obter a data efetiva de chegada.

Apesar de, nesta fase do projeto, apenas a *Carta de Aceitação* ter sido desenvolvida, as fontes de dados criadas permitem que as Escolas desenvolvam, de forma autónoma, *templates* para outros documentos necessários, eliminando a intervenção manual e assegurando a conformidade dos dados com o domínio.

Este desenvolvimento não se limita a resolver a necessidade de um único documento, uma vez que, estabelece uma base versátil para a geração de documentos no âmbito das nomeações *Incoming*. No futuro, estas fontes de dados poderão ser adaptadas para suportar também documentos relacionados com nomeações *Outgoing*, dado que os métodos implementados se baseiam em atributos comuns às instâncias do domínio de ambas as mobilidades.

4.1.7 Perfis de Acesso

O Fenix disponibiliza a criação de perfis de acesso que agrupam determinados utilizadores de acordo com os critérios pretendidos e que permitem diferentes acessos a diferentes menus e submenus. Para além disso, os *workflows* têm também perfis de acesso, para os intervenientes nos processos. Exemplos simples incluem o perfil de Reitor, que apenas inclui o Reitor, e o perfil de Vice-Reitor, que deverá conter como membros todos os Vice-Reitores. Nos *workflows*, cada perfil de acesso está associado a um tipo de processo específico; no caso deste projeto, o processo é de mobilidade pelo que será necessário ter perfis de acesso para os intervenientes, ou seja, os responsáveis do lado a ID – habitualmente o Núcleo de Mobilidade - e os próprios Alunos *Incoming*.

Assim e após os desenvolvimentos realizados até este ponto, foi necessário definir e implementar os perfis de acesso com o objetivo de adaptar as permissões de cada utilizador às suas responsabilidades no módulo e nos processos de mobilidade *Incoming*.

Começando pelos menus, e seguindo estrutura utilizada anteriormente neste âmbito, foram criados os 3 perfis de controlo de acesso aos menus e ilustrados na figura 4.20, por forma a refletir 3 níveis de acesso.
O `ULISBOA_MOBILITY_INCOMING_NOMINATION_STUDENT_PROFILE` é atribuído ao aluno incoming e deverá ter acesso às entradas de menu do aluno e da nomeação.
O `ULISBOA_MOBILITY_INCOMING_NOMINATION_BACKOFFICE_PROFILE`, e `ULISBOA_MOBILITY_INCOMING_NOMINATION_ADMIN_PROFILE` que atende às necessidades dos membros de *backoffice*, podendo dividir em 2 níveis configuráveis os acessos aos diferentes menus, de acordo com a necessidade e organização interna de cada Escola, como por exemplo, apenas o perfil de ADIMN poder apagar nomeações. Ao contrário do aluno, os membros deste perfil deverão ter acesso aos submenus *Tipos de Nomeação* e *Execuções de Nomeações* do menu *Incoming* do Módulo de Mobilidade[1].

[1] Menus e submenus ilustrados anteriormente na figura 4.16.

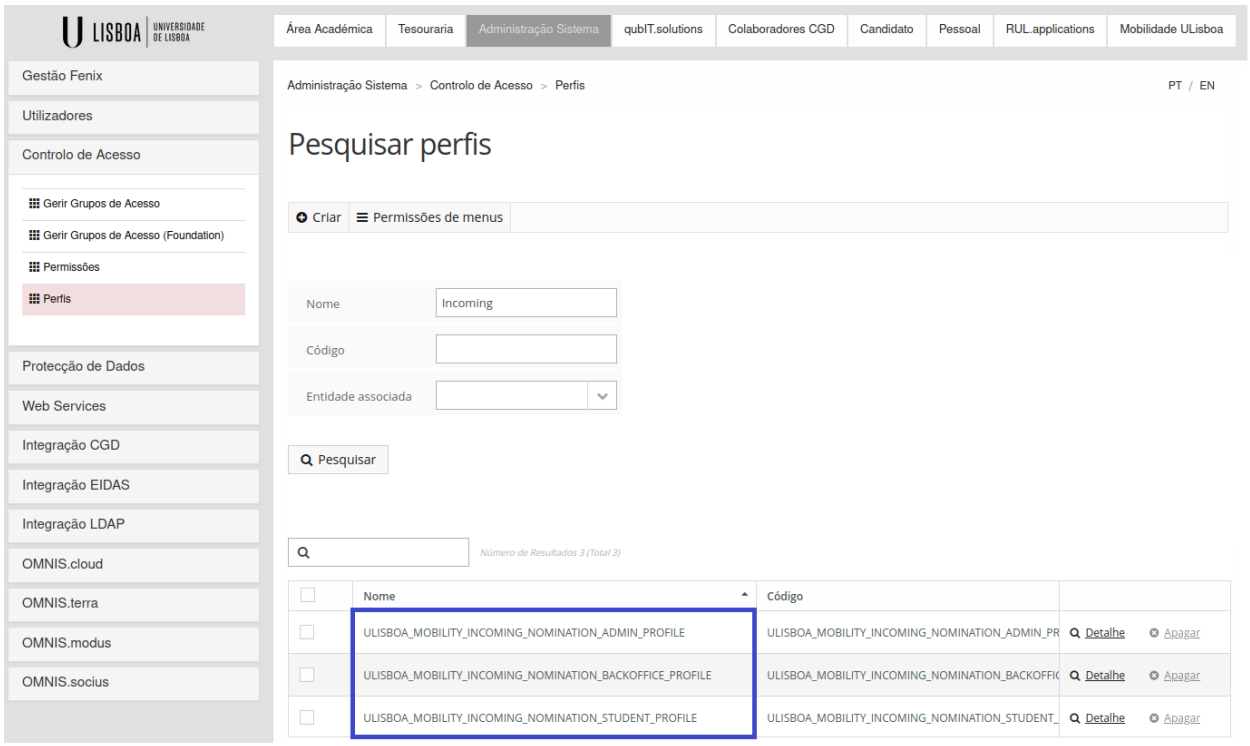


Figura 4.20: Controlo de acessos.

De seguida, para além dos perfis onde se definem os acessos aos menus, foi necessário ter perfis de acesso associados ao *workflow*, no caso, os perfis dos intervenientes identificados anteriormente em 3.2.1 e ilustrados na figura 4.21.

O NM já possui habitualmente um perfil pré-definido no sistema, o que dispensou nesta fase a criação de um perfil adicional. Este perfil preexistente permite ao NM o acompanhamento e gestão centralizada do módulo, e deverá proporcionar acesso a todas as instâncias do processo de mobilidade incoming, sendo direcionado a utilizadores que precisam de consultar e colaborar em múltiplos processos sem ter as permissões administrativas completas, os quais são manualmente adicionados pelas próprias equipas da Escola.

Por outro lado, para o Aluno *Incoming*, foi necessário proceder à criação de um perfil. Este perfil representa um caso particular, uma vez que, ao contrário dos perfis administrativos, cada aluno apenas deverá ter acesso às suas nomeações e informações. A este perfil, são adicionados de forma automática as conta do aluno geradas automaticamente após a aprovação do seu processo de nomeação, não existindo necessidade de intervenção manual por parte dos serviços administrativos. Para isso, e por forma a restringir o acesso exclusivamente ao seu próprio processo de mobilidade, foi implementada uma estratégia de validação que assegura que o aluno apenas visualiza informações relativas ao seu processo, promovendo a segurança e privacidade dos dados. A figura 4.22 ilustra a estrutura desta validação para o perfil de aluno.

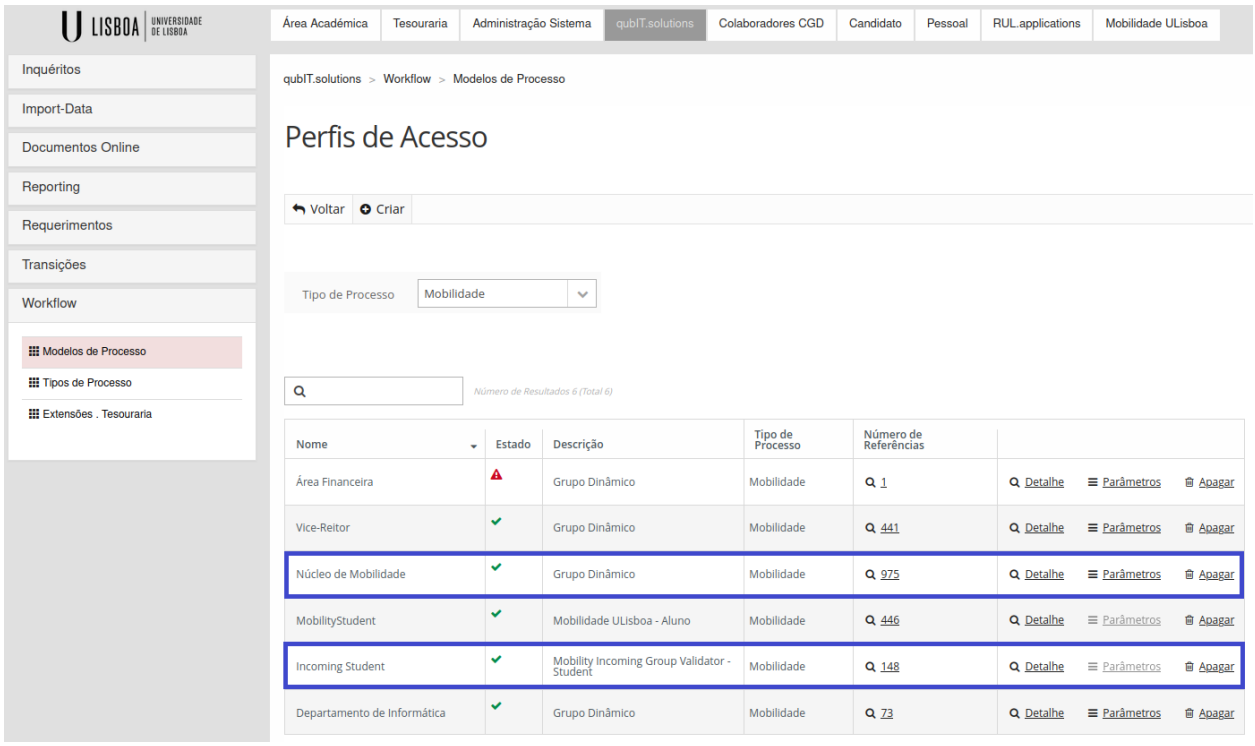


Figura 4.21: Perfis de Acesso: Mobilidade.

MobilityIncomingGroupValidator.java

```java
@WorkflowComponent(type = "ULISBOA_STUDENT_MOBILITY")
public class MobilityIncomingGroupValidator extends FenixWorkflowAccessControlGroupValidator {

    @Override
    protected Collection<Person> getMembers(final WorkflowInstance instance, final IHasWorkflowParameters ihwp) {
        final Person person = instance.getUlMobInStudentNomination().getIncomingStudent().getPerson();
        return person != null ? Collections.singleton(person) : Collections.emptySet();
    }

    @Override
    protected boolean isMember(final WorkflowInstance instance, final Person person, final IHasWorkflowParameters ihwp) {
        return Objects.equals(instance.getUlMobInStudentNomination().getIncomingStudent().getPerson(), person);
    }

    public static String getPresentationName() {
        return I18N.i18n(ArtifactHandler.BUNDLE, "Mobility Incoming Group Validator - Student");
    }
}
```

Figura 4.22: Excerto do validador desenvolbido para o Aluno *Incoming*.

Por último, para coordenar o acesso e assegurar as permissões adequadas para cada perfil, foram adicionadas permissões específicas à classe de permissões da mobilidade já existente. Esta classe foi expandida para incluir as permissões necessárias ao acesso das entradas de menu e execução de ações no *workflow*, de acordo com as funções de cada utilizador no sistema Fenix.

É esta classe que permitirá posteriormente estabelecer a associação entre os utilizadores e os seus respetivos perfis e permissões, facilitando a atribuição correta de acessos no *frontend*.

4.2 Avaliação

Nesta secção, é detalhada a metodologia adotada para a avaliação dos desenvolvimentos realizados no âmbito deste projeto, abordando as limitações existentes devido à natureza simulada de alguns testes. Em seguida, são analisados os resultados obtidos, com foco na validação das comunicações implementadas e dos fluxos de trabalho introduzidos no sistema.

4.2.1 Método de Avaliação

A avaliação dos desenvolvimentos deste projeto foi realizada essencialmente através de simulações e testes internos, aliada à verificação de *logs* de execução. Embora as comunicações reais com a rede EWP ainda não estejam operacionais, o objetivo principal dos testes foi validar o correto funcionamento da instância da Escola, utilizando o nó EWP para simular as operações esperadas no futuro ambiente de produção.

Para contornar as limitações do estado atual, foi implementada uma simulação de chamadas ao nó EWP, utilizando o separador "*Importar Nomeações Incoming EWP*" referido no ponto 4.1.3 e visível na figura 4.4. Esta funcionalidade devolve dados configurados manualmente, que simulam as nomeações recebidas via EWP. Este mecanismo permitiu testar a instanciação automática de nomeações *incoming*, o funcionamento do *workflow*, as transições de estado e a execução das operações desenvolvidas, assegurando o correto evoluir dos processos até ao estado de "Formalização de Matrícula através dos processos de inscrição".

O ambiente de testes foi configurado e disponibilizado pelo NDS, utilizando um *dump* da base de dados local dos desenvolvimentos realizados, ligado a um *EWP Node* de teste. Esta configuração permitiu simular a comunicação possível com o nó EWP, replicando parcialmente o comportamento esperado no futuro ambiente de produção. No entanto, trata-se de uma abordagem temporária em que a instância da Escola comunica diretamente com o nó EWP, assumindo o papel da instância da Reitoria. Esta escolha permite, nesta fase do projeto, desenvolver e validar as operações e transições de estado no *workflow*, uma vez que as comunicações entre as instâncias e a Reitoria apenas serão desenvolvidas numa fase posterior, conforme definido na arquitetura descrita em 3.3.

Durante os testes, foi possível verificar diretamente na interface que as operações de Aprovação, Rejeição, Solicitação de Documentos, entre outras, levam ao estado correspondente no *workflow*. Contudo, as comunicações realizadas com o nó EWP durante a execução dessas operações não são visíveis na interface. Para validar estas comunicações, foi realizada uma análise detalhada dos *logs* do *EWP Node Admin*, que confirmaram que, ao despoletar operações no *workflow*, as chamadas *GET* ou *POST* são realizadas corretamente.

Adicionalmente, este ambiente de testes permitiu à equipa do Núcleo de Gestão de Serviços Académicos (NGSA) realizar testes manuais do processo, contribuindo para identificar ajustes

necessários e garantir o alinhamento com os requisitos iniciais. A análise dos *logs* confirmou ainda a eficácia das transações realizadas entre o sistema e o nó EWP, nomeadamente nas operações de importação de nomeações, aprovação, rejeição e atualização de nomeações, apesar de, como mencionado, não estarmos ainda perante a configuração final da arquitetura prevista.

4.2.2 Avaliação da Usabilidade

A avaliação da usabilidade foi condicionada pela impossibilidade de realizar testes formais junto dos utilizadores finais, como a aplicação de questionários *System Usability Scale* (SUS), devido ao estado de desenvolvimento e às limitações do ambiente simulado para este projeto, bem como à limitada disponibilidade da equipa do NGSA. No entanto, no contexto dos testes internos, foi possível observar melhorias nos fluxos de trabalho e na simplicidade do processo, na medida em que a informação é recebida de forma automática e agregada num só espaço.

A interface foi desenhada de acordo com o padrão existente nas instâncias Fenix, facilitando a navegação intuitiva entre menus e submenus para os utilizadores já familiarizados com o sistema, prevendo uma fácil adopção. Para o perfil de alunos, o sistema restringe o acesso aos seus próprios processos, respeitando a privacidade e promovendo uma organização mais clara da informação visível. Já para os perfis de mobilidade e *backoffice*, a interface permite o acesso a todos os processos de nomeação, reunindo e concentrando todas as nomeações num espaço dedicado, o que simplifica a gestão das mesmas.

Além disso, a criação automática de nomeações elimina a necessidade de ações manuais, permitindo aos utilizadores das equipas de mobilidade das Escolas, visualizar todas as nomeações e aceder facilmente às operações disponíveis - como aprovar, rejeitar ou solicitar documentação adicional - em função do estado de cada processo. Esta abordagem facilita a interação com o sistema e assegura que as equipas de mobilidade poderão desempenhar as suas funções de forma mais eficiente, com menor margem para erro, visto que a informação é gerida automaticamente no domínio Fenix a partir dos dados recebidos.

4.2.3 Análise de Resultados

Os resultados da avaliação dos desenvolvimentos realizados basearam-se na análise de *logs* de comunicação gerados durante os testes descritos em 4.2.1. Estes testes permitiram validar a integração do sistema Fenix com o nó EWP, confirmando a automação dos processos de nomeação *incoming*.

Foi a análise destes logs de comunicações, atualmente possíveis, que confirmou o correto funcionamento das principais funcionalidades de comunicação entre o sistema Fenix e o nó EWP, cumprindo o principal objetivo inicial do projeto: a automação das nomeações *incoming* com o EWP.

A análise centrou-se nos fluxos de comunicação previamente descritos em 3.4.2, avaliando o comportamento do sistema nos seguintes pontos-chave: - Receção de nomeações simuladas via chamada *GET*, representando a operação de importação. - Instanciação automática de nomeações

no domínio Fenix e transição correta no *workflow*. - Envio de respostas *POST*, incluindo confirmações de receção, alterações de estado (aceitação/rejeição) e atualizações.

As figuras 4.23 a 4.29 ilustram exemplos de *logs* obtidos durante os testes, que confirmam o correto funcionamento das operações simuladas.

A figura 4.23 contém os *Communications Logs* após importar nomeações do nó EWP e demonstra, entre outras coisas, que os *GET* do *omobilities* (ID 4334 e 4330) e o *POST* do *CNR* do *imobilities* (ID 4338) - despoletados pela operação do separador "importar" que simula o recebimento de um *POST* de um *CNR* da IO e uma chamada *GET* em resposta (ilustrado na figura 3.2) - estão a ser bem sucedidos.

EWP Node Communication Logs admin

Search keyword

ID	Type	Source	Target	Start Processing Time	End Processing Time	Status
4338	HOST_IN	client-1	ulisboa.pt:imobility-cnr	Oct 30, 2024, 11:39:30 AM	Oct 30, 2024, 11:39:30 AM	SUCCESS
4334	HOST_IN	client-1	ulisboa.pt:omobilities[2]:get	Oct 30, 2024, 11:39:30 AM	Oct 30, 2024, 11:39:30 AM	SUCCESS
4330	HOST_IN	client-1	ulisboa.pt:omobilities[2]:get	Oct 30, 2024, 11:39:30 AM	Oct 30, 2024, 11:39:30 AM	SUCCESS
4329	HOST_IN	client-1	ulisboa.pt:omobilities[2]:specification	Oct 30, 2024, 11:39:30 AM	Oct 30, 2024, 11:39:30 AM	SUCCESS
4328	HOST_IN	client-1	general:versions-supported	Oct 30, 2024, 11:39:30 AM	Oct 30, 2024, 11:39:30 AM	SUCCESS
4324	HOST_IN	client-1	ulisboa.pt:omobilities[2]:index	Oct 30, 2024, 11:39:29 AM	Oct 30, 2024, 11:39:30 AM	SUCCESS
4323	HOST_IN	client-1	general:versions-supported	Oct 30, 2024, 11:39:29 AM	Oct 30, 2024, 11:39:29 AM	SUCCESS
55	HOST_IN	client-1	ulisboa.pt:imobility-cnr	Oct 29, 2024, 1:52:55 PM	Oct 29, 2024, 1:52:55 PM	SUCCESS
48	HOST_IN	client-1	ulisboa.pt:imobility-cnr	Oct 29, 2024, 1:51:45 PM	Oct 29, 2024, 1:51:45 PM	SUCCESS
44	HOST_IN	client-1	ulisboa.pt:omobilities[2]:get	Oct 29, 2024, 1:51:44 PM	Oct 29, 2024, 1:51:45 PM	SUCCESS

Showing 1 to 10 of 15 entries « ‹ 1 2 › »

Figura 4.23: *Communication Logs* após importar nomeações do nó EWP.

Por sua vez, a figura 4.24, ilustra o detalhe da comunicação com o ID 4334 (*GET*) corretamente importada e, a figura seguinte 4.25, o conteúdo dessa nomeação, com o formato definido pelo EWP.

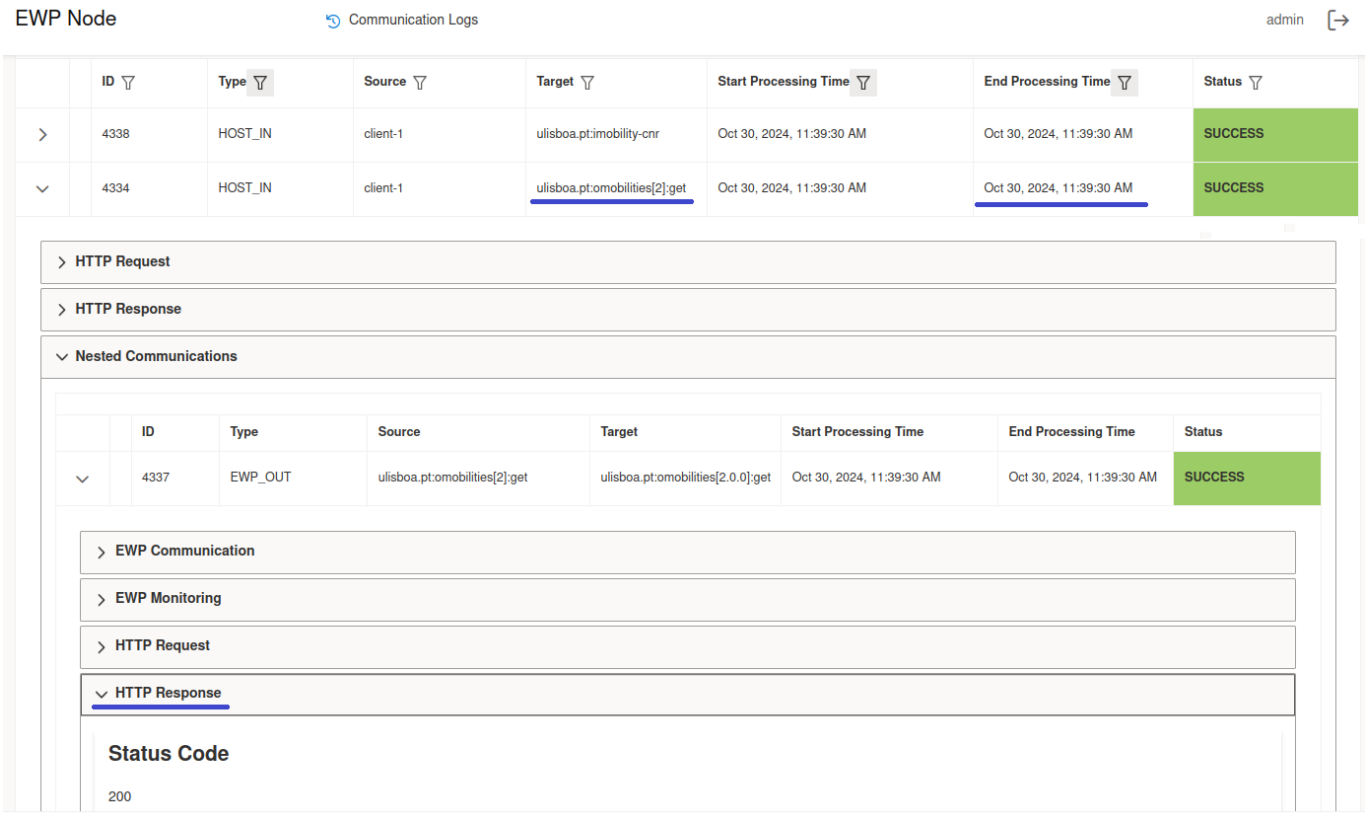


Figura 4.24: *Communication Logs*: Exemplo de uma nomeação corretamente importada - ID 4334.

```xml
<omobilities-get-response xmlns="https://github.com/erasmus-without-paper/ewp-specs-api-omobilities
   <student-mobility>                                    /blob/stable-v2/endpoints/get-response.xsd">
      <omobility-id>test2</omobility-id>
      <sending-hei>
         <hei-id>ulisboa.pt</hei-id>
         <ounit-id>ULREIT</ounit-id>
         <iia-id>iiaIdSendingTest</iia-id>
      </sending-hei>
      <receiving-hei>
         <hei-id>ulisboa.pt</hei-id>
         <ounit-id>ULREIT</ounit-id>
         <iia-id>iiaIdSendingTest2</iia-id>
      </receiving-hei>
      <sending-academic-term-ewp-id>2020/2021-1/1</sending-academic-term-ewp-id>
      <receiving-academic-year-id>2020/2021</receiving-academic-year-id>
      <student>
         <given-names xml:lang="PT">João</given-names>
         <given-names xml:lang="EN">John</given-names>
         <family-name xml:lang="PT">Ninguém</family-name>
         <family-name xml:lang="EN">Doe</family-name>
         <global-id>globalIDtest</global-id>
         <birth-date>1991-02-10Z</birth-date>
         <citizenship>PT</citizenship>
         <gender>1</gender>
         <email>johndoe@ulisboa.pt</email>
      </student>
      <status>nomination</status>
      <activity-type>student-studies</activity-type>
      <activity-attributes>long-term</activity-attributes>
      <eqf-level-studied-at-nomination>2</eqf-level-studied-at-nomination>
      <eqf-level-studied-at-departure>2</eqf-level-studied-at-departure>
      <nominee-isced-f-code>0001</nominee-isced-f-code>
   </student-mobility>
</omobilities-get-response>
```

Copy to clipboard

Validation

✓ Valid

Figura 4.25: *Communication Logs*: *BODY* da nomeação corretamente importada - ID 4334.

Nas figuras 4.26 e 4.27, é possível observar o detalhe da comunicação com o ID 4338, que realiza com sucesso um *POST* de um *CNR* que confirma à IO a receção das nomeações por si enviadas e com os *omobility_id test1* e *test2*.

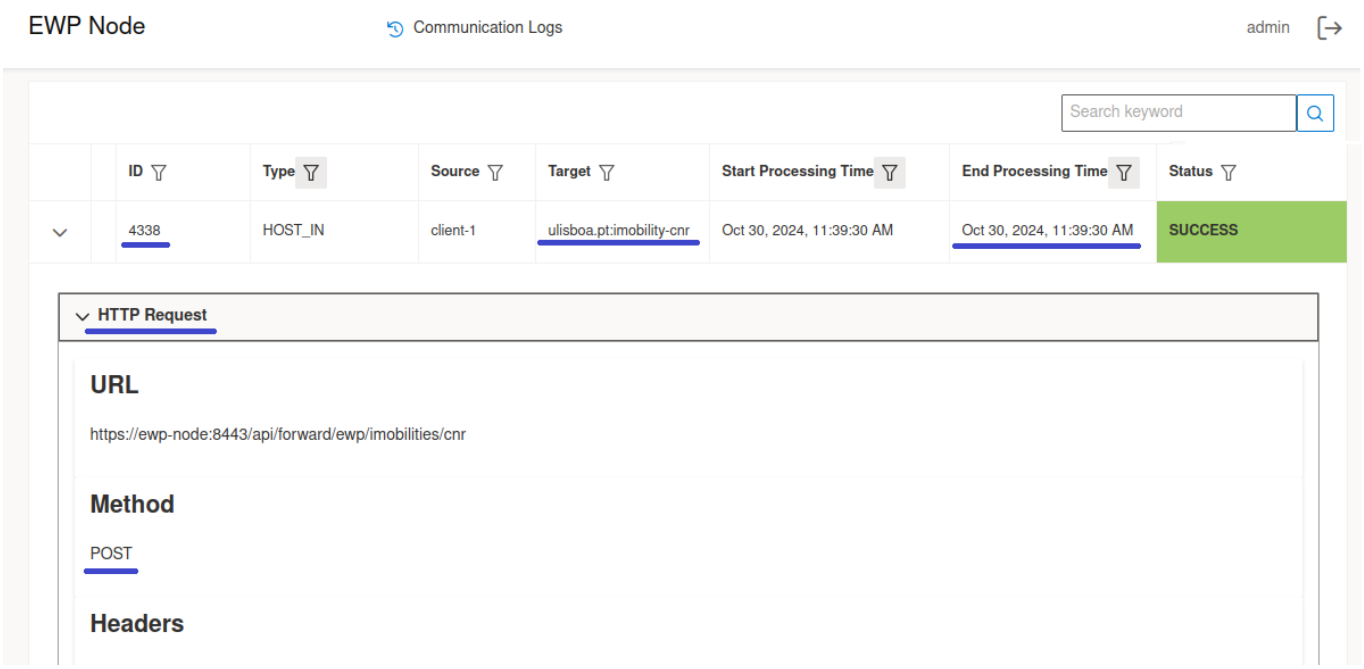


Figura 4.26: *Communication Logs*: *POST* de CNR - ID 4338.

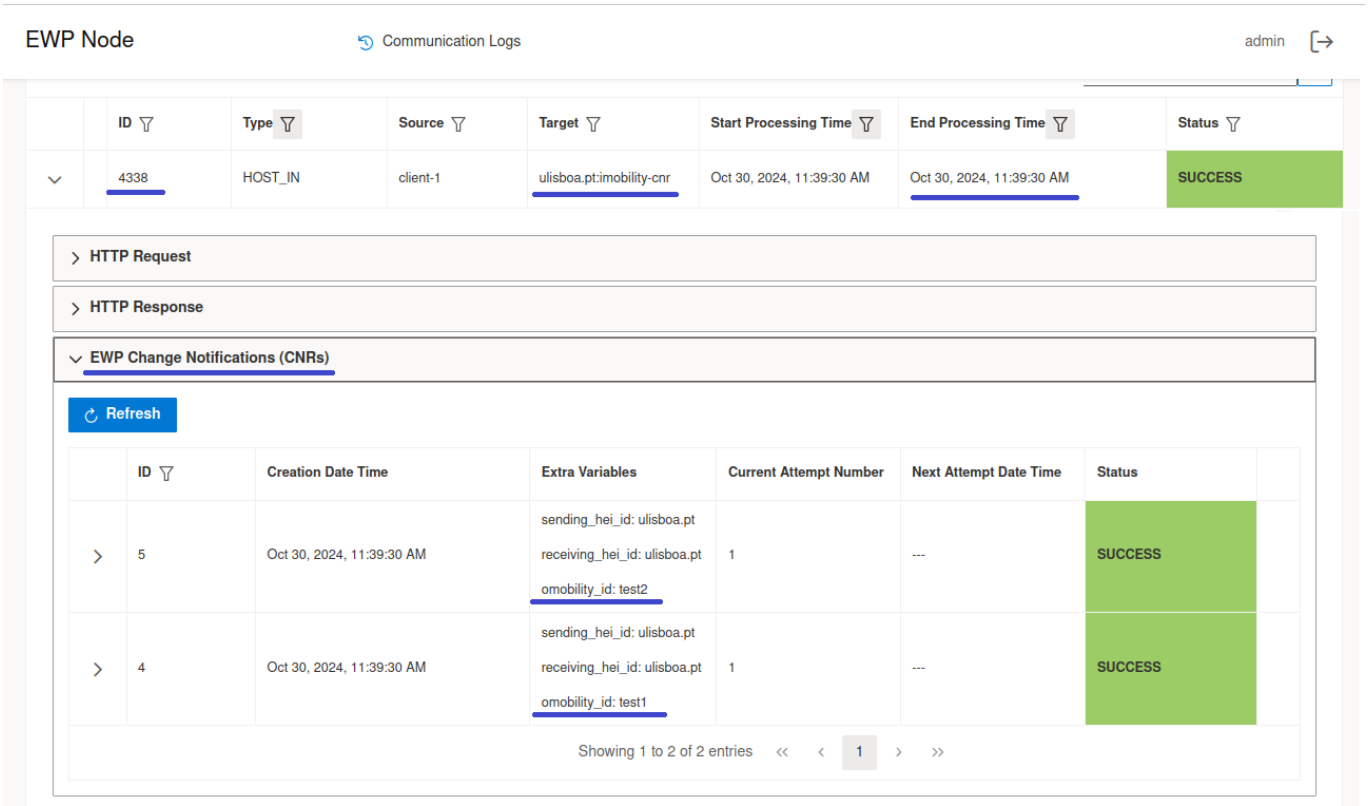


Figura 4.27: *Communication Logs*: CNR de confirmação de nomeações *test1* e *test2* corretamente importadas - ID 4338.

Por último, as figuras 4.28 e 4.29 representam comunicações despoletadas automaticamente pelo sistema na sequência da execução das operações de Aceitação e Rejeição, respetivamente. Estas comunicações, não visíveis na interface pelo utilizador, confirmam a realização de chamadas *POST CNR* para o nó EWP, como definido nos requisitos de comunicação do EWP.

A análise dos *logs* confirma assim que o sistema Fenix responde adequadamente às operações de receção, aceitação, rejeição e atualização de nomeações, refletindo corretamente as informações do EWP no domínio Fenix. Além disso, demonstram que, para além de transitarem entre estados no *workflow* - comportamento visível ao utilizador na interface - as operações de Aceitação e Rejeição também despoletam as comunicações necessárias com o nó EWP, conforme definido nos requisitos.

Além desta análise, constatou-se que a participação da equipa do NGSA, com *feedback* sobre as funcionalidades e requisitos do processo de negócio, contribuiu para melhorias incrementais. Esta abordagem permitiu ajustes pontuais e uma validação mais precisa dos resultados, alinhando o desenvolvimento com os objetivos iniciais.

Embora não tenha sido possível obter uma avaliação formal dos utilizadores finais devido às limitações mencionadas, conclui-se que o *Minimum Viable Product* (MVP) foi corretamente desenvolvido, e os resultados alcançados vão ao encontro dos objetivos delineados. Prevê-se, assim, uma melhoria significativa nos processos de mobilidade *incoming*, à medida que o restante projeto seja desenvolvido e implementado.

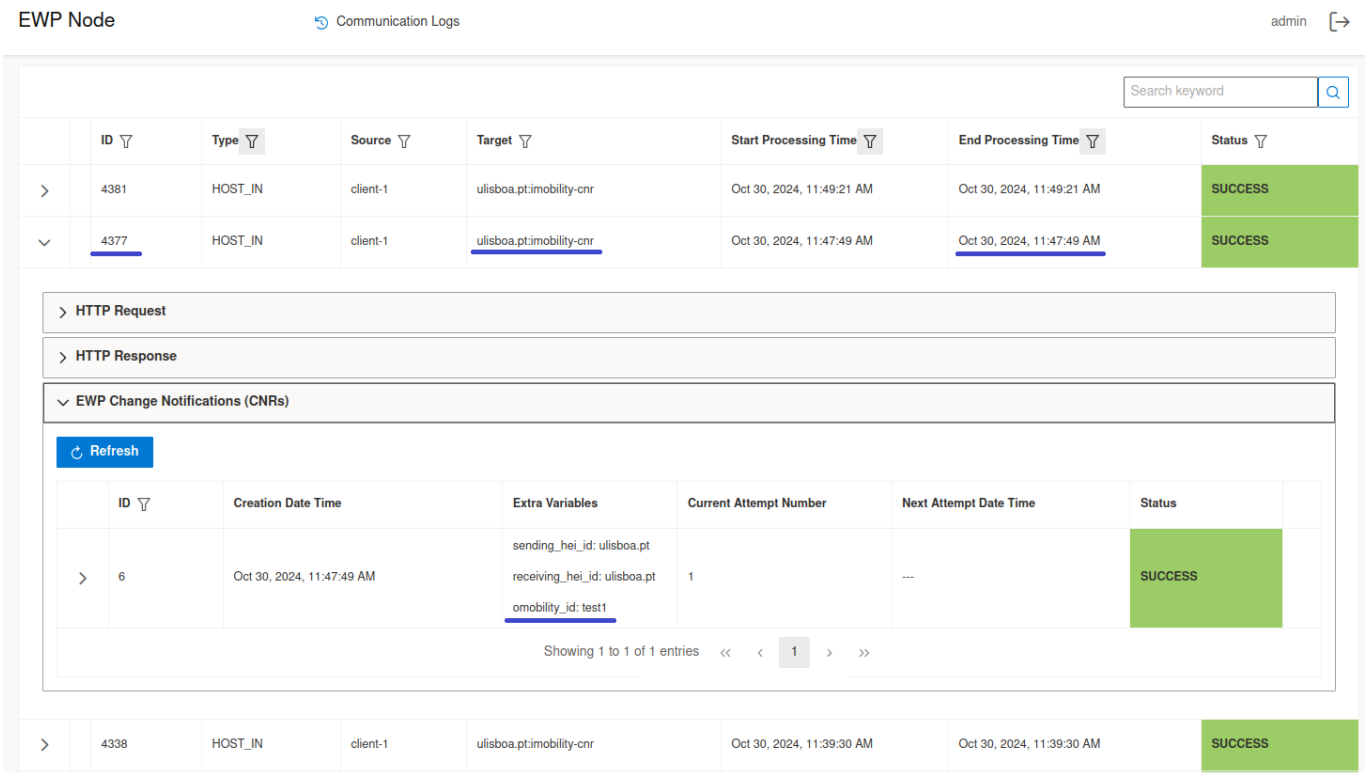


Figura 4.28: *Communication Logs*: *POST* de CNR relativo à Aceitação da nomeação *test1* - ID 4377.

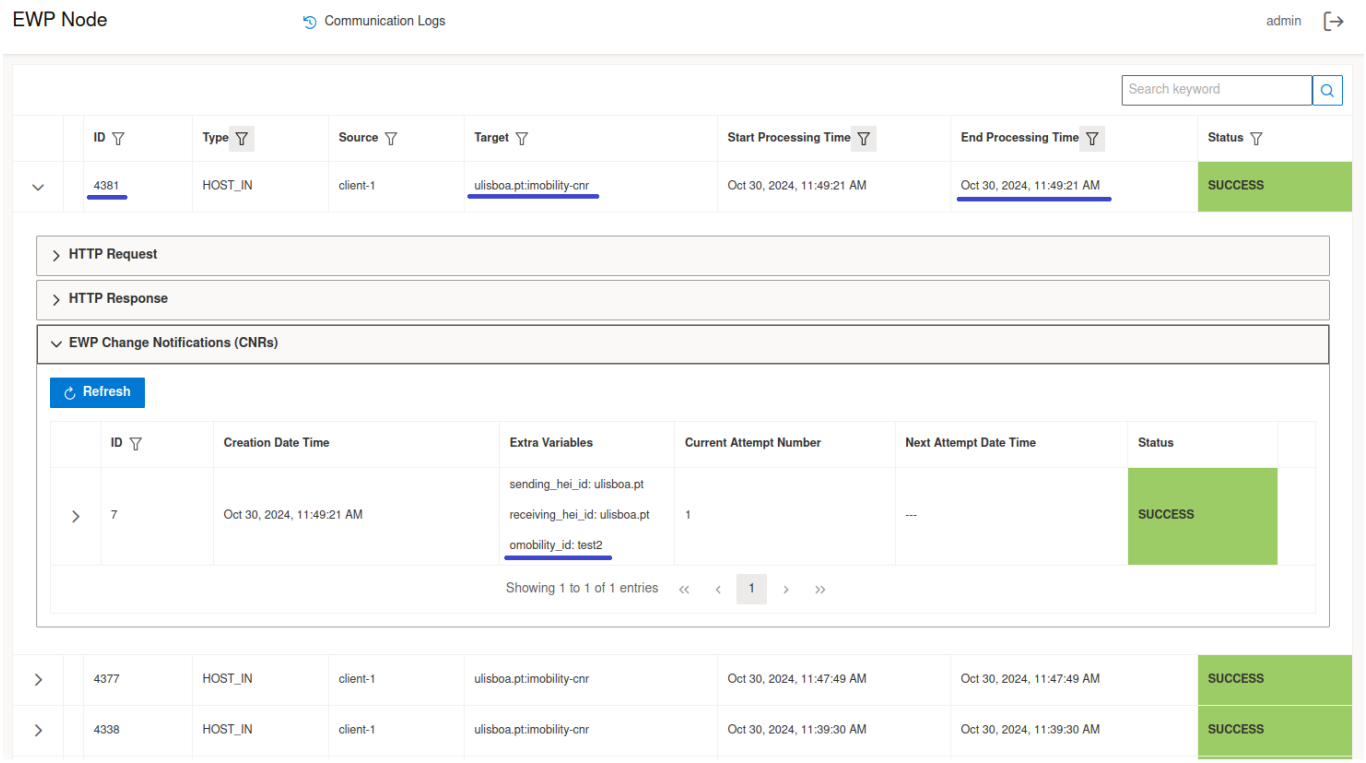


Figura 4.29: *Communication Logs*: *POST* de CNR relativo à Rejeição da nomeação *test2* - ID 4381.

4.3 Sumário

O quarto capítulo foca-se na implementação prática do projeto, detalhando os principais componentes desenvolvidos, incluindo o ambiente de desenvolvimento, a interface, os perfis de acesso, e a integração da geração de documentos no sistema. Este capítulo descreve ainda o processo de modelação do *workflow* de mobilidade *incoming*, desde a sua transposição do modelo BPMN até à implementação dos estados e operações.

Na segunda parte do capítulo, abordámos o método de avaliação, destacando as limitações encontradas e as soluções adotadas para contorná-las, bem como a análise detalhada dos resultados obtidos. O próximo capítulo apresentará as conclusões do projeto, sintetizando os principais contributos alcançados e propondo melhorias para trabalhos futuros.

Capítulo 5

Conclusão e Trabalho Futuro

Este projeto de mobilidade revelou-se um desafio multifacetado, exigindo um esforço significativo tanto em termos técnicos como organizacionais. A necessidade de articular soluções transversais e flexíveis para as 18 Escolas da ULisboa, cada uma com as suas especificidades, foi acompanhada pela complexidade de integrar um projeto europeu em constante evolução, como o EWP. Estas circunstâncias exigiram uma abordagem colaborativa e iterativa, garantindo a flexibilidade necessária para adaptar o sistema às diferentes necessidades e alterações impostas pelo EWP.

O levantamento de requisitos foi uma das etapas mais desafiantes. A ausência inicial de documentação consolidada por parte do EWP obrigou à análise direta de repositórios e ficheiros `.xsd`, o que implicou um grande investimento de tempo. Este desafio foi agravado pelo facto de o primeiro documento que resume os requisitos obrigatórios do EWP ter sido disponibilizado apenas em maio de 2024 [9]. Além disso, as alterações nas versões das especificações exigiram revisões contínuas dos requisitos e ajustamentos ao modelo de domínio.

Outro desafio significativo foi o desenvolvimento de uma solução que respeitasse a complexidade e a diversidade das Escolas, permitindo flexibilidade na adaptação dos fluxos de trabalho sem comprometer a consistência e os requisitos do EWP. Também foi essencial criar operações específicas no *workflow* que, além de possibilitarem a transição entre estados, integrassem as comunicações com o nó EWP. Estas operações — aceitação, rejeição e atualização de nomeações — asseguram que o sistema reflete corretamente as alterações realizadas e garante a integridade dos dados no domínio Fenix. Por fim, o cruzamento do domínio existente no sistema Fenix com as novas entidades e associações específicas do projeto *incoming* envolveu múltiplas iterações e ajustes, minimizando redundâncias e assegurando a integração e coerência dos dados.

Apesar destas dificuldades, o projeto alcançou resultados concretos. Foi possível implementar o MVP, que contempla a receção de nomeações *incoming* via EWP, permitindo a sua gestão (aceitação, rejeição ou atualização) e a criação automática de documentos, como a carta de aceitação. Adicionalmente, as fontes de dados desenvolvidas fornecem uma base sólida para a geração de futuros documentos, tanto no âmbito das nomeações *incoming*, como *outgoing*. O projeto incluiu também a modelação completa do processo de mobilidade *incoming* na forma de *workflow*, que, embora ainda tenha estados e operações por desenvolver, oferece uma visão clara do fluxo do processo e da direção a seguir. Estes desenvolvimentos estabelecem uma base ro-

busta para a continuidade dos trabalhos, promovendo a desmaterialização e a automatização de processos de mobilidade.

Com base nos progressos realizados, o trabalho futuro deverá concentrar-se na conclusão do *workflow* das nomeações *incoming*, desenvolvendo os estados e operações ainda necessários, como a gestão dos LA e dos ToR, para que deixem de utilizar documentos carregados e sejam integrados com os desenvolvimentos do EWP, caso as especificações estejam disponíveis. Será também essencial implementar a comunicação interna entre as instâncias da Reitoria e das Escolas, assegurando a sincronização total de dados de acordo com a arquitetura definida. Paralelamente, a extensão do projeto para a mobilidade *outgoing* poderá beneficiar diretamente das bases estabelecidas, promovendo a consistência entre os diferentes fluxos de mobilidade. Finalmente, será crucial acompanhar as evoluções do EWP, adaptando o sistema às novas especificações e requisitos que possam surgir.

Este projeto representa um marco na automação e integração dos processos de mobilidade na ULisboa, estabelecendo as fundações para melhorias contínuas. À medida que os próximos passos forem dados, espera-se que o sistema se consolide como uma ferramenta indispensável para a gestão de mobilidade, assegurando a adaptabilidade em resposta aos desafios futuros deste contexto.

Abreviaturas

API *Application Programming Interface*. 3, 25, 33, 34

BPMN *Business Process Model and Notation*. 5, 25, 29, 30, 32, 33, 36, 47, 67, 81, 84, 90

CEES Carta Erasmus para o Ensino Superior. 24

CIE Coordenadora Institucional de Erasmus. 90, 91

CNR *Change Notification Receiver*. xvi, 33, 65–67

CRUD *Create, Read, Update, Delete*. 12

DEA Diagrama de Entidades e Associações. 5, 20, 29, 30, 33–36, 38, 41

DI Departamento de Informática. 3

DML *Domain Modeling Language*. 9, 11–14, 17

DREI Divisão de Relações Externas e Internacionais. 22, 87, 88

DSL *Domain Specific Language*. 11–15, 17, 38

DTO *Data Transfer Object*. 38

ECTS Sistema Europeu de Transferência e Acumulação de Créditos. 7

EWP *Erasmus Without Paper*. vi, viii, xii, xvi, 2, 3, 5, 20–26, 29–33, 35–38, 41, 44, 47, 50, 51, 53–55, 60–63, 66, 69, 70

FA Faculdade de Arquitetura. 8

FBA Faculdade de Belas-Artes. 8

FC Faculdade de Ciências. xii, xvi, 3, 8, 25, 81, 83

FD Faculdade de Direito. 8

FF Faculdade de Farmácia. 8

FL Faculdade de Letras. xii, xvi, 3, 8, 25, 56, 87, 89

FM Faculdade de Medicina. 8

FMD Faculdade de Medicina Dentária. 8

FMH Faculdade de Motricidade Humana. 8

FMV Faculdade de Medicina Veterinária. 8

FP Faculdade de Psicologia. 8

GM Gabinete de Mobilidade. 22

GMA Gabinete de Mobilidade e Acolhimento. 84, 85

ICS Instituto de Ciências Sociais. 8

ID Instituição de Destino/Acolhimento. xv, 25, 26, 28, 30–33, 51, 52, 54, 56, 57, 87, 90

IDE *Integrated Development Environment*. 10, 13, 37

IE Instituto de Educação. 8

IES Instituições de Ensino Superior. 31, 32, 36

IGOT Instituto de Geografia e Ordenamento do Território. 8

IIA *Inter-Institutional Agreement*. 25

IO Instituição de Origem. 30–33, 51–55, 62, 64, 81, 84, 85, 87, 90

ISA Instituto Superior de Agronomia. 8

ISCSP Instituto Superior de Ciências Sociais e Políticas. xii, xvi, 3, 8, 25, 90, 92

ISEG Instituto Superior de Economia e Gestão. xii, xvi, 3, 8, 25, 84, 86

IST Instituto Superior Técnico. vi, 8, 11, 29, 30

JVM *Java Virtual Machine*. 10

LA *Learning Agreement*. 2, 22, 24–28, 30, 31, 52, 70, 81, 84, 87, 88, 90, 91

MA Mobilidade Académica. 90, 91

MVC *Model-View-Controller*. 9

MVP *Minimum Viable Product*. 66, 69

NDS Núcleo de Desenvolvimento de *Software*. xvii, 29, 33, 35, 60, 111

NGSA Núcleo de Gestão de Serviços Académicos. 60, 61, 66

NM Núcleo de Mobilidade. 21, 25–27, 30, 31, 51, 58, 81, 82

POM Project Object Model. 37

PSL *Presentation Specific Language*. 9, 11–13, 15–17, 43, 46

qubIT Quorum Born IT. 3, 12

RAIDES Registo de Alunos Inscritos e Diplomados do Ensino Superior. 31, 81, 84, 87, 91

RUL Reitoria da Universidade de Lisboa. 12

SCULisboa Serviços Centrais da Reitoria da Universidade de Lisboa. 3, 8, 29

SGA Sistema de Gestão de Aprendizagem. 7

SGC Sistema de Gestão de Conteúdos. 7

SGE Sistema de Gestão de Estudantes. 7

SIGA Sistema Integrado de Gestão Académica. vi, 1–4, 7–10, 17, 20, 28

SMP *Student Mobility for Placements*. 2, 22, 41

SMS *Student Mobility for Studies*. 2, 22, 41

SQL *Structured Query Language*. 10

SUS *System Usability Scale*. 61

ToR *Transcript of Records*. 22, 24, 25, 27, 28, 31, 70, 82, 84, 85, 87, 88, 90, 91

UC Unidade Curricular. 22, 31, 81, 84, 87, 88, 90, 91

ULisboa Universidade de Lisboa. vi, viii, 1–4, 8, 26–32, 37, 47, 69, 70

Bibliografia

[1] Joao Cachopo, Luis Cruz, Susana Fernandes, Fernando Mira da Silva, Carlos Ribeiro, Antonio Rito-Silva, and Artur Ventura. The FenixEdu Project: an Open-Source Academic Information Platform. https://docplayer.net/6319893-The-fenixedu-project-an-open-source-academic-information-platform.html, 2011. [Online - Acedido em: 30/11/2023].

[2] Francisco Daniel Eleutério Caeiro. Gestão de propostas de formação avançada no sistema Fenix. Master's thesis, Universidade de Lisboa, Faculdade de Ciências, 2021.

[3] European Commission. Erasmus+ European Student Card Initiative - Erasmus Without Paper (EWP) Benefits. https://erasmus-plus.ec.europa.eu/european-student-card-initiative/ewp/benefits, 2023. [Online - Acedido em: 09/01/2024].

[4] European Commission. Erasmus+ European Student Card Initiative - Erasmus Without Paper (EWP) How It Works. https://erasmus-plus.ec.europa.eu/european-student-card-initiative/ewp/how-it-works, 2023. [Online - Acedido em: 09/01/2024].

[5] European Commission. Erasmus+ European Student Card Initiative - Erasmus Without Paper (EWP) How To Join. https://erasmus-plus.ec.europa.eu/european-student-card-initiative/ewp/how-to-join, 2023. [Online - Acedido em: 09/01/2024].

[6] European Commission. Erasmus+ European Student Card Initiative - Erasmus Without Paper (EWP) Overview. https://erasmus-plus.ec.europa.eu/european-student-card-initiative/ewp, 2023. [Online - Acedido em: 09/01/2024].

[7] Oracle Corporation. Java™ Programming Language. https://docs.oracle.com/javase/8/docs/technotes/guides/language/index.html, 2018. [Online - Acedido em: 06/12/2023].

[8] Ministério da Educação e Ciência. Decreto-Lei n.º 266-E/2012. https://files.dre.pt/1s/2012/12/25202/0027900290.pdf, 2012. [Online - Acedido em: 04/12/2023].

[9] Erasmus Without Paper+ Consortium. *Mandatory Business Requirements and Semantic Interoperability Specification for Nominations*, 1st edition, 2024. Manuscript completed in April 2024. [Online - Acedido em: Maio de 2024.].

[10] Eclipse Foundation. Eclipse IDE. `https://www.eclipse.org/ide/`, 2023. [Online - Acedido em: 06/12/2023].

[11] MariaDB Foundation. About MariaDB. `https://mariadb.org/about/`, 2023. [Online - Acedido em: 06/12/2023].

[12] Object Management Group. Business process model and notation (bpmn) version 2.0. `https://www.omg.org/spec/BPMN/2.0`, 2011. Consultado em: 12 de novembro de 2024.

[13] Ibtisam Rauf Abdul Majeed. MVC Architecture: A Detailed Insight to the Modern Web Applications Development. `https://docslib.org/doc/4505403/mvc-architecture-a-detailed-insight-to-the-modern-web-applications-development`, 2018. [Online - Acedido em: 05/12/2023].

[14] Diana Paiva Marques. Gestão de Alunos Erasmus no Sistema Fénix. Master's thesis, Universidade de Lisboa, Faculdade de Ciências, 2021.

[15] Miguel Ramalho Morais. Automação dos Procedimentos de Provas de Agregação: Uma Solução Baseada em Processos Inter-Instância do Sistema Fenix. Master's thesis, Universidade de Lisboa, Faculdade de Ciências, 2022.

[16] João Filipe Proença Neto. Sistema de Garantia da Qualidade para o Sistema Integrado de Gestão Académica Fénix. Master's thesis, Universidade de Lisboa, Faculdade de Ciências, 2022.

[17] Oracle. The Java™ Tutorials: Getting Started. `https://docs.oracle.com/javase/tutorial/getStarted/intro/definition.html`, 2022. [Online - Acedido em: 06/12/2023].

[18] Erasmus Without Paper. EWP specifications: Architecture. `https://github.com/erasmus-without-paper/ewp-specs-architecture#cnr`, n.d. Consultado Online a 30 de outubro de 2024. Última atualização: 13 de fevereiro de 2024, versão 1.15.0.

[19] Erasmus Without Paper. EWP specifications: Mobility flowcharts. `https://github.com/erasmus-without-paper/ewp-specs-mobility-flowcharts`, n.d. Consultado Online a 30 de outubro de 2024. Última atualização: 11 de julho de 2023, versão 0.5.1.

[20] Apache Maven Project. What is Maven? `https://maven.apache.org/what-is-maven.html`, 2023. [Online - Acedido em: 06/12/2023].

[21] Amazon Web Services. Docker on AWS. https://aws.amazon.com/docker/, 2023. [Online - Acedido em: 06/12/2023].

[22] Simplilearn. Git Tutorial. https://www.simplilearn.com/tutorials/git-tutorial/what-is-git, 2022. [Online - Acedido em: 06/12/2023].

[23] Simplilearn. What is Tomcat? https://www.simplilearn.com/what-is-tomcat-article, 2023. [Online - Acedido em: 06/12/2023].

Apêndice A

Macroprocesso de Mobilidade Erasmus

A.1 Macroprocesso de Mobilidade Erasmus+

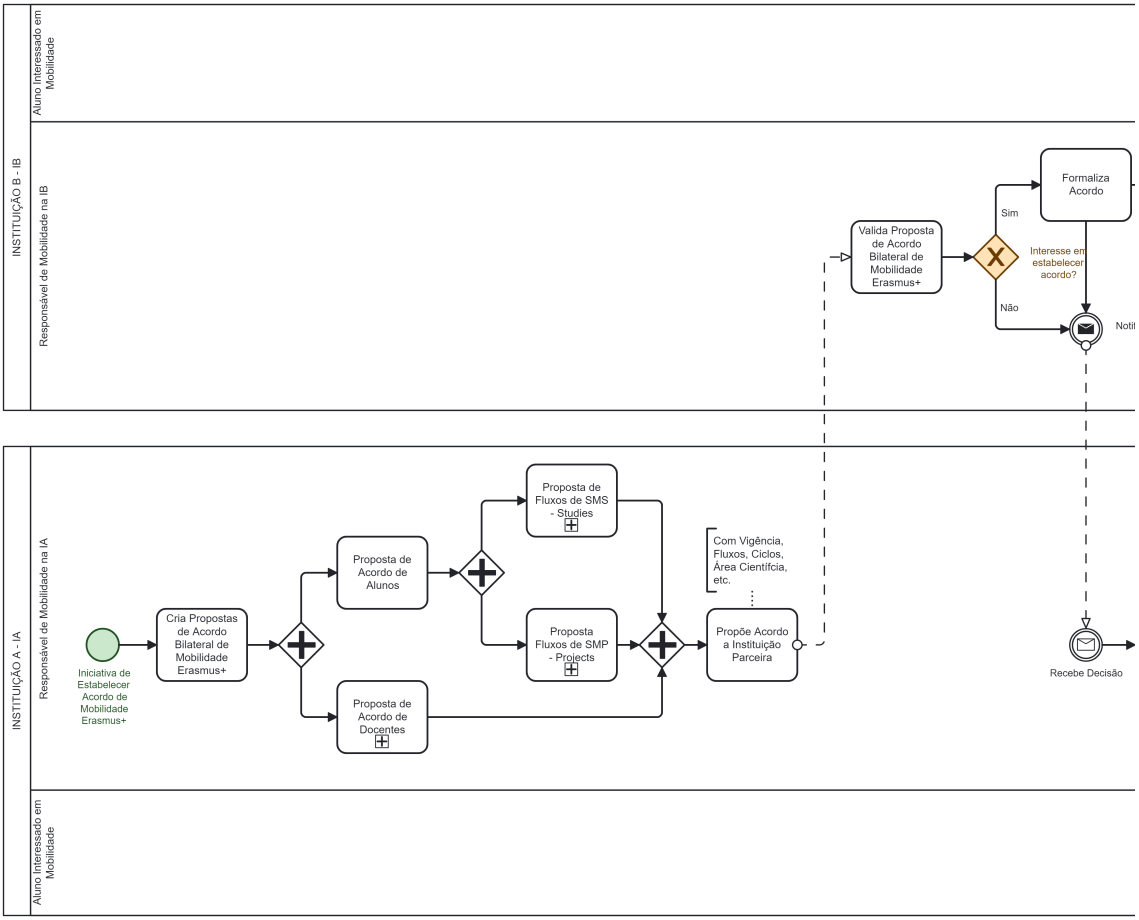


Figura A.1: Macroprocesso de Mobilidade Erasmus+ Parte 1.

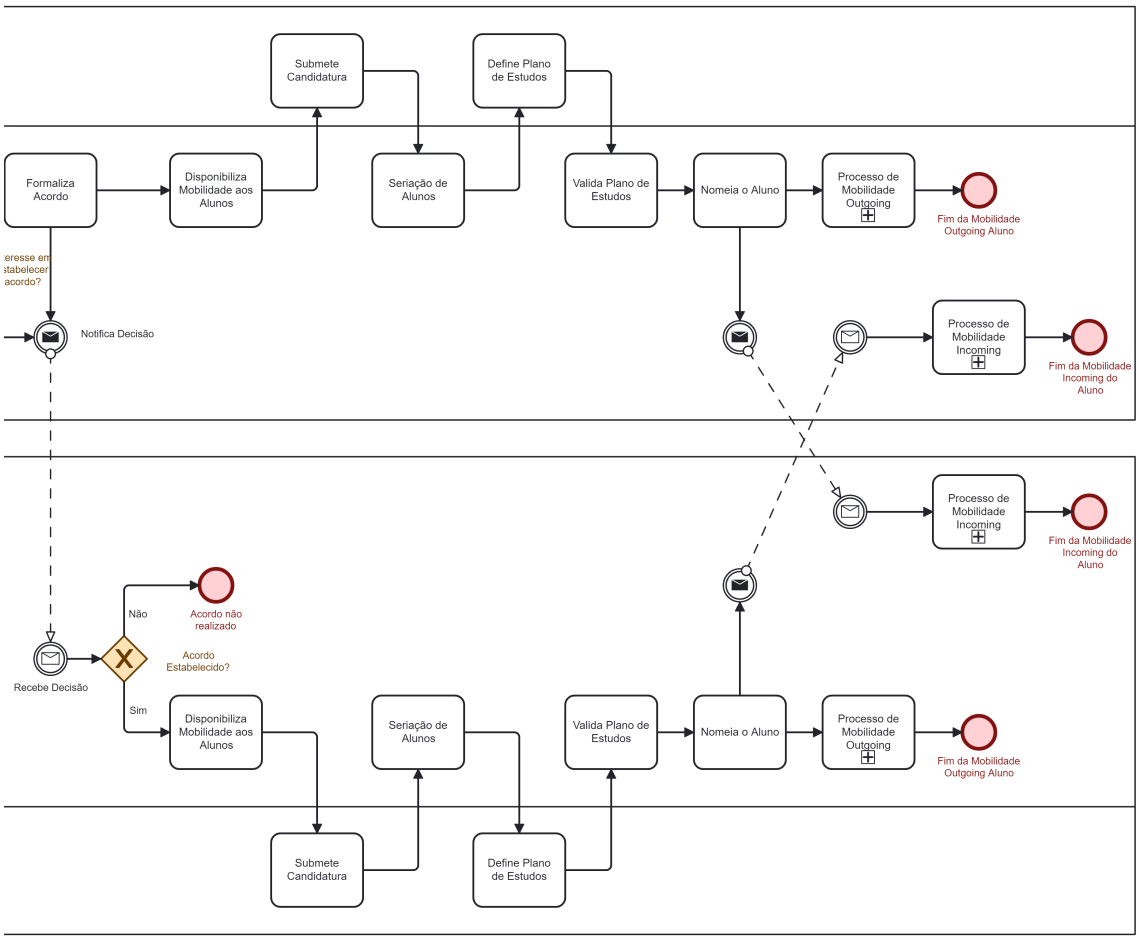


Figura A.2: Macroprocesso de Mobilidade Erasmus+ Parte 2.

Apêndice B

Processo de Mobilidade Erasmus *Incoming*

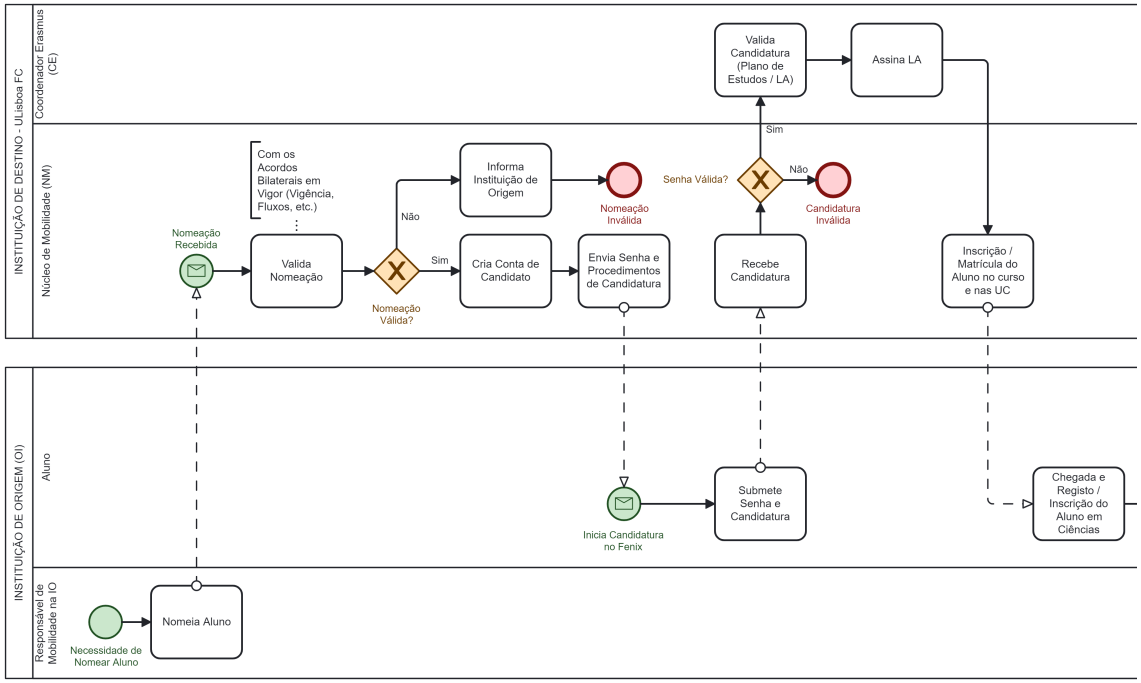
B.1 Faculdade de Ciências (FC)

Detalhe e respectiva representação BPMN, do processo de mobilidade Erasmus *Incoming* de FC, apurado na reunião de levantamento de requisitos de dia 11 de Dezembro de 2023.

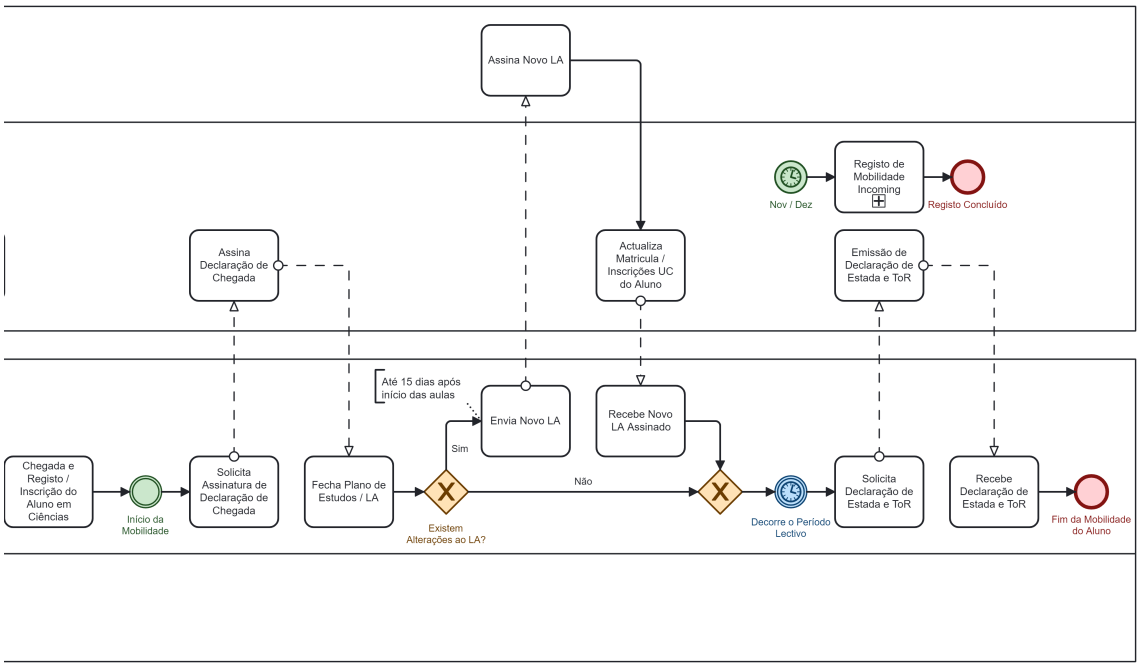
1. O NM de FC recebe por email, a nomeação do aluno em mobilidade (de Janeiro até Maio para ano lectivo seguinte, até Outubro para o segundo semestre).
2. O NM valida os dados necessários e constrói/alimenta uma base de dados interna com informações que detalharemos numa segunda fase (nome, IO, código de instituição, etc).
3. O NM formaliza a inscrição/matrícula do aluno no curso e nas UCs aprovadas no LA.
4. O NM cria registo de mobilidade *incoming* no Fenix (RAIDES).
    (a) Este registo apenas costuma ser efetuado em Dezembro/Janeiro.
5. Dá-se início oficialmente à mobilidade com a emissão da declaração de chegada.
    (a) É o aluno que solicita este documento, após a sua chegada e registo/inscrição em Ciências.
6. A partir do início das aulas e até ao início de Outubro, os alunos podem solicitar alterações ao LA (cerca de 15 dias).
    (a) O aluno solicita alteração, enviando o LA da IO com as alterações desejadas ao NM, que por sua vez, envia para validação do coordenador de Erasmus.
    (b) Caso o coordenador de Erasmus aprove, o NM reflete essas alterações de inscrição a UCs no Fenix e o novo LA, passa a ser o final.
7. Decorre o período de mobilidade de 1 ano, (1º semestre) ou de 2º semestre.

8. Fim da mobilidade com emissão de ToR (documento gerado automaticamente pelo Fenix) e declaração de estada/saída.

   (a) É o aluno que tem o ónus de solicitar os documentos ao NM.



(a) Parte 1.



(b) Parte 2.

Figura B.1: Processo de Mobilidade Erasmus *Incoming* - FC.

B.2 Instituto Superior de Economia e Gestão (ISEG)

Detalhe e respectiva representação BPMN, do processo de mobilidade Erasmus *Incoming* de ISEG, apurado na reunião de levantamento de requisitos de dia 13 de Dezembro de 2023.

1. Gabinete de Mobilidade e Acolhimento (GMA) do ISEG recebe a nomeação do aluno em mobilidade.
   (a) Nomeação realizada no Fenix pela IO, através do *workflow*: *Nomeação Candidaturas Erasmus*.
   (b) Prazos: Até 30 de Abril para o 1º ano/1º Semestre; 1 a 30 de Setembro para o 2º Semestre.
2. GMA valida a nomeação conforme os acordos/fluxos em vigor existentes.
3. GMA cria e disponibiliza ao aluno, conta de candidato para este iniciar candidatura.
   (a) candidatura realizada no Fenix pelo aluno, através do *workflow*: *Candidaturas Incoming*.
   (b) Prazos: Até 31 de Maio para 1º ano/1º Semestre; 2º Semestre até 31 de Outubro.
4. GMA valida os dados da candidatura e formaliza a matrícula do aluno (Dados serão aprofundados numa segunda fase, mas incluem, por exemplo: ID, ToR e LA).
5. O aluno dá entrada no ISEG.
   (a) GMA cria registo de mobilidade *Incoming* no Fenix (RAIDES) - Dez/Jan.
6. Início da mobilidade com início das aulas.
7. Assinatura da declaração de chegada a pedido do aluno.
   (a) A maioria das IO tem os seus próprios modelos de declarações, pelo que, o GMA, assina as declarações que lhes são entregues pelos alunos.
   (b) Não são emitidas declarações, salvo pontuais pedidos.
8. A partir do início das aulas, os alunos têm cerca de 15 dias para alterações ao plano de estudos.
   (a) A formalização do novo LA pode ocorrer posteriormente.
   (b) Caso exista alteração ao plano de estudos, seja por iniciativa do aluno, seja por limitação de vagas, seja por não abertura de UCs, etc. o aluno apenas tem que se inscrever/alterar UCs pretendidas e enviar posteriormente o LA atualizado.
   (c) Ao receber o LA atualizado,o GMA valida se as UCs às quais o aluno está efetivamente inscrito, correspondem ao que está no LA.

9. Decorre o período de mobilidade (1 ano ou 1 semestre).

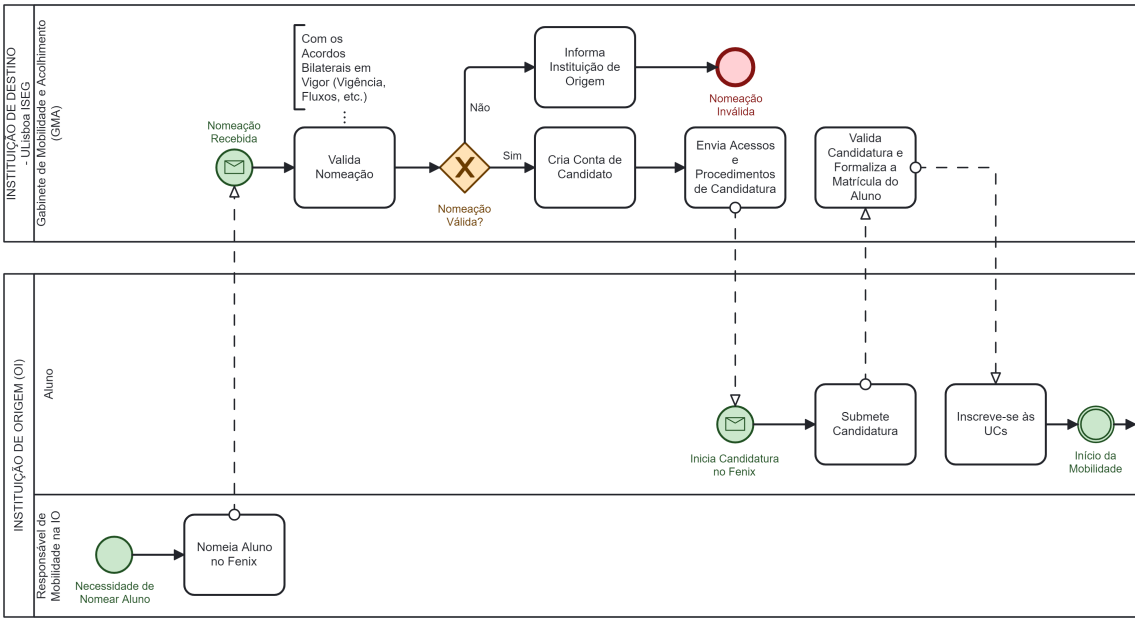
10. Fim da mobilidade.

    (a) Emissão de ToR (Fenix).

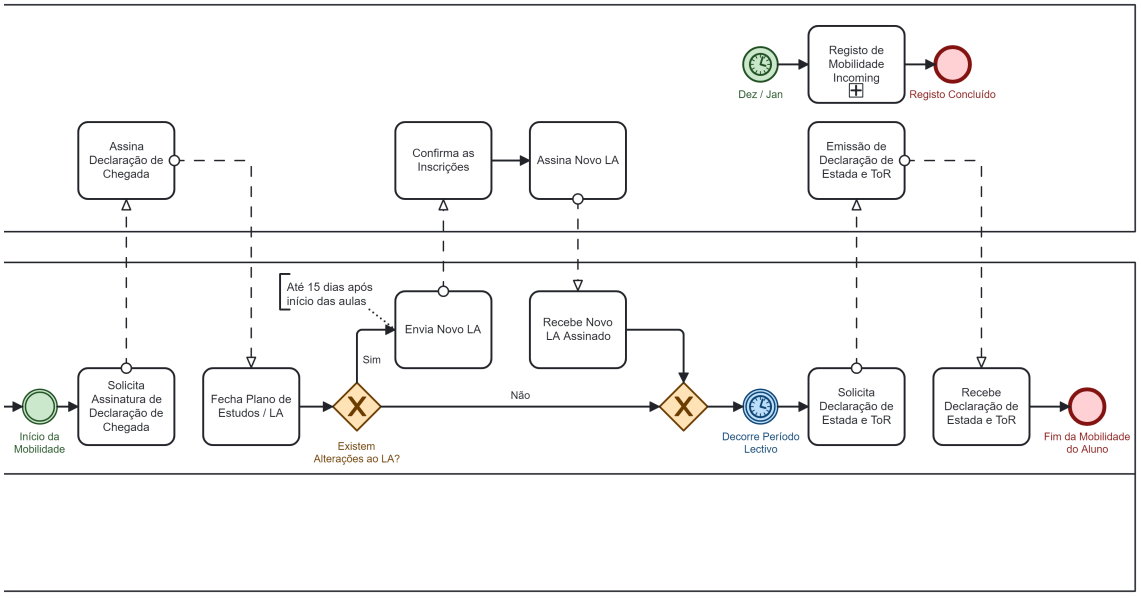
    (b) Assinatura da declaração de saída/estada a pedido do aluno e após exames.

    (c) Tal como nas declarações de chegada, a maioria das IO tem os seus próprios modelos de declarações de estada, pelo que, novamente, o GMA assina as declarações que lhes são entregues pelos alunos.

    (d) Não são emitidas declarações, salvo pontuais pedidos.



(a) Parte 1.



(b) Parte 2.

Figura B.2: Processo de Mobilidade Erasmus *Incoming* - ISEG.

B.3 Faculdade de Letras (FL)

Detalhe e respectiva representação BPMN, do processo de mobilidade Erasmus *Incoming* da FL, apurado na reunião de levantamento de requisitos de dia 15 de Dezembro de 2023.

1. A Divisão de Relações Externas e Internacionais (DREI) da Faculdade de Letras (FL) recebe a nomeação do aluno em mobilidade.
   (a) nomeação realizada pela IO, através do preenchimento de um formulário online para o efeito.
   (b) Prazos: Durante o mês de Abril para o 1º ano/1º Semestre. Durante o mês de Setembro para o 2º Semestre.
2. A DREI valida manualmente a nomeação conforme os acordos/fluxos em vigor existentes.
3. De seguida o aluno tem que se registar para iniciar a candidatura.
   (a) Candidatura realizada e formalizada no Fenix pelo aluno, através do *workflow*: *Candidaturas Incoming V3*.
   (b) Prazos: Durante o mês de Maio para 1º ano/1º Semestre; 2º Semestre ocorre durante o mês de Outubro.
4. A DREI valida os dados da candidatura submetida para análise (Os dados serão aprofundados numa segunda fase, mas incluem: ID, ToR, LA, fotografia, entre outros).
5. O aluno dá entrada na FL.
   (a) A matrícula e inscrição são da responsabilidade do aluno de mobilidade.
   (b) Prazos: De 1 a 11 de Setembro para 1º ano/1º Semestre; 2º Semestre 2 a 22 de Janeiro.
   (c) A DREI cria registo de mobilidade *Incoming* no Fenix (RAIDES) - Dez/Jan.
6. Início da mobilidade com o início das aulas.
   (a) *Welcome Session* (WS) a 7 de Setembro (1 ano/1º Semestre) e a 18 de Janeiro (2º Semestre) – ambos uns dias antes do início das aulas.
   (b) A partir deste momento, cabe ao aluno solicitar a assinatura/emissão da Declaração de Chegada ao DREI através do *workflow* de requerimento, para o efeito.
7. Alterações à mobilidade: A partir do início das aulas, os alunos têm cerca de 15 dias (no primeiro semestre) e 5 dias (no segundo semestre) para alterações ao plano de estudos.
   (a) Caso existam alterações ao plano de estudos, seja por iniciativa do aluno, seja por limitação de vagas, seja por não abertura de UCs, etc., o aluno tem que se inscrever/alterar UCs pretendidas e requerer posteriormente a modificação do LA.

i. As alterações às UCs devem seguir o processo definido para alteração de inscrição em unidades curriculares e mudança de turma, recorrendo ao requerimento de alteração de inscrição com aprovação do(s) Docente(s).

ii. A formalização do novo LA com base nas alterações de UCs, deve ocorrer posteriormente através do *workflow de requerimento: LA During* e é da responsabilidade do aluno.

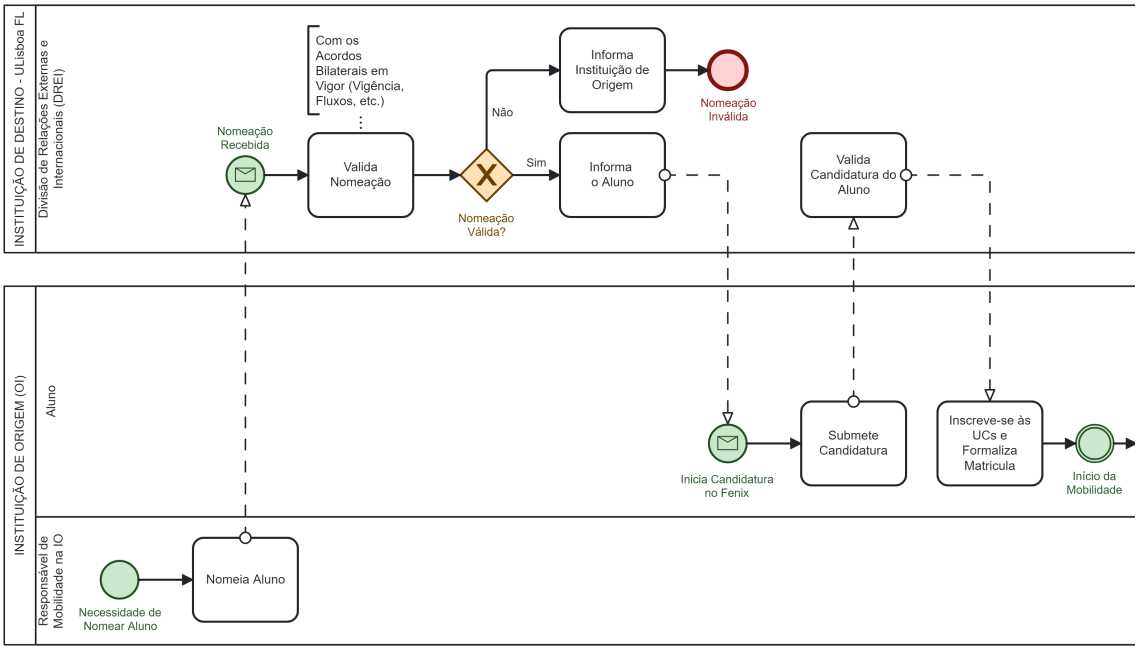
(b) Ao receber o LA atualizado, a DREI valida se as UCs às quais o aluno está efetivamente inscrito, correspondem ao que está no LA e assina o documento.

8. Decorre o período de mobilidade (1 ano ou 1 semestre).

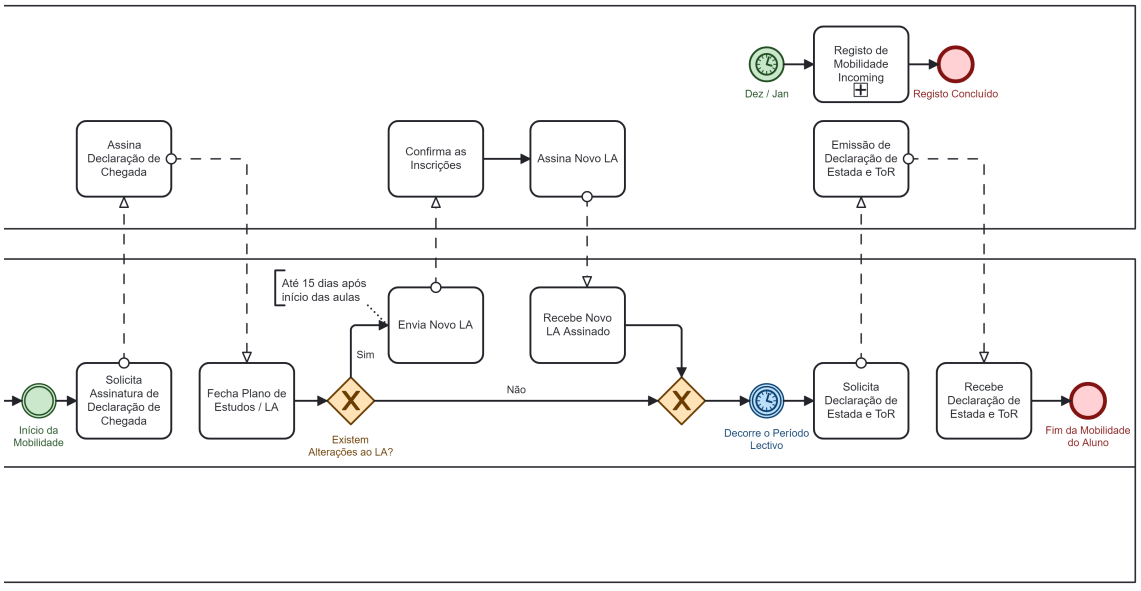
9. Fim da mobilidade.

(a) O DREI notifica o aluno do fim da mobilidade.

(b) Os alunos podem requerer o ToR e o da Declaração de Saída à DREI, através dos *workflows* de requerimentos existentes para o efeito.



(a) Parte 1.



(b) Parte 2.

Figura B.3: Processo de Mobilidade Erasmus *Incoming* - FL.

B.4 Instituto Superior de Ciências Sociais e Políticas (ISCSP)

Detalhe e respectiva representação BPMN, do processo de mobilidade Erasmus *Incoming* da ISCSP, apurado na reunião de levantamento de requisitos de dia 19 de Dezembro de 2023.

1. A Mobilidade Académica (MA) do ISCSP recebe a nomeação do aluno em mobilidade.
   (a) Nomeação realizada pela Instituição Parceira de Origem (IO), através de um *Google Forms* criado e disponibilizado pela MA, para o efeito.
   (b) Prazos: Até 1 de Junho 1º ano/1º Semestre; até 1 de Novembro, 2º Semestre.
2. MA valida a nomeação conforme os Acordos/Fluxos em vigor existentes.
3. MA informa o aluno de que se deve registar para iniciar a candidatura.
   (a) Candidatura realizada no Fenix pelo aluno, através do *workflow*: *Mobilidade Alunos Incoming*.
   (b) Prazos: Até 15 de Junho para 1º ano/1º Semestre; 2º Semestre até 15 de Novembro.
4. MA valida os dados da candidatura e reencaminha o LA, para a assinatura da Coordenadora Institucional de Erasmus (CIE) (Os dados serão aprofundados numa segunda fase, mas incluem: ID, ToR, LA, etc.).
5. Com LA assinado, MA gera Carta de Aceitação e partilha estes dois documentos com o aluno.
   (a) Documentação gerada/guardada e partilhada com o aluno, através do *Processo Individual Digital* (*Workflow* de requerimento).
   (b) Este processo de requerimento é iniciado pela MA.
   (c) Tem toda a documentação relacionada com a mobilidade do aluno, servindo como "repositório".
   (d) Está disponível para o MA e para o aluno.
6. Início da mobilidade com o início das aulas.
   (a) *Welcome Session* (WS) na sexta-feira antes do início das aulas.
   (b) Após a WS, a MA formaliza a matricula no Fenix a todos os alunos que tiveram presentes na sessão. Após este procedimento, os alunos ficam "Matriculados - com conta criada".
   (c) A partir deste momento, abre o processo de inscrição.
7. O aluno inscreve-se às UCs e formaliza a matricula.
8. Alterações à mobilidade: A partir do início das aulas, os alunos têm cerca de 15 dias para alterações ao plano de estudos.

   (a) Caso existam alterações ao plano de estudos, seja por iniciativa do aluno, seja por limitação de vagas, seja por não abertura de UC, etc, o aluno tem que se inscrever/alterar UCs pretendidas e requerer, posteriormente, a assinatura do LA atualizado.

   (b) Ao receber o LA atualizado, a MA valida se as UCs às quais o aluno está efetivamente inscrito, correspondem ao que está no LA e remete para assinatura da CIE.

   (c) CIE devolve a MA, que atualiza o *Processo Individual Digital*.

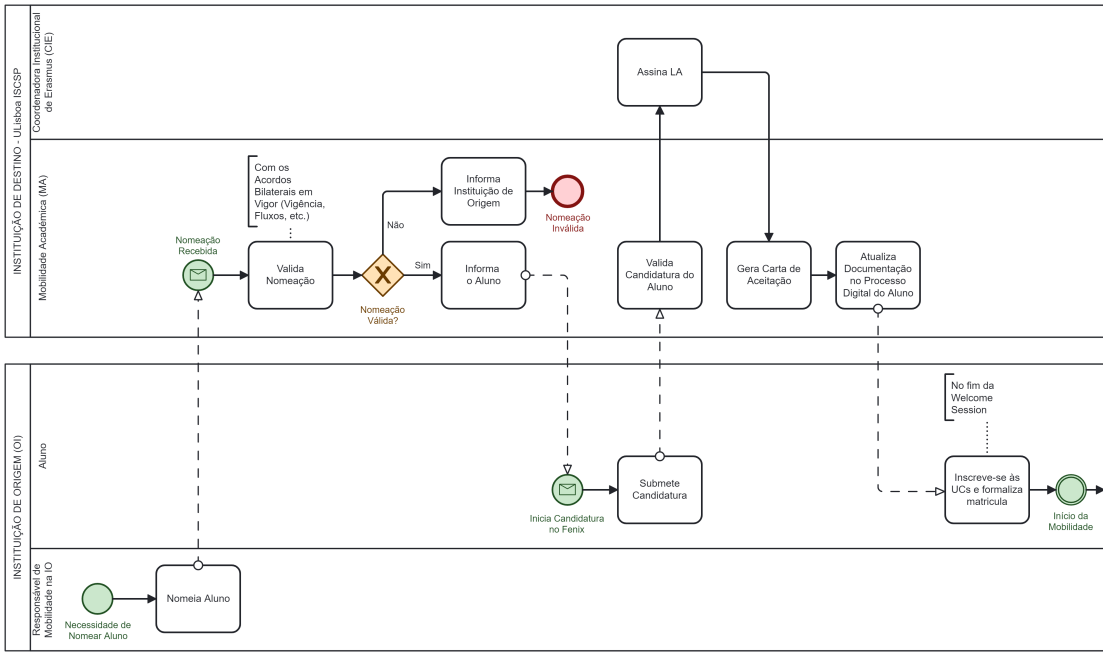
9. A MA cria registo de mobilidade *Incoming* no Fenix (RAIDES) em Novembro e Fevereiro/Março.

10. Decorre o período de mobilidade (1 ano ou 1 semestre).

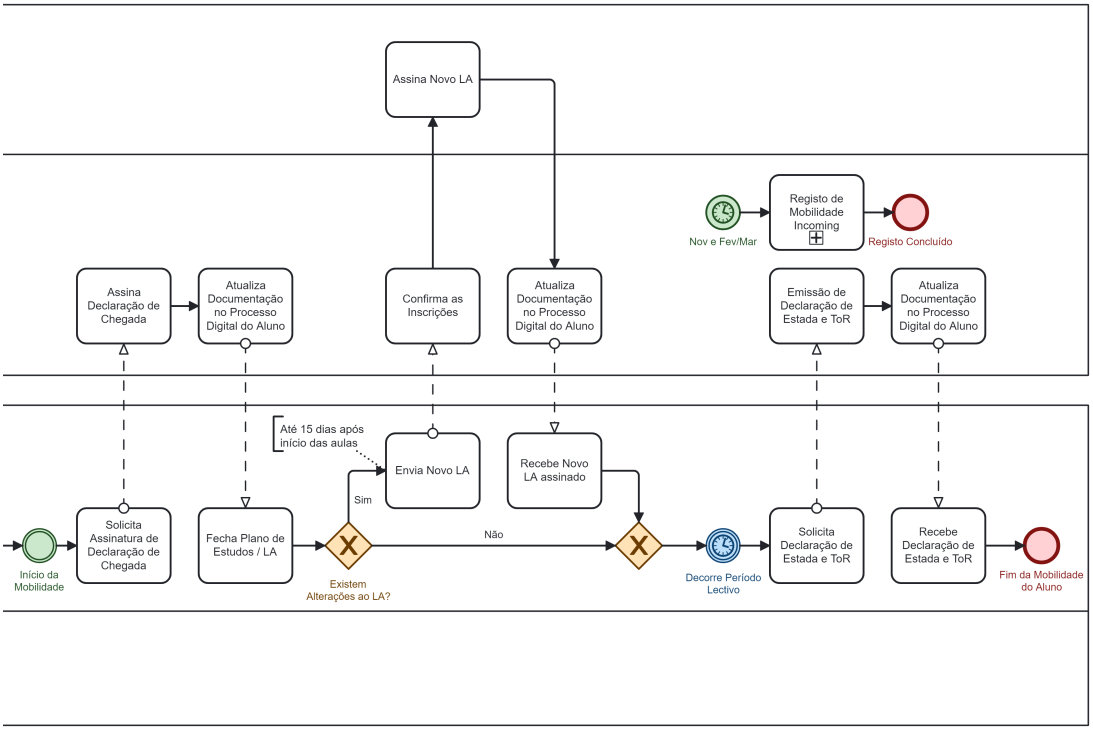
11. Fim da mobilidade.

   (a) O aluno pode solicitar o ToR através do requerimento: *Transcript of Records*.

   (b) E a declaração de estada/saída através do Processo Individual Digital.



(a) Parte 1.



(b) Parte 2.

Figura B.4: Processo de Mobilidade Erasmus *Incoming* - ISCSP.

Apêndice C

Mobilidade *Incoming* ULisboa

C.1 Desenho do Processo de Mobilidade Erasmus *Incoming* ULisboa

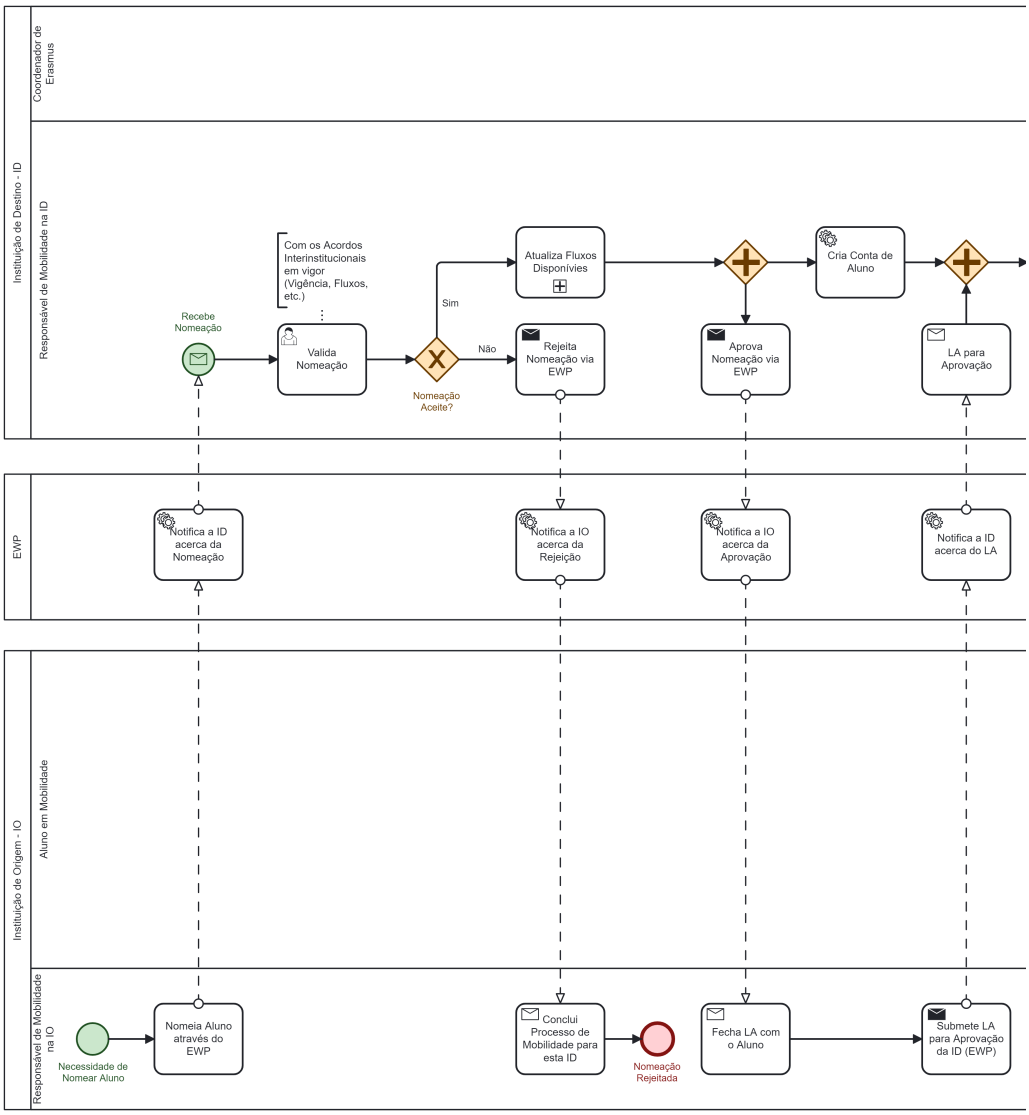


Figura C.1: Proposta de Processo de Mobilidade Erasmus *Incoming* ULisboa - Parte 1.

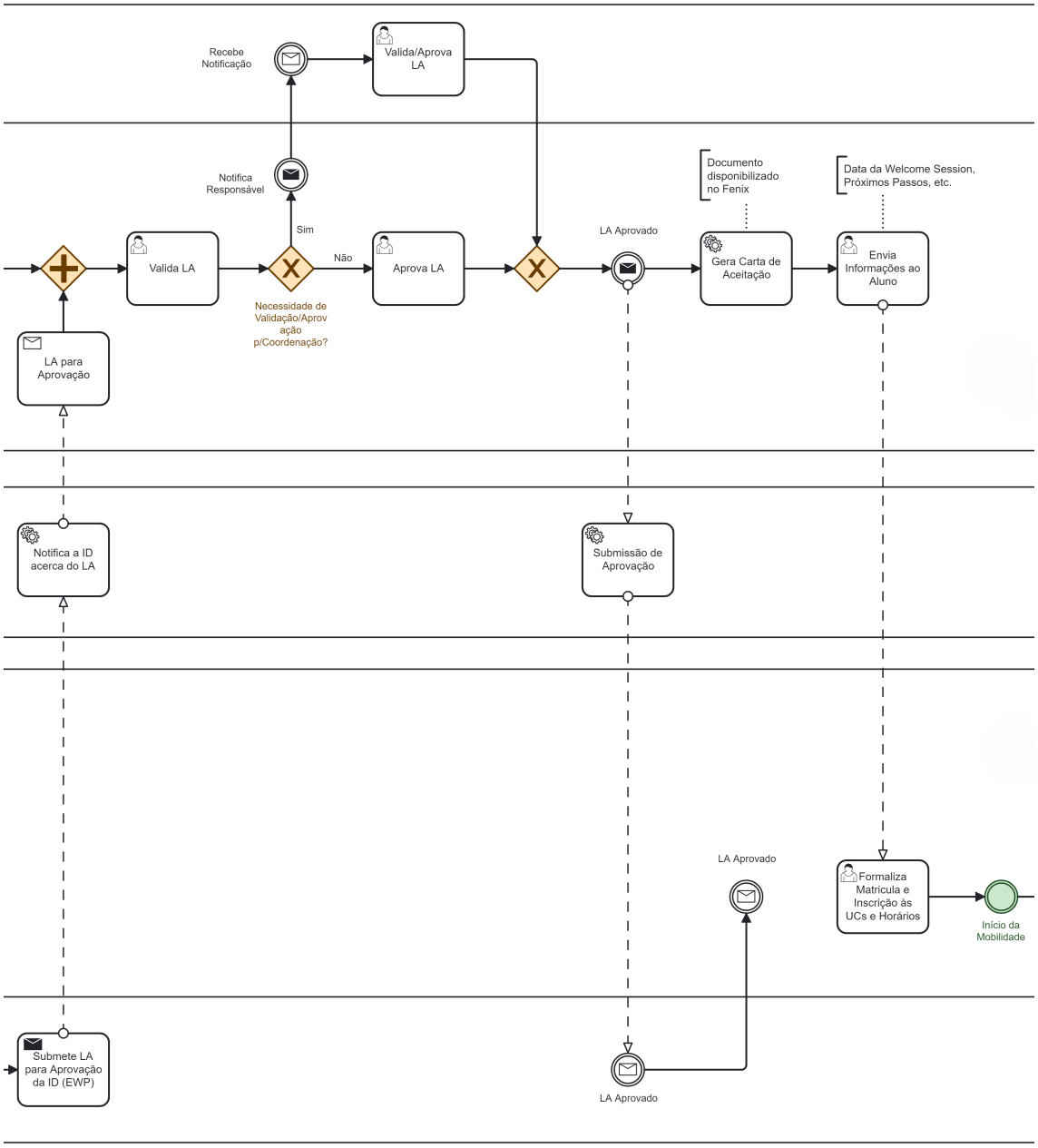


Figura C.2: Proposta de Processo de Mobilidade Erasmus *Incoming* ULisboa - Parte 2.

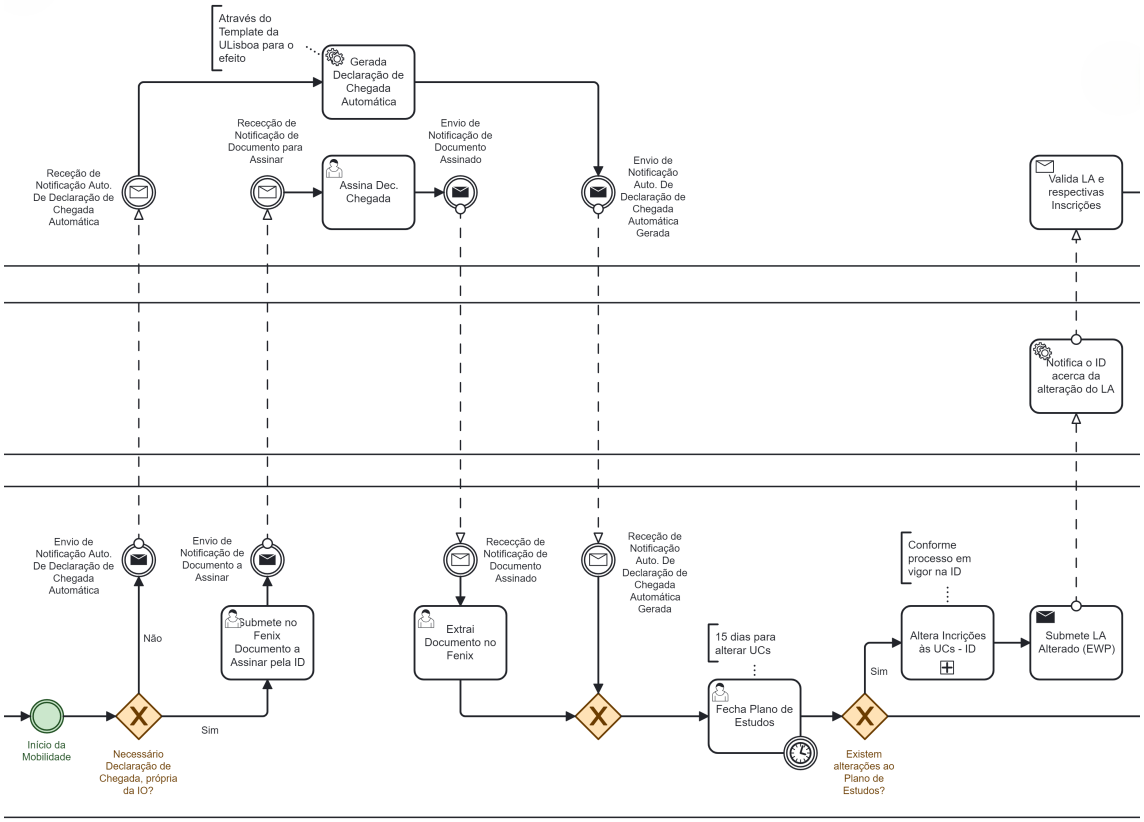


Figura C.3: Proposta de Processo de Mobilidade Erasmus *Incoming* ULisboa - Parte 3.

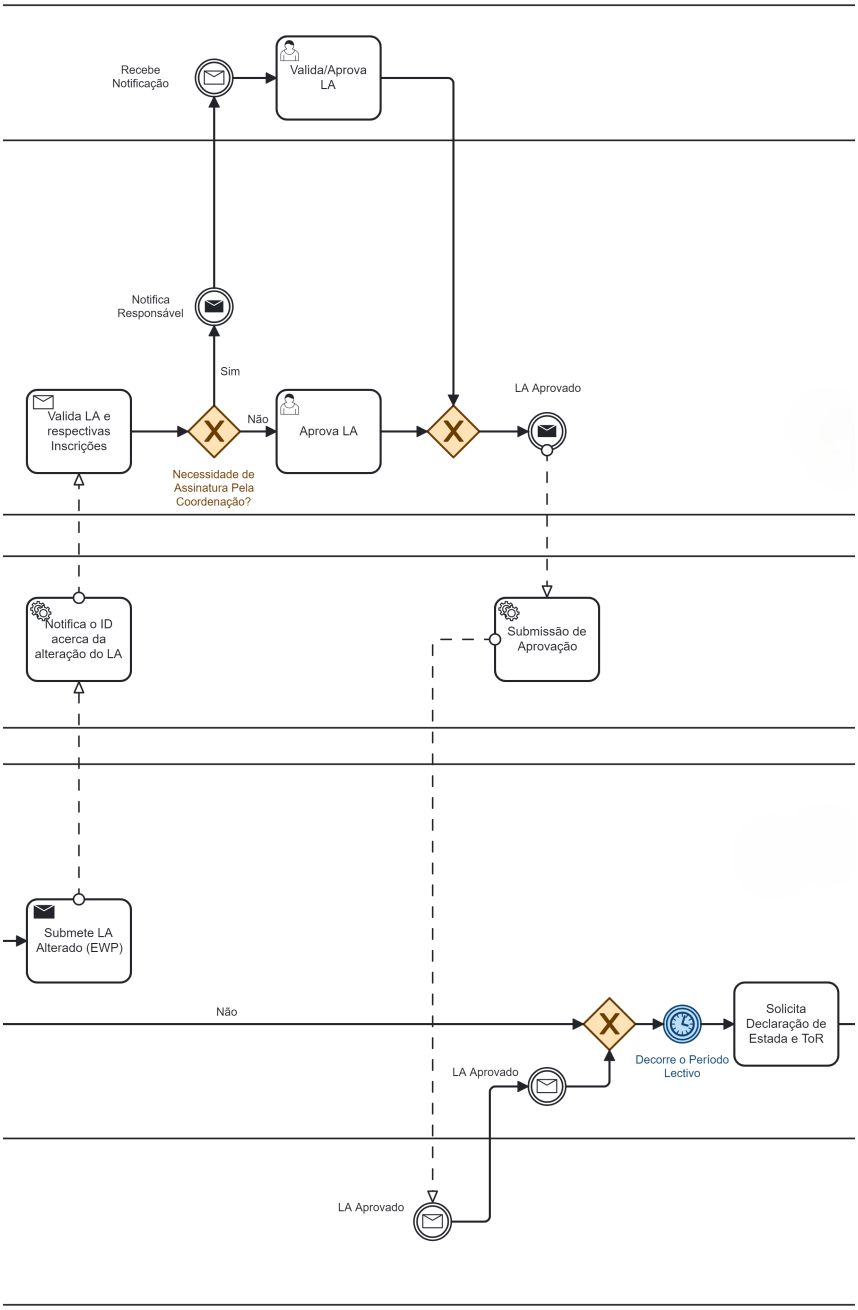


Figura C.4: Proposta de Processo de Mobilidade Erasmus *Incoming* ULisboa - Parte 4.

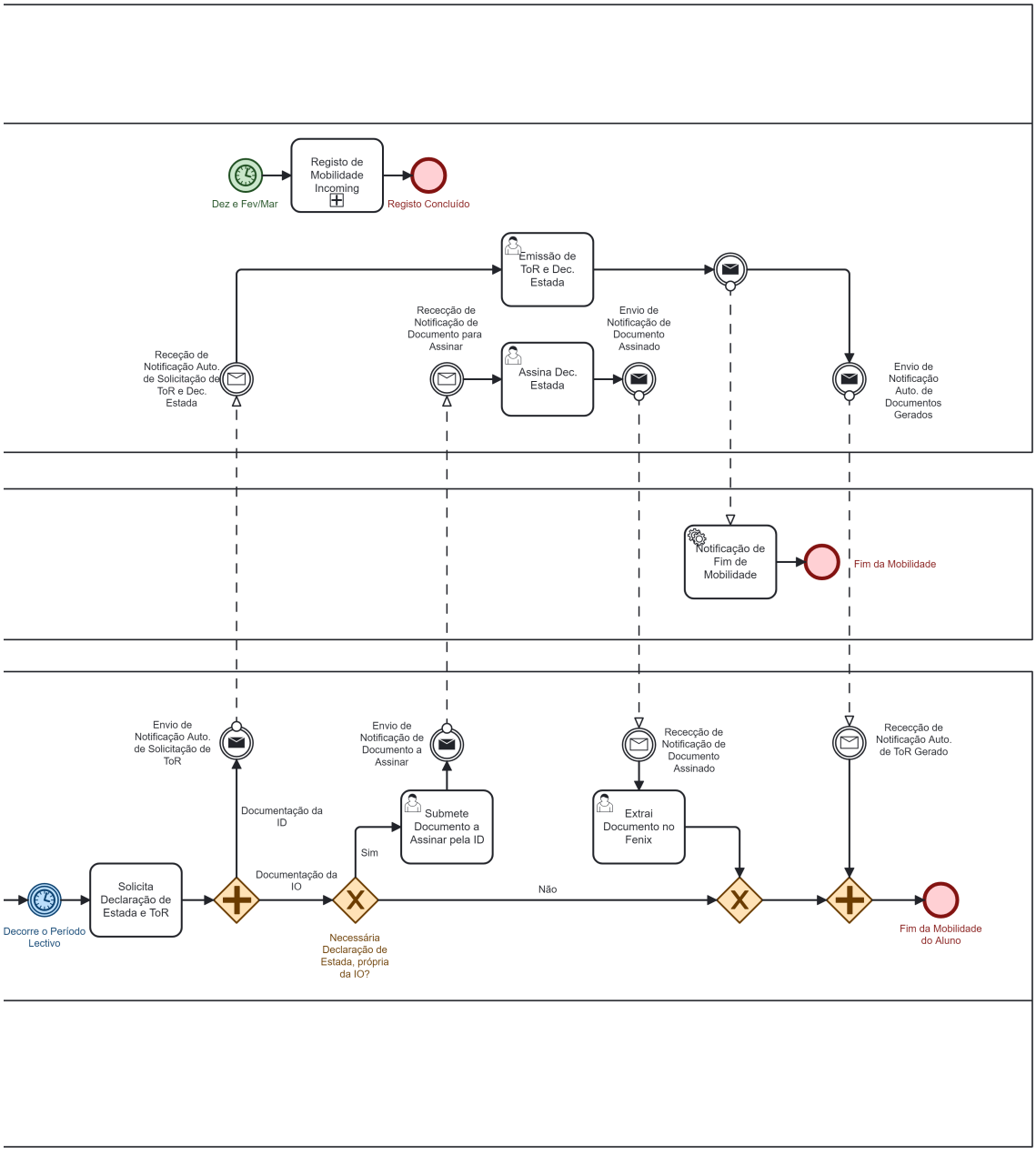


Figura C.5: Proposta de Processo de Mobilidade Erasmus *Incoming* ULisboa - Parte 5.

Apêndice D

Workflow Fenix - Mobilidade *Incoming*

D.1 Detalhe Workflow Fenix - Mobilidade *Incoming*

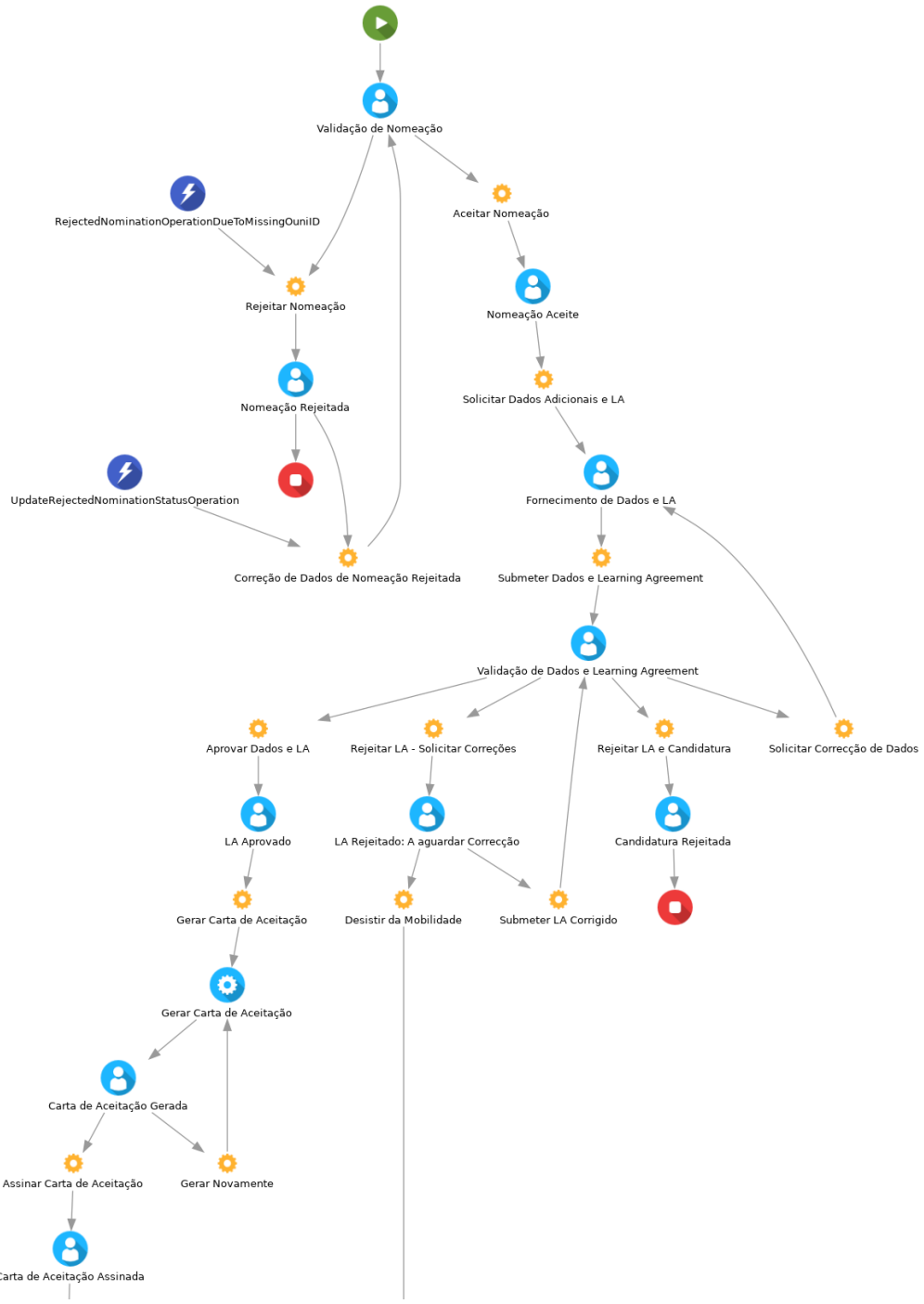


Figura D.1: *Workflow* de Mobilidade *Incoming* - Parte 1.

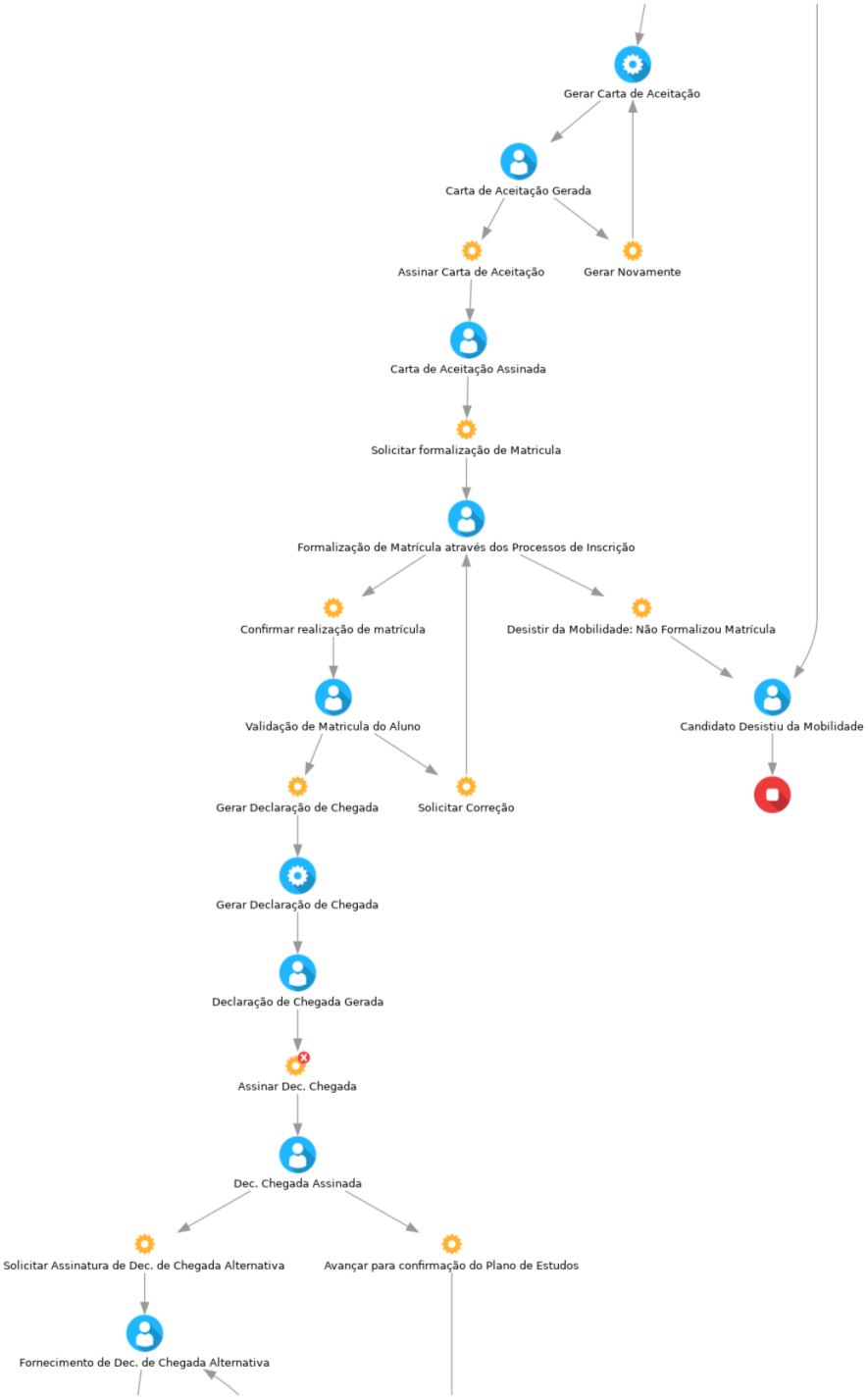


Figura D.2: *Workflow* de Mobilidade *Incoming* - Parte 2.

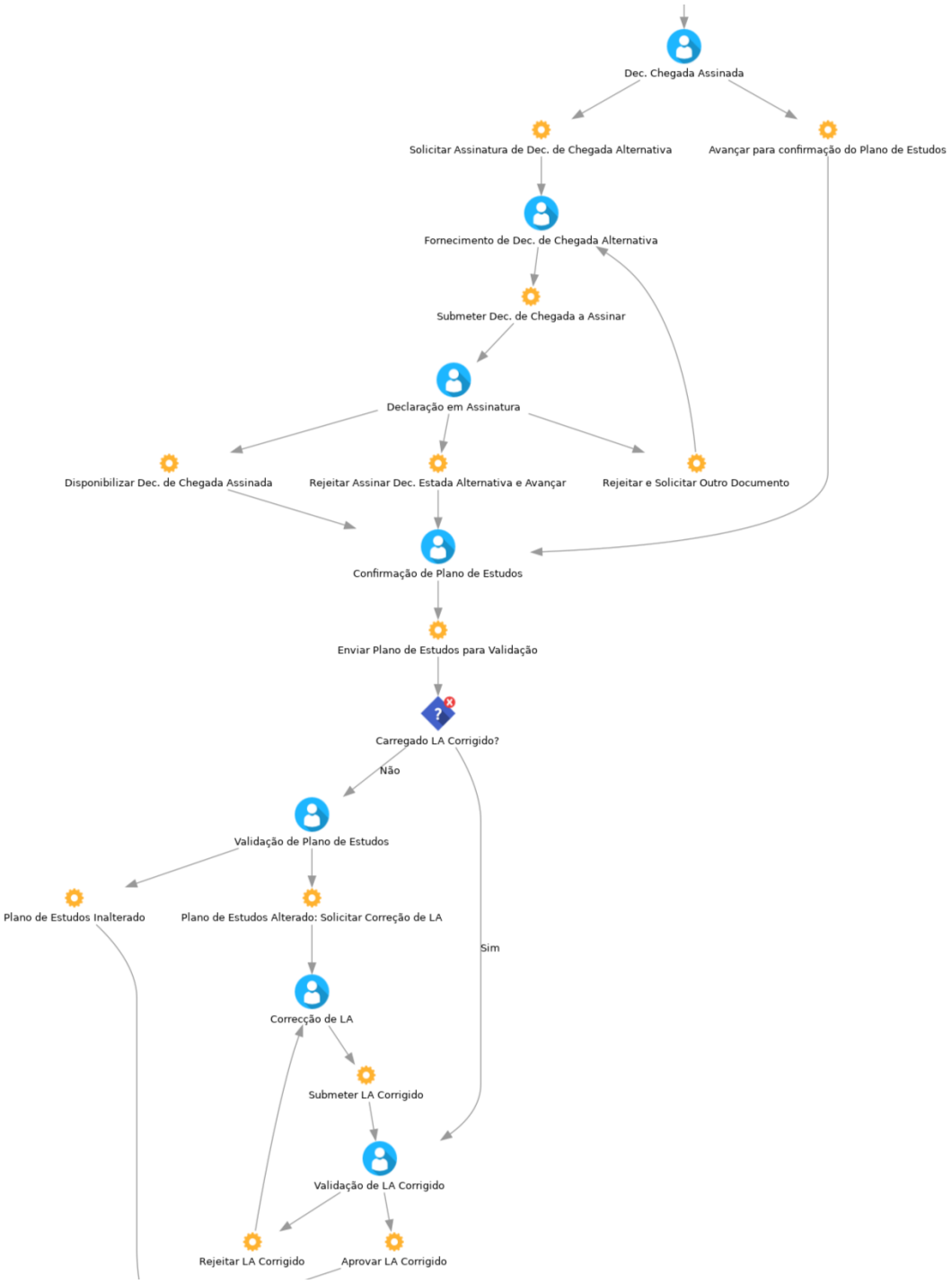


Figura D.3: *Workflow* de Mobilidade *Incoming* - Parte 3.

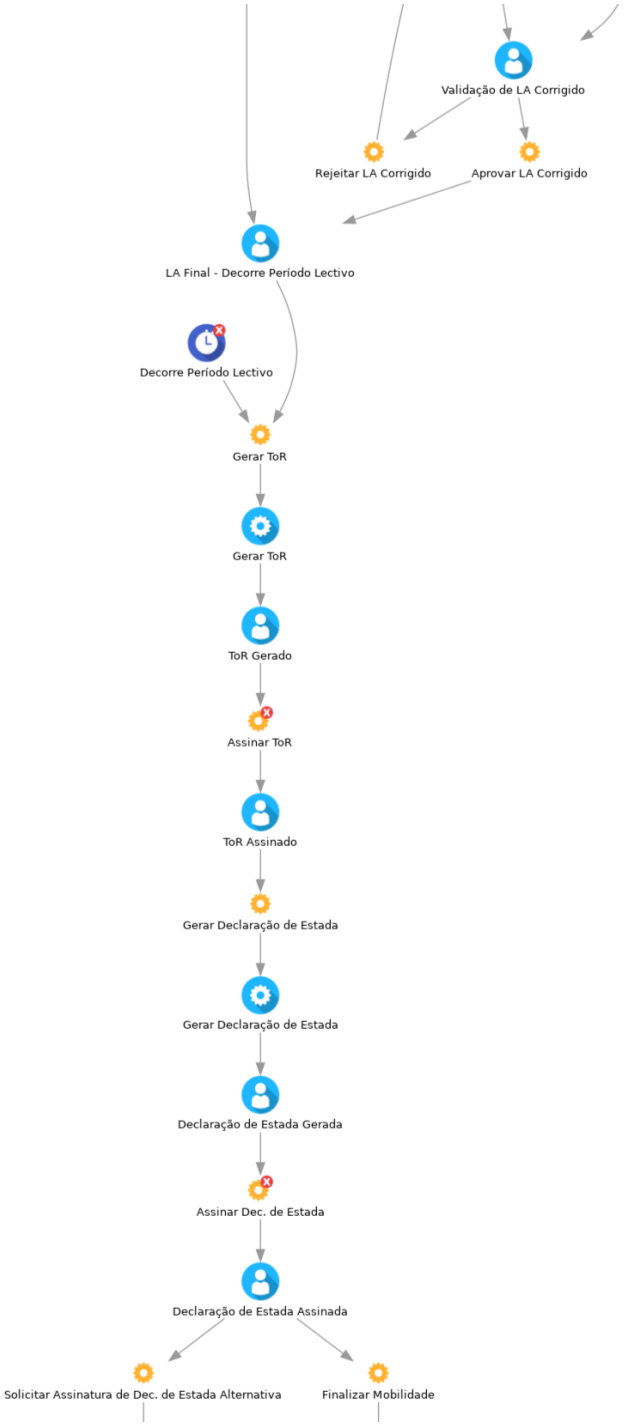


Figura D.4: *Workflow* de Mobilidade *Incoming* - Parte 4.

Figura D.5: *Workflow* de Mobilidade *Incoming* - Parte 5.

D.2 Documentação do Workflow Fenix - Mobilidade *Incoming*

Estados	Descrição	Man/Aut.	Notificações	Notificações	Mensagem	Separadores	Dados	Permissão W	Permissão R	Validações de Separadores	Operações	Permissões de Operações	Estado de Destino
									Serviços Centrais + Manager				
Validação de Nomeação	NM valida a Nomeação, ou seja, os dados recebidos via EVP e, aceita ou rejeita a Nomeação.	Man	Sim, NM	À entrada de estado		Monitorização do Processo	Workflow nominationType omobilityId ActivityAttributes ActivityType MobilityStatus NominationStatus receivingAcademicYearId sendingAcademicTermEwpId sendingHeiComment [Optional] eqfLevelStudiedAtDeparture agreementIscedFCode nomineeIscedFCode sendingHeiId sendingOunitId [Optional] receivingOunitId [Optional] sendingIaId [Optional] receivingIaId [Optional]	Serviços Centrais	candidato, NM		Aceitar Nomeação	NM	Nomeação Aceite
											Rejeitar Nomeação	NM	Nomeação Rejeitada
						Dados da Nomeação	globalId gender birthDate nationality email givenNames familyName:		candidato, NM				
						Dados do Incoming Student							
						Comentário da Instituição de Dest	receivingHeiComment [Optional]	NM	NM				
Nomeação Rejeitada	NM rejeitou a Nomeação	FINAL	Sim, Aluno	À entrada de estado		Monitorização do Processo	Workflow nominationType omobilityId ActivityAttributes ActivityType MobilityStatus NominationStatus receivingAcademicYearId sendingAcademicTermEwpId sendingHeiComment [Optional] eqfLevelStudiedAtDeparture agreementIscedFCode nomineeIscedFCode sendingHeiId sendingOunitId [Optional] receivingOunitId [Optional] sendingIaId [Optional] receivingIaId [Optional]	Serviços Centrais	candidato, NM				
						Dados da Nomeação	globalId gender birthDate nationality email givenNames		candidato, NM				
						Dados do Incoming Student	familyName		candidato, NM				
						Comentário da Instituição de Dest	receivingHeiComment [Optional]		candidato, NM				
Nomeação Aceite	NM aceitou a Nomeação	Man	Sim, Aluno	À entrada de estado		Monitorização do Processo	Workflow nominationType omobilityId ActivityAttributes ActivityType MobilityStatus NominationStatus receivingAcademicYearId sendingAcademicTermEwpId sendingHeiComment [Optional] eqfLevelStudiedAtDeparture agreementIscedFCode nomineeIscedFCode sendingHeiId sendingOunitId [Optional] receivingOunitId [Optional] sendingIaId [Optional] receivingIaId [Optional]	Serviços Centrais	candidato, NM		Submeter Dados e Learning Ag	NM	Fornecimento de Dados e LA
						Dados da Nomeação			candidato, NM				

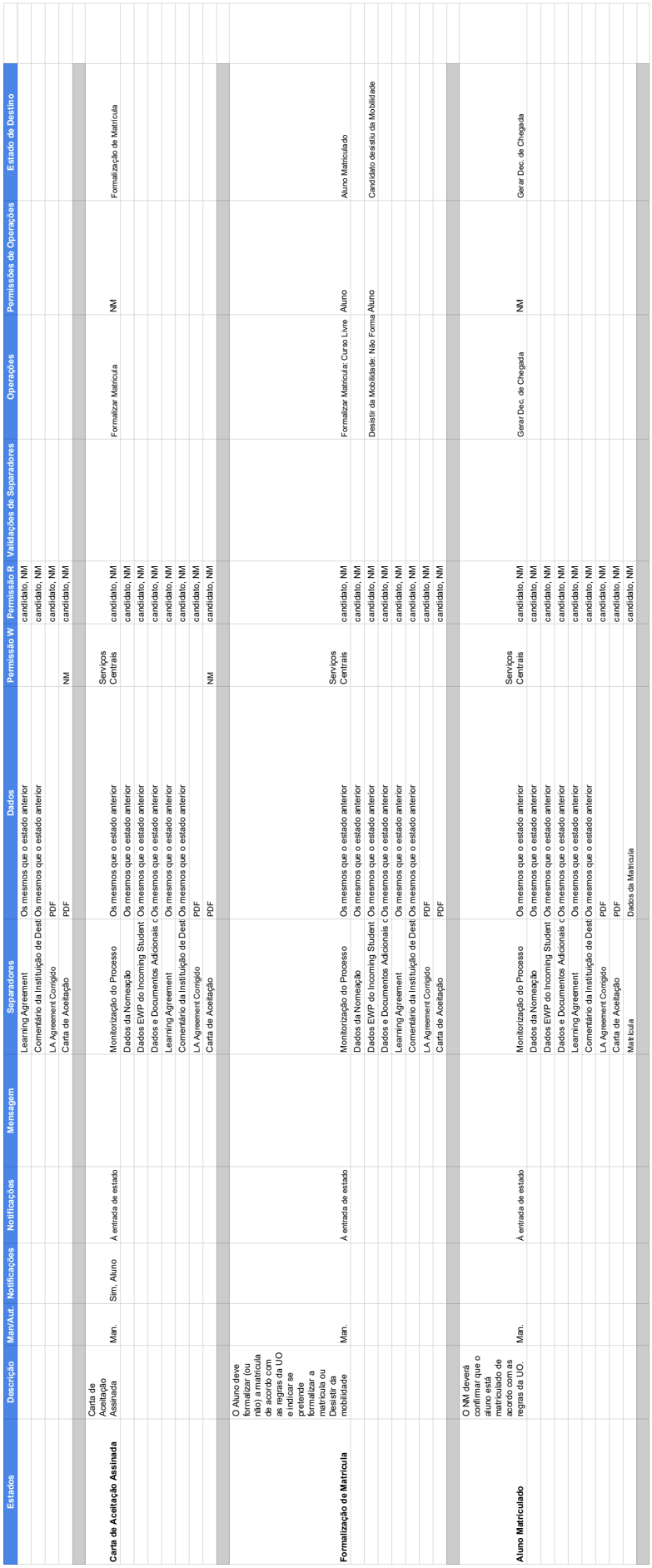
Figura D.6: Excerto da Documentação do *Workflow* Fenix - Mobilidade *Incoming* - 1.

Estados	Descrição	Man/Aut.	Notificações	Notificações	Mensagem	Separadores	Dados	Permissão W	Permissão R	Validações de Separadores	Operações	Permissões de Operações	Estado de Destino
							globalId gender birthDate nationality email givenNames						
						Dados do Incoming Student	familyName		candidato, NM				
						Comentário da Instituição de Dest	receivingHeiComment [Optional]	NM	candidato, NM				
Fornecimento de Dados e LA	Candidato fornece dados solicitados e carrega LA	Man	Sim, Aluno	À entrada de estado		Monitorização do Processo	Workflow nominationType omobilityId ActivityAttributes ActivityType MobilityStatus NominationStatus receivingAcademicYearId sendingAcademicTermEwpId sendingHeiComment [Optional] eqfLevelStudiedAtDeparture agreementIscedFCode nomineeIscedFCode sendingHeiId receivingHeiId sendingOunitId [Optional] receivingOunitId [Optional] sendingIiaId [Optional] receivingIiaId [Optional]	Serviços Centrais	candidato, NM		Submeter Dados e Learnig Agreement	candidato, NM	Validação de Dados e LA Learning Agreement
						Dados da Nomeação	globalId gender birthDate nationality email givenNames familyName		candidato, NM				
						Dados EWP do Incoming Student	• Tipo Documento Identificação* • Número Documento Identificação*						
						Dados e Documentos Adicionais c	• Transcrição de notas emitida e assinada pela IES de origem; • Comprovativo de matrícula, aplicável aos estudantes visitantes e investigadores de pós-doutoramento; • Cartão Europeu de Seguro de Doença, ou documento comprovativo de seguro de saúde privado, ou comprovativo de solicitação do mesmo; • Comprovativo de passaporte válido, caso; • Formulário de reconhecimento de QECR de Língua Portuguesa, assinado pelo gabinete responsável pela nomeação do aluno, ou pelo departamento de português da IES de origem. • Documento institucional comprovativo do nível de língua concluído de acordo com o QECR, emitido por entidade competente, para cada língua que o aluno pretenda frequentar na Faculdade de Letras; • Outros documentos que o aluno considere relevantes.	candidato, NM	candidato, NM				
						Learning Agreement	PDF		candidato, NM	Validar se tem PDF carregado			
						Comentário da Instituição de Dest	receivingHeiComment [Optional]		candidato, NM				
Validação de Dados e LA Learning Agreement	NM valida os dados submetidos pelo candidato e decide em conformidade.	Man	Sim, Aluno	À entrada de estado		Monitorização do Processo	Workflow nominationType omobilityId ActivityAttributes ActivityType MobilityStatus NominationStatus receivingAcademicYearId sendingAcademicTermEwpId sendingHeiComment [Optional] eqfLevelStudiedAtDeparture agreementIscedFCode nomineeIscedFCode sendingHeiId receivingHeiId sendingOunitId [Optional] receivingOunitId [Optional] sendingIiaId [Optional] receivingIiaId [Optional]	Serviços Centrais	candidato, NM		Solicitar Correção de Dados	NM	Fornecimento de Dados e LA
						Dados da Nomeação					Aprovar LA	NM	LA Aprovado

Figura D.7: Excerto da Documentação do *Workflow* Fenix - Mobilidade *Incoming* - 2.

Estados	Descrição	Man/Aut.	Notificações	Notificações	Mensagem	Separadores	Dados	Permissão W	Permissão R	Validações de Separadores	Operações	Permissões de Operações	Estado de Destino
							globalId gender birthDate nationality email givenNames familyName						
						Dados EWP do Incoming Student	• Tipo Documento Identificação* • Número Documento Identificação* • Transcrição de notas emitida e assinada pela IES de origem; • Comprovativo de matrícula, aplicável aos estudantes visitantes e investigadores de pós-doutoramento; • Cartão Europeu de Seguro de Doença, ou documento comprovativo de seguro de saúde privado, ou comprovativo de solicitação do mesmo; • Fotografia tipo passaporte (4x3), a cores; • Formulário de reconhecimento de QECR de Língua Portuguesa, assinado pelo gabinete responsável pela comida do aluno, pelo departamento de português da IES de origem; • Documento institucional comprovativo do nível de língua concluído de acordo com o QECR, emitido por entidade competente, para cada língua que o aluno pretenda frequentar na Faculdade de Letras;				Rejeitar LA - Solicitar Correções	NM	LA Rejeitado: Em Correção
						Dados e Documentos Adicionais c	• Outros documentos que o aluno considere relevantes.				Rejeitar LA e Candidatura	NM	Candidatura Rejeitada
						Learning Agreement	PDF						
						Comentário da Instituição de Dest	receivingHeiComment [Optional]						
Candidatura Rejeitada	NM rejeitou a Candidatura	FINAL	Sim, Aluno	À entrada de estado		Monitorização do Processo	Os mesmos que o estado anterior	Serviços Centrais	candidato, NM				
						Dados da Nomeação	Os mesmos que o estado anterior						
						Dados EWP do Incoming Student	Os mesmos que o estado anterior						
						Dados e Documentos Adicionais c	Os mesmos que o estado anterior						
						Learning Agreement	Os mesmos que o estado anterior						
						Comentário da Instituição de Dest	Os mesmos que o estado anterior						
LA Aprovado	NM Aprovou o LA	Man.	Sim, Aluno	À entrada de estado		Monitorização do Processo	Os mesmos que o estado anterior	Serviços Centrais	candidato, NM		Gerar Carta de Aceitação	NM	Gerar Carta de Aceitação
						Dados da Nomeação	Os mesmos que o estado anterior		candidato, NM				
						Dados EWP do Incoming Student	Os mesmos que o estado anterior		candidato, NM				
						Dados e Documentos Adicionais c	Os mesmos que o estado anterior		candidato, NM				
						Learning Agreement	Os mesmos que o estado anterior		candidato, NM				
						Comentário da Instituição de Dest	Os mesmos que o estado anterior		candidato, NM				
LA Rejeitado: Em Correção	NM rejeitou o LA	Man.	Sim, Aluno	À entrada de estado		Monitorização do Processo	Os mesmos que o estado anterior	Serviços Centrais	candidato, NM		Desistir da Mobilidade	candidato, NM	Candidato Desistiu da Mobilidade
						Dados da Nomeação	Os mesmos que o estado anterior		candidato, NM				
						Dados EWP do Incoming Student	Os mesmos que o estado anterior		candidato, NM		Submeter LA Corrigido	candidato, NM	Validação de Dados e Learning Agreement
						Dados e Documentos Adicionais c	Os mesmos que o estado anterior		candidato, NM				
						Learning Agreement	Os mesmos que o estado anterior		candidato, NM				
						Comentário da Instituição de Dest	Os mesmos que o estado anterior		candidato, NM				
						LA Agreement Corrigido	PDF		candidato, NM	Validar a presença do pdf			
Candidato Desistiu da Mobilidade	Cand. desistiu	FINAL				Monitorização do Processo	Os mesmos que o estado anterior	Serviços Centrais	candidato, NM				
						Dados da Nomeação	Os mesmos que o estado anterior		candidato, NM				
						Dados EWP do Incoming Student	Os mesmos que o estado anterior		candidato, NM				
						Dados e Documentos Adicionais c	Os mesmos que o estado anterior		candidato, NM				
						Learning Agreement	Os mesmos que o estado anterior		candidato, NM				
						Comentário da Instituição de Dest	Os mesmos que o estado anterior		candidato, NM				
						LA Agreement Corrigido	PDF		candidato, NM				
Gerar Carta de Aceitação	Criação auto do Documento	Aut.											Carta de Aceitação Gerada
Carta de Aceitação Gerada	Carta de Aceitação Gerada	Man.	Sim, Aluno	À entrada de estado		Monitorização do Processo	Os mesmos que o estado anterior	Serviços Centrais	candidato, NM		Assinar Carta de Aceitação	NM	Carta de Aceitação Assinada
						Dados da Nomeação	Os mesmos que o estado anterior		candidato, NM				
						Dados EWP do Incoming Student	Os mesmos que o estado anterior		candidato, NM				
						Dados e Documentos Adicionais c	Os mesmos que o estado anterior		candidato, NM				

Figura D.8: Excerto da Documentação do *Workflow* Fenix - Mobilidade *Incoming* - 3.



Estados	Descrição	Man/Aut.	Notificações	Notificações	Mensagem	Separadores	Dados	Permissão W	Permissão R	Validações de Separadores	Operações	Permissões de Operações	Estado de Destino
						Learning Agreement	Os mesmos que o estado anterior		candidato, NM				
						Comentário da Instituição de Dest	Os mesmos que o estado anterior		candidato, NM				
						LA Agreement Corrigido	PDF		candidato, NM				
						Carta de Aceitação	PDF	NM	candidato, NM				
Carta de Aceitação Assinada	Carta de Aceitação Assinada	Man.	Sim, Aluno	À entrada de estado		Monitorização do Processo	Os mesmos que o estado anterior	Serviços Centrais	candidato, NM		Formalizar Matricula	NM	Formalização de Matrícula
						Dados da Nomeação	Os mesmos que o estado anterior		candidato, NM				
						Dados EWP do Incoming Student	Os mesmos que o estado anterior		candidato, NM				
						Dados e Documentos Adicionais c	Os mesmos que o estado anterior		candidato, NM				
						Learning Agreement	Os mesmos que o estado anterior		candidato, NM				
						Comentário da Instituição de Dest	Os mesmos que o estado anterior		candidato, NM				
						LA Agreement Corrigido	PDF		candidato, NM				
						Carta de Aceitação	PDF	NM	candidato, NM				
Formalização de Matrícula	O Aluno deve formalizar (ou não) a matrícula de acordo com as regras da UO e indicar se pretende formalizar a matrícula ou desistir da mobilidade	Man.		À entrada de estado		Monitorização do Processo	Os mesmos que o estado anterior	Serviços Centrais	candidato, NM		Formalizar Matricula: Curso Livre	Aluno	Aluno Matriculado
						Dados da Nomeação	Os mesmos que o estado anterior		candidato, NM				
						Dados EWP do Incoming Student	Os mesmos que o estado anterior		candidato, NM		Desistir da Mobilidade: Não Forma	Aluno	Candidato desistiu da Mobilidade
						Dados e Documentos Adicionais c	Os mesmos que o estado anterior		candidato, NM				
						Learning Agreement	Os mesmos que o estado anterior		candidato, NM				
						Comentário da Instituição de Dest	Os mesmos que o estado anterior		candidato, NM				
						LA Agreement Corrigido	PDF		candidato, NM				
						Carta de Aceitação	PDF		candidato, NM				
Aluno Matriculado	O NM deverá confirmar que o aluno está matriculado de acordo com as regras da UO.	Man.		À entrada de estado		Monitorização do Processo	Os mesmos que o estado anterior	Serviços Centrais	candidato, NM		Gerar Dec. de Chegada	NM	Gerar Dec. de Chegada
						Dados da Nomeação	Os mesmos que o estado anterior		candidato, NM				
						Dados EWP do Incoming Student	Os mesmos que o estado anterior		candidato, NM				
						Dados e Documentos Adicionais c	Os mesmos que o estado anterior		candidato, NM				
						Learning Agreement	Os mesmos que o estado anterior		candidato, NM				
						Comentário da Instituição de Dest	Os mesmos que o estado anterior		candidato, NM				
						LA Agreement Corrigido	PDF		candidato, NM				
						Carta de Aceitação	PDF		candidato, NM				
						Matrícula	Dados da Matrícula		candidato, NM				

Figura D.9: Excerto da Documentação do *Workflow* Fenix - Mobilidade *Incoming* - 4.

Apêndice E

Modelação de Domínio

E.1 Domínio do Módulo de Mobilidade Fenix

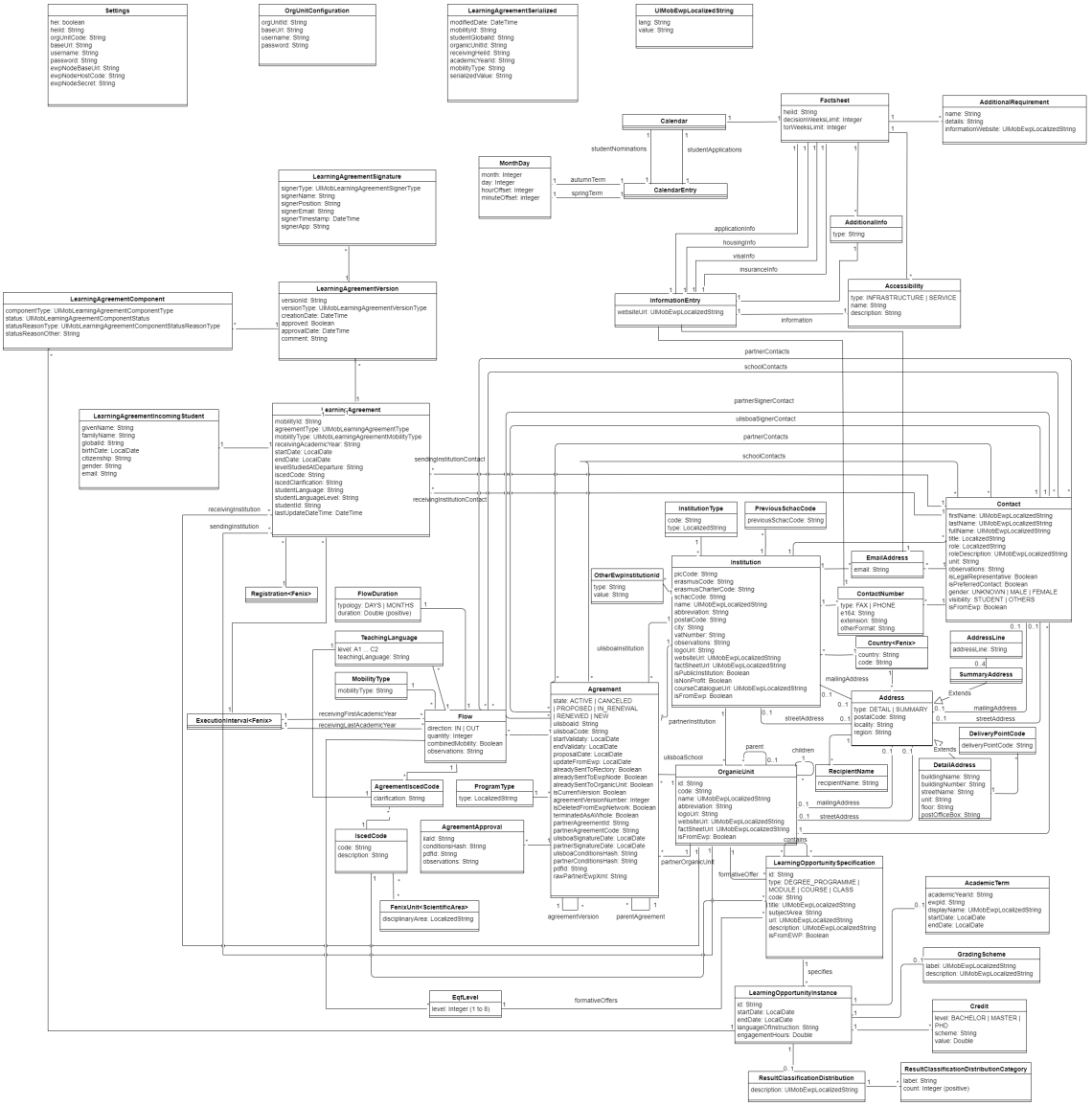


Figura E.1: Modelo de Domínio do Módulo de Mobilidade Fenix - fornecido pelo NDS.

E.2 Mapeamento de Entidades e Atributos

Url: https://github.com/erasmus-without-paper/ewp-specs-api-imobilities/blob/stable-v2/endpoints/get-response.xsd
Ref: **ewp-specs-api-imobilities/endpoints**
Date: **/get-response.xsd**
Master: **4/9/2024**
Estrutura: ReceivingHEI

```
<xs:element name="imobilities-get-response">
  <xs:group ref="SequenceOfIncomingMobilities" minOccurs="1" maxOccurs="1">
    <xs:group name="SequenceOfIncomingMobilities">
      <xs:group ref="SingleIncomingMobilityObject" minOccurs="0" maxOccurs="unbounded">
        <xs:group name="SingleIncomingMobilityObject">
          <xs:element ref="student-mobility-for-studies"/>
```

Entidade	Especificação	Tipo de Dados	Obrigatório?	Implementação - Entidade	Implementação	Tipo de Dados Implementação	Exemplo Especificação	Notas
-	omobility-id	AsciiPrintableIdentifier	Sim	UIMobilStudentNomination		String	<omobility-id>d331d178-4430-3d6f-8fec-7be72f02502</omobility-id>	Identifier of the mobility (as assigned by the sending HEI).
-	status	**NominationStatus**	Sim	UIMobilNominationStatus		String	<status>rejected</status>	Só há 3 possíveis e são definidos pelo Master.
NominationStatus	-	String	Sim	UIMobilNominationStatus	public enum UIMobilNominationStatus	enum	-	This should be na Enum PENDING | VERIFIED | REJECTED
NominationStatus	pending	Value	-	PENDING	pending	String	<xs:enumeration value="pending"/>	The nomination proposal has not yet been accepted nor rejected by the receiving HEI
NominationStatus	verified	Value	-	VERIFIED	verified	String	<xs:enumeration value="verified"/>	The nomination was recognized by the receiving HEI as formally correct and complying with cooperation conditions. Now it's time for setting formalities, sending student's documents, creating the first Learning Agreement, etc.
NominationStatus	rejected	Value	-	REJECTED	rejected	String	<xs:enumeration value="rejected"/>	The nomination has been rejected by the receiving HEI.
-	comment	xs:string	Não	UIMobilStudentNomination	receivingHeiComment	String	<comment>Student leaved after one week.</comment>	These comments MUST be visible only to the IRO members, not the students.
-	actual-arrival-date	xs:date	Não	UIMobilStudentNomination	actualArrivalDate	LocalDate	<actual-arrival-date>2018-06-03</actual-arrival-date>	Actual means "when the student has actually arrived"
-	actual-departure-date	xs:date	Não	UIMobilStudentNomination	actualDepartureDate	LocalDate	<actual-departure-date>2018-06-10</actual-departure-date>	Actual end date of the mobility. "Actual" means "when the student has actually left". When the student is supposed to arrive | This date is provided by the sending HEI before the mobility.
-	planned-arrival-date	xs:date	Não	UIMobilStudentNomination	plannedArrivalDate	LocalDate	<planned-arrival-date>2010-02-01</planned-arrival-date>	
-	planned-departure-date	xs:date	Não	UIMobilStudentNomination	plannedDepartureDate	LocalDate	<planned-departure-date>2010-06-20</planned-departure-date>	When the student is supposed to leave | This date is provided by the sending HEI before the mobility.
-	planned-virtual-start-date	xs:date	Não	UIMobilStudentNomination	plannedVirtualStartDate	LocalDate		
-	planned-virtual-end-date	xs:date	Não	UIMobilStudentNomination	plannedVirtualEndDate	LocalDate		
-	actual-virtual-start-date	xs:date	Não	UIMobilStudentNomination	actualVirtualStartDate	LocalDate		
-	actual-virtual-end-date	xs:date	Não	UIMobilStudentNomination	actualVirtualEndDate	LocalDate		

Figura E.2: Representação Mapeamento de Entidades e Atributos - API EWP *Incoming* V2.

Url: ewp-specs-api-omobilities/endpoints/get-response.xsd at stable-v3 · erasmus-without-paper/ewp-specs-api-omobilities · GitHub
Ref: **ewp-specs-api-omobilities/endpoints/get-response.xsd**
Date: **4/9/2024**
Master: SendingHEI

Estrutura:

```
<xs:element name="omobilities-get-response">
<xs:group ref="SequenceOfMobilities" minOccurs="1" maxOccurs="1">
<xs:group name="SequenceOfMobilities">
<xs:group ref="SingleMobilityObject" minOccurs="0" maxOccurs="unbounded">
  <xs:group name="SingleMobilityObject">
  <xs:element name="student-mobility">
```

This describes a single Student Mobility: Every mobility has its unique sending-hei, receiving-hei and nominee, which **MUST NOT** change after the mobility is created.

Entidade	Atributo	Tipo de Dados	Obrigatório?	Implementação - Entidade	Implementação	Tipo de Dados Implementação	Exemplo Especificação	Exemplo Implementação	Notas
-	omobility-id	AsciiPrintableIdentifier	Sim	UlMobInStudentNomination	omobilityId	String	<omobility-id>d331d178-4430-3dbf-8fec-7be72f022502</omobility-id>		Identifier of the mobility (as assigned by the sending HEI). Immutable.
sending-hei	-	**complexType**	Sim	Institution	-		-		This describes the sending HEI - the home institution of the student being sent.
sending-hei	hei-id	String	Sim	?	?	String	<hei-id>uio.no</hei-id>		SCHAC ID of the institution. Immutable.
sending-hei	ounit-id	AsciiPrintableIdentifier	Não	?	?	String	<ounit-id>140</ounit-id>		If given, then it refers to the unit within the sending HEI's organizational structure
sending-hei	iia-id	AsciiPrintableIdentifier	Não	?	?	String	<iia-id>dd87538e-49e3-44ae-844e-55ad1b819ce9</iia-id>		Optional ID of the Institutional Agreement based on which the student is being nominated/sent.
receiving-hei	-	**complexType**	Sim	Institution	-		-		This describes the receiving HEI - the home institution of the student being sent.
receiving-hei	hei-id	String	Sim	?	?	String	<hei-id>uw.edu.pl</hei-id>		SCHAC ID of the institution. Immutable.
receiving-hei	ounit-id	AsciiPrintableIdentifier	Não	?	?	String			If given, then it refers to the unit within the receiving HEI's organizational structure
receiving-hei	iia-id	AsciiPrintableIdentifier	Não	?	?	String	<iia-id>549117543</iia-id> <!-- We recommend UUIDs though. -->		Optional ID of the Institutional Agreement based on which the student is being nominated/sent.
-	sending-academic-term-ewp-id	EwpAcademicTermId	Sim choice	UlMobInEwpAcademicTermId	-		<sending-academic-term-ewp-id>2009/2010-2/2</sending-academic-term-ewp-id>		<xs:choice minOccurs="1" maxOccurs="1"> | The sending HEI is REQUIRED to either identify this term, or to clearly state that the sending mobility period is "non-standard", and is not contained by any single academic term.
EwpAcademicTermId		**complexType**		UlMobInEwpAcademicTermId	-	-			
EwpAcademicTermId				UlMobInEwpAcademicTermId	sendingAcademicTermEwpId	String			
-	non-standard-mobility-period	ewp:Empty	Sim choice	UlMobInNonStandardMobilityPeriod	-	?			<xs:choice minOccurs="1" maxOccurs="1"> |
ewp:Empty		**complexType**		UlMobInNonStandardMobilityPeriod	-	-			
ewp:Empty				UlMobInNonStandardMobilityPeriod	startDate	LocalDate			
ewp:Empty				UlMobInNonStandardMobilityPeriod	endDate	LocalDate			
-	receiving-academic-year-id	trm:AcademicYearId	Sim	UlMobInAcademicYearId			<receiving-academic-year-id>2009/2010</receiving-academic-year-id>		Academic year during which this mobility takes place e.g. "2010/2011" vs. "2010/2010" or "2011/2011"
trm:AcademicYearId		**complexType**		UlMobInAcademicYearId	-	-			
trm:AcademicYearId				UlMobInAcademicYearId	receivingAcademicYearId	String			
student	-	**complexType**	Sim	UlMobInStudent					
student	given-names	StringWithOptionalLang	Sim 1..*	UlMobInStudent	givenNames	UlMobEwpLocalizedString	<given-names>Ivan Petrovich</given-names> <given-names xml:lang="ru-Cyrl">Иван Петрович</given		The given names of the person. | It MAY be specified in multiple languages
student	family-name	StringWithOptionalLang	Sim 1..*	UlMobInStudent	familyName	UlMobEwpLocalizedString	<family-name>Sidorov</family-name> <family-name xml:lang="ru-Cyrl">Сидоров</family-name>		The given names of the person. | It MAY be specified in multiple languages
student	global-id	String	Sim	UlMobInStudent	globalId	String	<global-id>urn:schac:personalUniqueCode:int:esi:uio.no:1234567890</global-id>		Global student identifier. Should follow the specification of the European Student Identifier
student	birth-date	Date	Sim	UlMobInStudent	birthDate	LocalDate	<birth-date>1985-05-05</birth-date>		
student	nationality	ewp:CountryCode	Sim	UlMobInStudent	nationality	String	<citizenship>NO</citizenship>		Code of the country to which the person belongs administratively and that issues the ID card and/or passport
student	gender	ewp:Gender	Sim	UlMobInStudent	gender	integer	<gender>1</gender>		Optional code of the gender
student	email	ewp:Email	Sim	UlMobInStudent	email	String	<email>ivan@example.com</email>		Email address at which student should be contacted.
-	status	MobilityStatus	Sim	UlMobInMobilityStatus			<status>live</status>		Current status of the mobility
MobilityStatus	-	Enum	Sim	UlMobInMobilityStatus					This should be na Enum NOMINATION | LIVE | RECOGNIZED | CANCELLED
MobilityStatus	nomination	value	-	NOMINATION	nomination				The sending HEI has nominated the student for mobility. The proposal has not yet been accepted nor rejected by the receiving HEI. | This is the default status with which mobility entities are first created.
MobilityStatus	live	value	-	LIVE	live				The nomination has been accepted by the receiving HEI, and all initial formalities have been settled (i.e. a first LA version has been signed). This status doesn't usually change throughout the mobility.
MobilityStatus	recognized	value	-	RECOGNIZED	recognized				The student has returned from the mobility and all achievements have been recognized as indicated on the `component-recognized` list. At this point, the mobility SHOULD become read-only. The latest revision of it SHOULD be approved by all parties (no subsequent draft revisions should exist).
MobilityStatus	cancelled	value	-	CANCELLED	cancelled				The nomination has been cancelled (either by the student, or by one of the partner HEIs.
-	activity-type	MobilityActivityType	Sim	UlMobInActivityType					Student mobility activity type.
MobilityActivityType	-	Enum	Sim	UlMobInActivityType					This should be na Enum STUDENT_STUDIES | STUDENT_TRAINEESHIPS
MobilityActivityType	student-studies	value	-	STUDENT_STUDIES	studentStudies		<activity-type>student-studies</activity-type>		Student Mobility for Studies (SMS).
~~MobilityActivityType~~	~~student-traineeships~~	~~value~~	~~-~~	~~STUDENT_TRAINEESHIPS~~	~~studentTraineeships~~				~~Student Mobility for Traineeships (SMT).~~
-	activity-attributes	MobilityActivityAttributes	Sim	UlMobInStudentNomination	1-1	UlMobInActivityAttributes			Student mobility activity attributes, such as the length of the activity, study level, or the nature of the educational components.
MobilityActivityAttributes	-	Enum	Sim	1-*	UlMobInActivityAttributes	-			This should be na Enum LONG_TERM | SHORT_TERM_BLENDED | SHORT_TERM_DOCTORAL
MobilityActivityAttributes	long-term	value	-	LONG_TERM	longTerm	String	<activity-attributes>long-term</activity-attributes>		Long-term physical mobility (with an optional virtual component).
MobilityActivityAttributes	short-term-blended	value	-	SHORT-TERM_BLENDED	shortTermBlended	String			Short term blended mobility (with a compulsory virtual component).
MobilityActivityAttributes	short-term-doctoral	value	-	SHORT-TERM_DOCTORAL	shortTermDoctoral	String			Short term doctoral mobility (with an optional virtual component).
-	eqf-level-studied-at-departure	ewp:EqfLevel	Sim	UlMobInStudentNomination	eqfLevelStudiedAtNomination	EqfLevel	<eqf-level-studied-at-departure>8</eqf-level-studied-at-departure>		This is similar to the `eqf-level-studied-at-nomination` element, but this time we are interested in the EQF level the student **will have** when the mobility actually begins
-	agreement-isced-f-code	Isced	Sim		agreementIscedFCode	Isced			
-	nominee-isced-f-code	Isced	Sim		nomineeIscedFCode	Isced	<nominee-isced-f-code>1234</nominee-isced-f-code>		This field contains the ISCED-F field of education code that is closest to the subject this programme
Isced	-	String	Sim				value="[0-9]{4}"/>		
Isced	nominee-isced-f-code	String	-						
additional-requirement	-	**complexType**	Não 0..*	UlMobInStudentNomination	*	UlMobInAdditionalRequirement			Requirements listed in the Factsheet API of the receiving HEI, that SHOULD be satisfied by the incoming students by sending a file or a URL. The name and type (file or URL) MUST be exactly as documented in the Factsheet API.
additional-requirement	name	String	Sim	UlMobInAdditionalRequirement	name	String	<name>Transcript of Records</name>		Name of requirement MUST be exactly as documented in the Factsheet API
additional-requirement	file	String	Sim choice	UlMobInAdditionalRequirement	file	String	<file>2f521e8a-d9fd-4f49-9941-df9318bf190f</file>		Identifier of the file that MUST be accessible to the receiving HEI through the EWP File API.
additional-requirement	url	ewp:HTTPS	Sim choice	UlMobInAdditionalRequirement	url	String	<url>https://www.youtube.com/watch?v=gdg34IFo</url>		URL of the file that meets all restrictions mentioned in Factsheet API

Figura E.3: Representação do Mapeamento de Entidades e Atributos - API EWP *Outgoing* V3.

Apêndice F

Carta de Aceitação

F.1 Exemplo de proposta para *Carta de Aceitação*

Template Proposal - Please adapt

LISBOA

UNIVERSIDADE DE LISBOA

Acceptance Letter

The Study Abroad from the International Office of the School of Arts and Humanities of ULisboa hereby confirms that João Doe, student from the ulisboa.pt, has been accepted as an Erasmus+ student for the period of 2020/2021-1/1.

The Academic Calendar for 2024/2025 is as follows:

Semester 1 (S1)

- Welcome Session - 12th of September ,2024
- Registration in Curricular Units - 13th of September, 2024
- Semester Period - 14th of September to 10th of January, 2025
- Changes to Curricular Units - 23rd to 26th of September, 2024
- Alternative Assessment Period - 27th to 31st of January, 2025 (only for students who failed courses during the academic period described above with a minimum grade of 7, 8 or 9)

Semester 2 (S2)

- Welcome Session - 6th of February, 2025
- Registration in Curricular Units - 7th of February, 2025
- Semester Period - 10th of January to 10th of May, 2025
- Changes to Curricular Units - 17th to 20th of February, 2025
- Alternative Assessment Period - 16th to 20th of May, 2025 (only for students who failed the curricular units with a grade of 7, 8 or 9)

Erasmus+ Students acording to program regulations are exempt from payment of the tuition fee due for the duraiton of their mobility period.

All students are required to pay a registration fee of €20,00, that includes student card, school ensurance and administrative costs.

Class attendance is mandatory.

-●-

O Núcleo de Cooperação Internacional da Divisão de Relações Externas e Internacionais da Faculdade de Letras da Universidade de Lisboa confirma que o/a aluno/a João Doe, proveniente de ulisboa.pt, foi aceite como estudante Erasmus+ durante o peíodo de 2020/2021-1/1.

O calendário académico para este ano lectivo é:

Semestre 1 (S1)

- Sessão de Boas-vindas - 12 de Setembro de 2024
- Inscrição nas Unidades Curriculares - 12 e 13 de Setembro de 2024
- Período do Semestre - 14 de Setembro a 10 de Dezembro de 2024
- Alterações das Unidades Curriculares - 23 a 26 de Setembro de 2024
- Época de Avaliação Final Alternativa - 27 a 31 de Janeiro de 2025 (apenas para alunos que reprovaram às unidades curriculares com nota entre 7 e 9)

Semestre 2 (S2)

- Sessão de Boas-vindas - 23 de Janeiro de 2025
- Inscrição nas Unidades Curriculares - 23 e 24 de Janeiro de 2025
- Período do Semestre - 10 de Fevereiro a 30 de Maio de 2025
- Alterações das Unidades Curriculares - 17 a 20 de Fevereiro de 2025
- Época de Avaliação Final Alternativa - 16 a 20 de Junho de 2025 (apenas para alunos que reprovaram às unidades curriculares com nota entre 7 e 9)

O/a estudante Erasmus+ de acordo com as regras do programa está isento/a do pagamento de propinas durante o período oficial da sua mobilidade

Todos os alunos estão sujeitos ao pagamento de uma taxa de inscrição no valor de €20,00, que inlcui o cartão de estudante, o seguro escolar e gestão administrativa.

A frequência das aulas é obrigatória.

Link de Publicação / Publication link:
http://localhost:8080/fenix/documents/?id=webDocs5bba37ff-a601-4c84-9f49-96ee966e5d2c