Trabalho Prático

Relatório Meta 2

Instituto Superior de Engenharia

Politécnico de Coimbra

Engenharia Informática

Programação Avançada

2021/22

João Baptista - 2020131684

Pedro Sequeira - 2020132079

Decisões na implementação

ArrayLists: Para guardar os dados dos ficheiros csv relativos aos docentes, alunos, propostas e candidaturas foram criados ArrayLists para cada um.

Estados fechados: De modo a fechar estados como referido no enunciado foram criados novos estados semelhantes aos estados das fases normais, no entanto apenas com as opções de visualização, já que não é permitido alterar informação num estado fechado.

Classes

Aluno: Representa os alunos, têm variáveis relativas ao número de estudante (nr_estudante), nome, email, curso, ramo, classificação e possibilidade de aceder a estágio além de projeto (EeP). Contém o construtor de alunos e os métodos get para estas variáveis, assim como um toString que imprime os seus valores.

App: Esta classe é responsável pela leitura de dados a partir de ficheiros csv, estes valores são inseridos nos respetivos ArrayList para alunos, docentes, propostas e candidaturas. Contém ainda vários métodos de verificação, para docentes, repetições de candidatura e propostas de alunos.

AppManager: App que funciona como “facade” da classe App, esta classe limita-se a redirecionar as chamadas das funções para a class App.

Docente: Representa os docentes, tem variáveis relativas ao nome e email do docente, assim como o construtor de docentes, métodos get para as variáveis e um toString que as imprime.

Candidatura: Tem um ArrayList com o número de aluno e métodos get para o mesmo e para o ArrayList, tem ainda um toString que imprime ambos.

Proposta: Esta classe contém variáveis para o código de identificação de proposta (id), número do aluno e o título da proposta; construtores para propostas, um para propostas atribuídas e outro para não atribuídas, e os métodos get para o id, número de aluno e título da proposta. Tem ainda um toString que imprime estes valores.

Autoproposta: Classe que estende da classe Proposta, tem o construtor para autopropostas, com o número de aluno, id e título da proposta. Contém ainda um toString que adiciona "Autoproposta:" ao toString da Proposta e um método get para o Ramo.

Estagio: Classe que estende da classe Proposta, tem variáveis ramo, referente ao ramo da proposta e id_acolhimento. Contém dois construtores de estágios, pois o estágio pode ter ou não a indicação do aluno a que deve ser atribuído. Tem métodos get para o ramo e id de acolhimento e ainda um toString que adiciona "Estagio:" ao toString da Proposta.

Projeto: Tem variáveis relativas ao ramo e email do docente que propôs o projeto. Contém dois construtores de projetos, pois um projeto pode ter a indicação do aluno ao qual deve ser atribuído. Tem métodos get para o ramo e email do docente e ainda um toString que adiciona "Projeto:" ao toString da Proposta e indica o ramo e email do docente que sugeriu o projeto.

adicionaAtribuicaoProposta: Classe responsável por adicionar a a ttribuição de uma proposta.

AppContext: Classe que contém todos os métodos da máquina de estados, contém ainda a serialização e deserialização

AppState: Enum que contém todos os estados da máquina de estados e uma factory que retorna o estado em que o programa se encontra.

AppStateAdapter: Classe que contém o Override de todos os métodos usados pela máquina de estados.

AtribuicaoManualProp: Classe que permite a atribuição manual de uma proposta a um aluno.

EsperaEscolha: Classe que espera a escolha do utilizador da atribuição de um orientador a um aluno.

Fase1State: Classe correspondente à fase 1, tem o método getState que devolve o estado atual (FASE1) e métodos que remetem para outros estados, sendo estes a gestão de alunos, gestão de docentes, gestão de propostas, fase 2 e um método que fecha a fase 1.

Fase1FechadaState: Neste estado não são possíveis alterações de dados, é apenas possível consultar os dados já alterados da fase 1.

Fase2State: Classe correspondente à fase 2, tem o método getState que devolve o estado atual (FASE2), métodos que remetem para outros estados, fase 3 e fase 1. Tem ainda métodos que listam alunos, propostas e candidaturas.

Fase2FechadaState: Neste estado não são possíveis alterações de dados, é apenas possível consultar os dados já alterados da fase 2.

Fase3State: Classe correspondente à fase 3, tem o método getState que devolve o estado atual (FASE3), métodos que remetem para outros estados, fase 2 e fase 4. Tem ainda métodos que exportam dados, listam alunos, propostas e candidaturas. Permite a remoção e atribuição manual de propostas.

Fase3FechadaState: Neste estado não são possíveis alterações de dados, é apenas possível consultar os dados já alterados da fase 3.

Fase4State: Classe correspondente à fase 4, tem o método getState que devolve o estado atual (FASE4), métodos que remetem para outros estados, fase 3 e fase 5. Tem ainda métodos que exportam dados referentes às fases 4, 5 e gestão de orientadores (GESTAO_ORIENTADORES), listam alunos com e sem orientador e os próprios orientadores.

Fase5State: Classe correspondente à fase 5, tem o método getState que devolve o estado atual (FASE5). Tem ainda métodos que exportam dados referentes às fases 4 e 5, lista alunos com propostas atribuídas, orientadores, alunos sem proposta atribuída com candidatura e lista ainda as propostas atribuídas e não atribuídas.

GestaoAlunosState: Esta classe permite a leitura de alunos do ficheiro csv e listar os mesmos.

GestaoDocentesState: Esta classe permite a leitura de docentes do ficheiro csv e listar os mesmos.

GestaoOrientadoresState: Classe que lista orientadores e os seus respetivos alunos.

GestaoPropostasState: Esta classe permite a leitura de propotas do ficheiro csv e listar os mesmos.

IAppState: Interface com todas as funções.

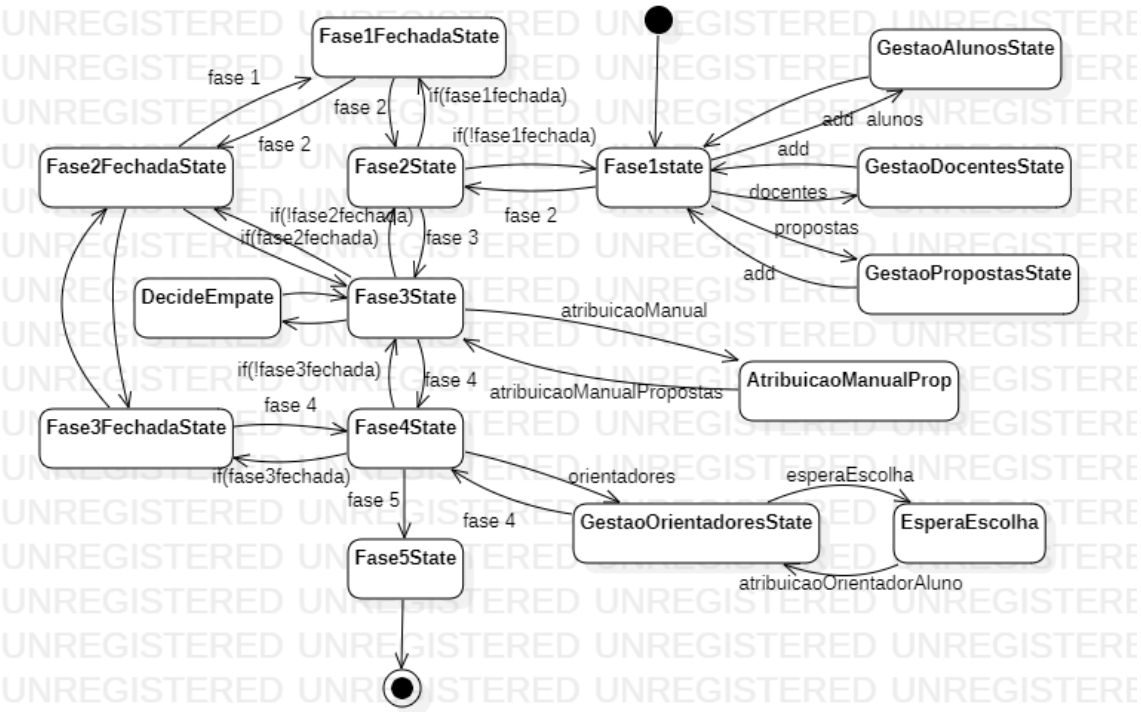
AppUI: Classe com os menus de escolha mostrados ao utilizador em cada fase.

User Interface

De modo a implementar o trabalho em JavaFX foram criadas diversas classes acabadas com UI, estas representam a user interface da sua classe, por exemplo Fase1UI corresponde à user interface da fase 1.

Existem ainda classes como MainJFX, que atua como main para o interface gráfico, a classe RootPane onde são criadas as diversas views para cada classe tratando também os seus updates e a classe ToastMessage que apresenta pop-ups ao utilizador.

Diagrama da máquina de estados



O que não foi cumprido

As operações de Undo e Redo não foram implementadas.