Departamento de Engenharia Informática e de Sistemas

Instituto Superior de Engenharia de Coimbra

Licenciatura em Engenharia Informática

Programação Avançada - 2021/2022

Relatório do Trabalho Prático:

2ª meta

Turma Prática Nº 1

Francisco Simões | a2019133920@isec.pt

Ricardo Ferreira | a2016020798@isec.pt

Índice

1. Descrição sintética acerca das opções e decisões tomadas na implementação

A aplicação apresenta diversas classes, que foram colocadas nos respetivos packages, de acordo com as suas funções específicas. No package Data, encontram-se todos os dados relativos à aplicação.

Por sua vez, no package fsm, existe uma interface *IGestaoEstagioState* que tem os diferentes métodos que vão implementar ações, de acordo com o respetivo estado em que se encontram.

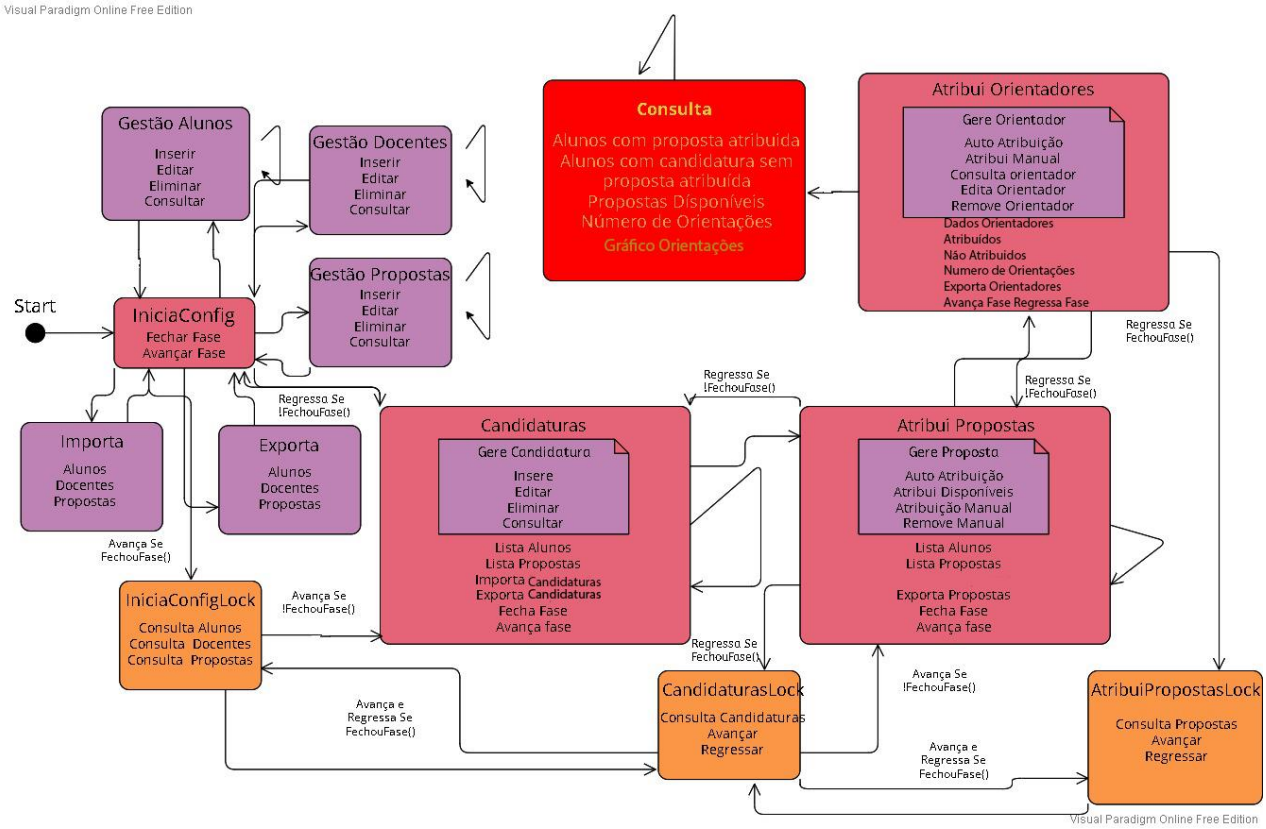
A classe abstrata *GestaoEstagioStateAdapter*, que implementa a interface *IGestaoEstagioState*, tem uma referência para os dados da aplicação, de modo que os diferentes estados consigam aceder a essa informação.

No package *Model*, estão presentes todas as classes relacionadas com a implementação da lógica da app.

Em relação aos estados em si, e de acordo com a máquina de estados que implementámos, foi decidida a criação de 14 estados necessários para a implementação da lógica da app, que vão ser explicados no próximo tópico.

Existe um package *ui*, onde se encontra o package *text*, e por sua vez a classe GestaoEstagioUI, responsável pela interação com o utilizador, apresentando-lhe a informação necessária. A ligação entre ui e estados é realizada pela classe GestaoEstagioContext. Na implementação do "fechar fase" foi criado um estado **FECHADO**, representativo de cada fase, exceto a última, que se limita à consulta de dados.

2. Diagrama da máquina de estados

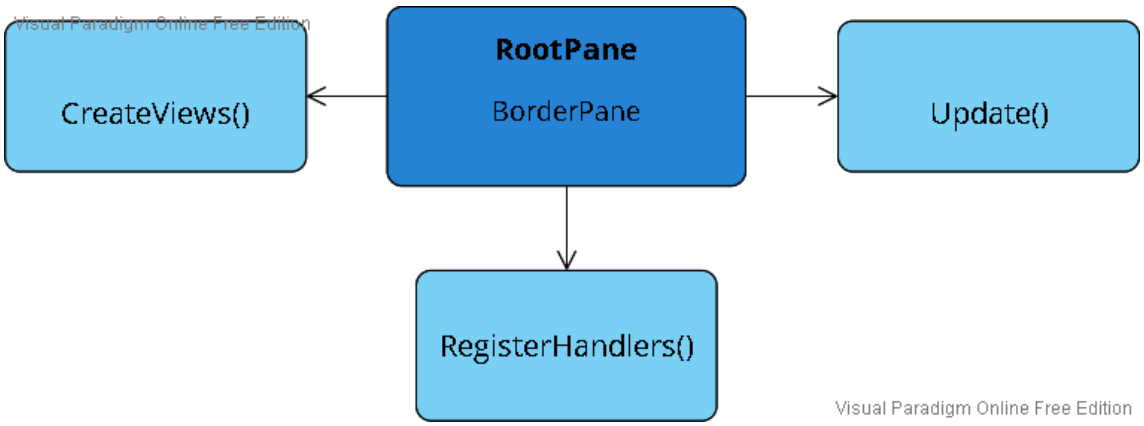


A máquina de estados vai começar por definir o estado atual, como sendo o estado **INICIA_CONFIG**, aquando de uma nova instância. Este vai ser responsável por gerir 3 modos de gestão **(GERE_ALUNOS, GERE_DOCENTES, GERE_PROPOSTAS)** os quais se alternam entre si dependendo da vontade do utilizador, podendo adicionalmente importar, exportar dados, fechar ou avançar de fase.

As próximas fases/estados **(GERE_CANDIDATURAS, ATRIBUICAO_PROPOSTA, ATRIBUICAO_ORIENTADORES)** contêm ações que o utilizador poderá selecionar em "***runtime",*** sendo que cada fase terá um estado ***"lock"*** associado, que representa o fechar fase, sempre que se verifiquem as condições necessárias para tal (**não pode ser reaberta**).

Por último, temos a fase de consulta de dados **CONSULTA** que se dedica à consulta de dados não sendo possível o regresso a qualquer uma das fases anteriores.

3. Diagrama de outros padrões de programação



- ***CreateViews()***: Cria a vista gráfica para o utilizador, de acordo com o contexto observável no momento;

- ***RegisterHandlers()***: Quando o utilizador interage com a interface gráfica, dependendo da ação pretendida é acionado um método do contexto observável para realiza-la;

- ***Update()***: Depois da ação do utilizador é registado o update que altera a propriedade do componente gráfico através do *firePropertyChange*;

4. Descrição das classes utilizadas no programa

Package pt.isec.pa.apoio_poe:

- ***Main***: Contém os objetos correspondentes à máquina de estado e às interfaces que vão interagir com o utilizador. Para além disso, vai dar início à aplicação.

Package pt.isec.pa.apoio_poe.model:

- **GestaoEstagioData:** armazena os dados da app e da sua lógica, para o bom funcionamento de determinada instância.

Package pt.isec.pa.apoio_poe.model.fsm:

- **GestãoEstagioAdapter:** Contém a implementação origem dos métodos do contexto para o contexto observável da aplicação;
- **GestãoEstagioContext:** Contém o incapsolamento dos métodos da data consoante o estado da aplicação;
- **GestãoEstagioManager:** Contém os métodos observáveis do contexto para serem aplicados na Interface gráfica;
- **GestãoEstagioState:** Contém os estados da aplicação e realiza a factory desses estados;
- **IGestãoEstagioState:** Contém os métodos da interface do contexto;
- **Iniciotate:** Contém o método de ponto de partida da aplicação;
- **IniciaConfigState:** Contém um método recebeModo(int option) que, dependendo da opção escolhida no ui, vai redirecionar para o modo pretendido(estado);
- **GereAlunosState:** Composto por métodos necessários para a gestão de alunos;
- **GereDocentesState:** Composto por métodos necessários para a gestão de docentes;
- **GerePropostaState:** Composto por métodos necessários para a gestão de Propostas;
- **ImportaDadosState:** Contém todos os métodos de importação de dados;

- **ExportaDadosState:** Contém todos os métodos para exportação de dados;
- **GereCandidaturaState:** Composto por métodos necessários para a gestão de candidaturas;
- **AtribuicaoPropostaState:** Composto por métodos necessários à atribuição de alunos a propostas que estejam disponíveis;
- **AtribuicaoOrientadorState:** Composto por métodos necessários à atribuição de orientadores a propostas;
- **GereConsultaState:** Composto por métodos de consulta de dados do sistema;
- **IniciaConfigStateLock:** Permite a consulta de dados que foram modificados ou adicionados ao sistema durante a fase **INICIA_CONFIG**;
- **GereCandidaturaStateLock:** Permite a consulta de dados que foram modificados ou adicionados ao sistema durante a fase **GERE_CANDIDATURA**;
- **AtribuicaoPropostaStateLock:** Permite a consulta de dados que foram modificados ou adicionados ao sistema durante a fase **ATRIBUICAO_PROPOSTA**;
- **CommandAdapter:** Contém a implementação origem dos comandos de edição (undo/redo) da aplicação;
- **CommandManager:** Contém o incapsolamento dos comandos do para edição (undo/redo) da aplicação;
- **ICommand:** Interface dos comandos para edição (undo/redo) da aplicação;
- **AtribuiPropostaCommand:** Contém o comando atribui proposta manual;
- **AtribuiProponenteCommand:** Contém o comando atribui proponente manual;
- **RemovePropostaCommand:** Contém o comando remove proposta manual;
- **RemoveProponenteCommand:** Contém o comando remove proponente manual;

Package pt.isec.pa.apoio_poe.model.data:

Aluno: Contém as caraterísticas representativas de um aluno;

Candidaturas: Contém o número de Aluno e as opções que o mesmo tomou aquando do registo da sua candidatura

Docente: Composto pelo seu email e nome;

Package projetodata:

- **Propostas:** Contém a informação sobre as propostas existentes que podem ser de vários tipos:
  - **Estágio,Projeto,ProjetoAutoProposto:**
    - São os três um tipo de proposta que se diferenciam pelas suas caraterísticas adicionais.

pt.isec.pa.apoio_poe.ui.text:

- **GestaoEstagioUI:** É constituído por métodos que permitem interação aplicação/utilizador;

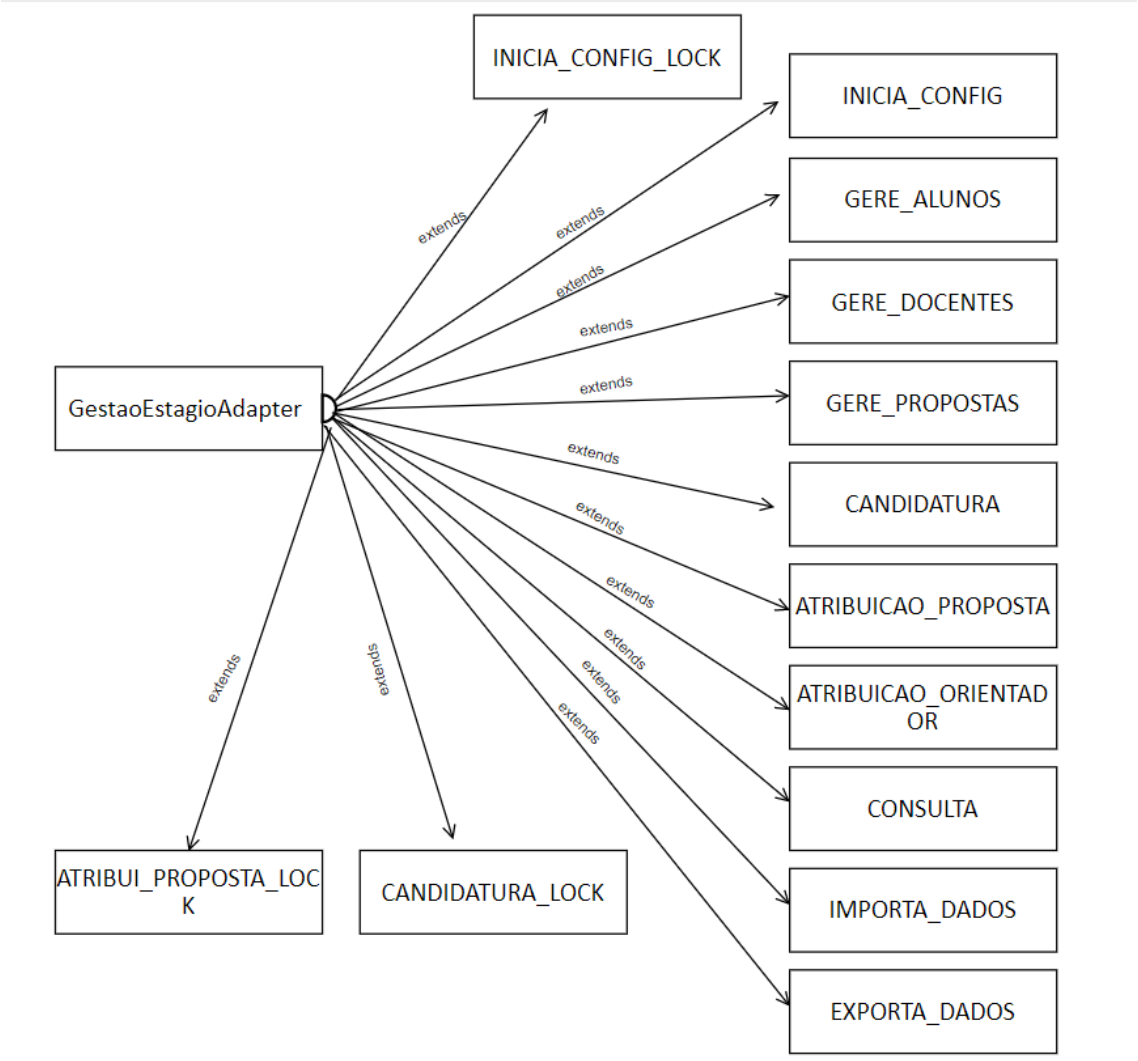
pt.isec.pa.apoio_poe.ui.gui :

- **MainJFX:** Inicia a interface gráfica através da criação de um Stage;
- **RootPane:** Contém a StackPane com todos os componentes UI;
- **InicioUI:** Ponto Origem da Aplicação;
- **IniciaConfigUI:** BorderPane aliado a uma AppBar para gerir alunos, docentes, propostas, importações, exportações e mudanças de fase;
- **GestaoAlunosUI:** BorderPane aliado a um contexto observável para realizar os métodos do GestãoAlunosState;
- **GestaoDocentesUI:** BorderPane aliado a um contexto observável para realizar os métodos do GestãoDocentesState**;**
- **GestaoPropostaUI:** BorderPane aliado a um contexto observável para realizar os métodos do GestãoPropostasState;
- **ImportaDadosUI:** BorderPane aliado a um contexto observável para realizar os métodos do ImportaDadosState;
- **ExportaDadosUI:** BorderPane aliado a um contexto observável para realizar os métodos do ExportaDadosState;
- **GereCandidaturaUI:** BorderPane aliado a um contexto observável para realizar os métodos do GereCandidaturaState;
- **AtribuicaoPropostaUI:** BorderPane aliado a um contexto observável para realizar os métodos do AtribuicaoPropostaState;

- **AtribuicaoOrientadorUI:** BorderPane aliado a um contexto observável para realizar os métodos do AtribuicaoOrientadorState;
- **GereConsultaUI:** BorderPane aliado a um contexto observável para realizar os métodos do GereConsultaState;
- **IniciaConfigLockUI:** BorderPane aliado a um contexto observável para consultar dados que foram modificados ou adicionados á aplicação durante a fase **INICIA_CONFIG**;
- **GereCandidaturasLockUI:** BorderPane aliado a um contexto observável para consultar dados que foram modificados ou adicionados á aplicação durante a fase **GERE_CANDIDATURA**;
- **AtribuicaoPropostaLockUI:** BorderPane aliado a um contexto observável para consultar dados que foram modificados ou adicionados á aplicação durante a fase **ATRIBUICAO_PROPOSTA**;
- **AppBar:** BorderPane aliado a um contexto observável que gere os estados filho do **INICIA_CONFIG** ;
- **AppMenu:** MenuBar alidado a um contexto observável serve para guardar um ficheiro com a informação da aplicação, abrir um ficheiro com informação pre-existente da aplicação, editar (undo/redo) dados específicos da aplicação e fechar a aplicação;

5. Descrição do relacionamento entre as classes

A classe GestaoEstagioAdapter vai implementar a interface IGestaoEstagioState, definindo implementações default para os métodos da mesma. As classes que representam estados são extends da classe GestaoEstagioAdapter.



6. Funcionalidades implementadas

Funcionalidades	Implementada	Implementada Parcialmente	Não Implementada
5 fases da app	X		
Interface modo Texto	X		
Importação/Exportação de dados	X		
Gravação/Carregamento da app	X		
Undo / Redo	X		
Interface Gráfica	X		
Verificações Acesso a Estágio			X
Gráficos com Dados da App		X	

Gráficos com Dados da App:

Apenas apresentamos o gráfico de orientações com a Média, Mínimo, Máximo e orientação por docente no ConsultaUI