

**Programação Avançada**

Gestão de Projetos e Estágios



**Sérgio Costa 2020129026**

**Roberto Tarta 2020136419**

Licenciatura em Engenharia Informática – Curso Europeu

Coimbra, 25 de abril de 2022

Índice

[Introdução 3](#_Toc102332870)

[Organização 3](#_Toc102332871)

[Decisões tomadas na implementação 3](#_Toc102332872)

[Diagrama da máquina de estados 4](#_Toc102332873)

[Classes utilizadas 5](#_Toc102332874)

[Descrição do relacionamento entre as classes 5](#_Toc102332875)

[Funcionalidade da aplicação 5](#_Toc102332876)

# Introdução

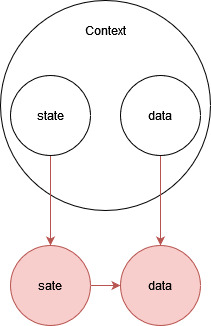
Este trabalho prático está a ser desenvolvido no âmbito da unidade curricular de Programação Avançada. O projeto consiste na implementação de uma aplicação de apoio ao processo de gestão de projetos e estágios do Departamento de Engenharia Informática e de Sistemas do ISEC.

# Organização

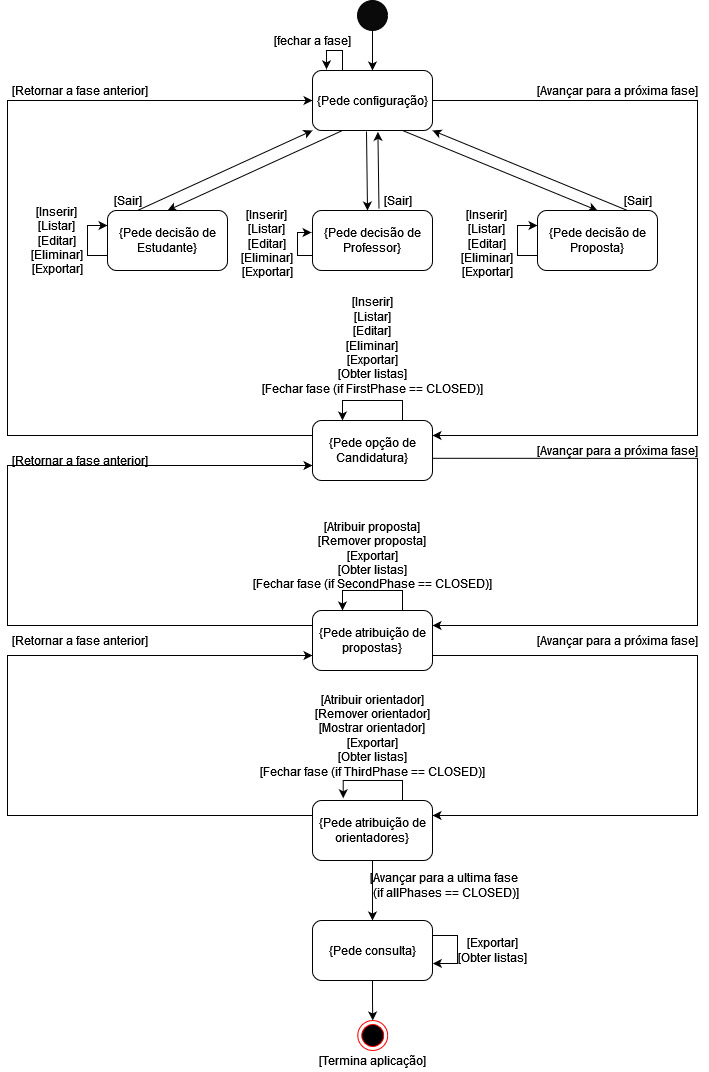
Seguindo as indicações sobre a arquitetura do projeto, nós criamos a seguinte organização dentro da package “**pt.isec.pa.apoio\_poe**”, incluindo dentro desta as seguintes packages:

* **model** - contem todas as classes que implementam a gestão de PoE e toda a logica associada.
  + **fsm** - contém toda a hierarquia dos estados, incluindo a interface e a classe Context, para a interface poder aceder a todas as funcionalidades.
  + **data** - contém as classes que representam as estruturas de dados e que disponibilizam todas as funcionalidades.
* **ui** - contem todas as classes que implementam a interface, seja em modo texto (meta 1), ou em modo gráfico (meta 2)
  + **text** - classe que implementa a interface em modo texto.
  + **gui** - classes que implementam a interface em modo gráfico em JavaFx (apenas será desenvolvido na segunda meta).
* **utils** – contem a classe que vai ajudar a ler os dados introduzidos pelo utilizador, é usado pela interface

# Decisões tomadas na implementação

Para poder seguir as várias implementações que nos foram ensinadas nas aulas. Nós optamos por criar um contexto que guarda o estado e os dados. Para fazer a serialização e deserialização, pensamos em criar uma “capa” para carregar somente o método, mas depois implementamos da seguinte forma:

# Diagrama da máquina de estados

Para ajudar na implementação deste projeto, nos criamos um diagrama de máquinas de estados para ajudar na interação UI – Aplicação. Com este processo, nos conseguimos aceder a toda a informação essencial e perceber quando é que o programa vai mudar de estado e como devemos lidar com o processo. Na imagem abaixo encontra-se o nosso diagrama de estados, que serviu como base para a nossa implementação. Os estados escolhidos foram com base nos momentos em que o utilizador tem que tomar uma decisão e essa mesma decisão provoca a alteração dos dados e por consequência permite que o programa possa continuar em funcionamento.

# Classes utilizadas

Todas as ações descritas no diagrama de maquina de estados como *“pede configuração”*, “*pede decisão do estudante*”, “*pede decisão do professor*”, “*pede decisão da proposta*”, “*pede opção de candidatura*”, “*pede atribuição de propostas*”, “*pede atribuição de orientadores*” e “*pede consultas*”, deram origem as classes “*FirstPhaseState*”, “*StudentState*”, “*TeacherState*”, “*ProposalState*”, “*SecondPhaseState*”, “*ThridPhaseState*”, “*FourthPhaseState*” e “*FifthPhaseState*” respetivamente implementadas no código deste projeto.

Também foi necessário implementar mais classes, como a *Student*, *Teacher*, *Application* *ManagementPoE*, *Proposal* e as suas derivadas, *ProposalIntership*, *ProposalProject*, *ProposalSelfProposed*, estas permitam criar elementos como alunos, professores, candidaturas e propostas. No caso da classe *ManagementPoE* ela permite guardar todas as informações com as listas dos elementos descritos acima.

# Descrição do relacionamento entre as classes

Para poder criar código mais funcional decidimos derivar a classe *Proposal* em 3 derivadas, como descrito no ponto acima, para poder ter um melhor aproveitamento do código.

# Funcionalidade da aplicação

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Feito** | **Parcial** | **Por fazer** |
| Fase 1 | *X* |  |  |
| Fase 2 | *X* |  |  |
| Fase 3 | *X* |  |  |
| Fase 4 | *X* |  |  |
| Fase 5 | *X* |  |  |
| Serialização | *X* |  |  |
| Exportar e importar dados de ficheiros |  | *X* |  |