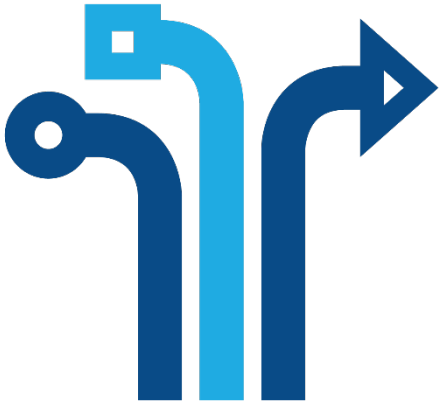
Trabalho Prático – Linguagem de Programação 2



ESCOLA
SUPERIOR
DE TECNOLOGIA
IPCA

Duarte Ribeiro de Melo, 21149

Professor Luís Ferreira

Licenciatura em Engenharia de Sistemas Informáticos

Descrição

O relatório infra descreve o desenvolvimento do projeto realizado no âmbito da unidade curricular Linguagem de Programação 2, do curso de Engenharia de Sistemas Informáticos do Instituto Politécnico do Cávado e do Ave.

Tema escolhido

Escolhi um dos temas presentes no enunciado:

"Sistema que permita a gestão de ocorrências de incêndios florestais e outras catástrofes naturais."

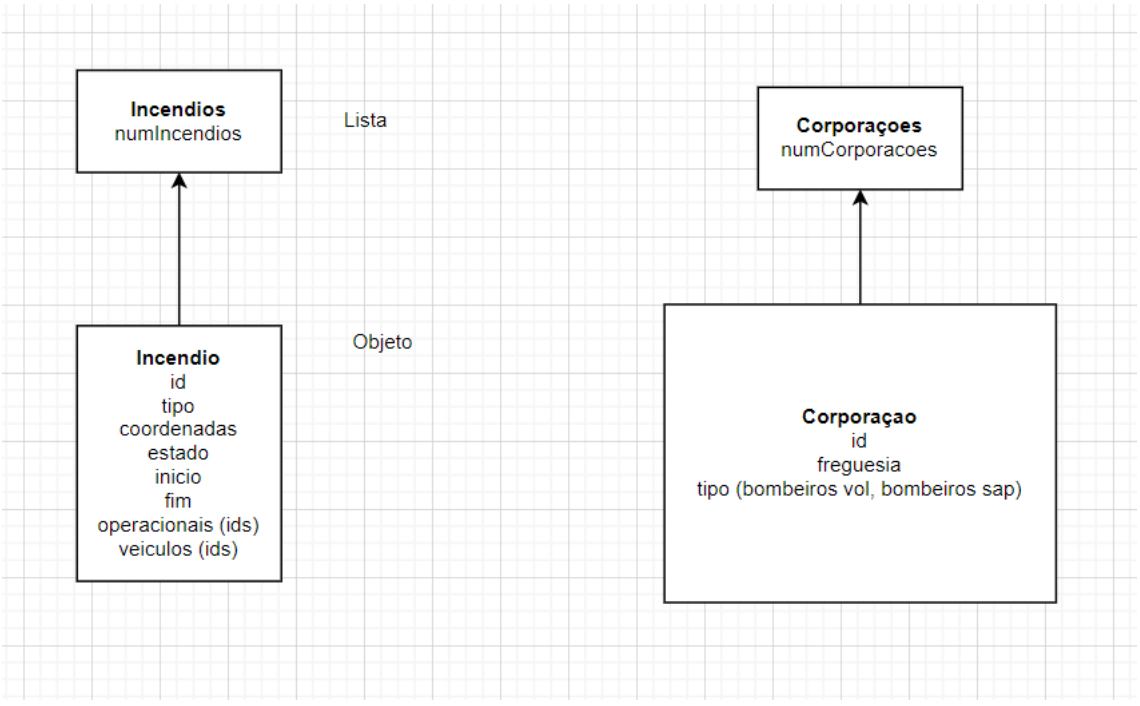
Decidi, a partir do tema supramencionado, focar-me mais na área dos incêndios – florestais ou não. O objetivo da minha aplicação é gerir as catástrofes (neste caso, os incêndios) – adicionar, mudar o estado (se já esta extinto), adicionar operacionais no terreno, adicionar viaturas no terreno, etc.

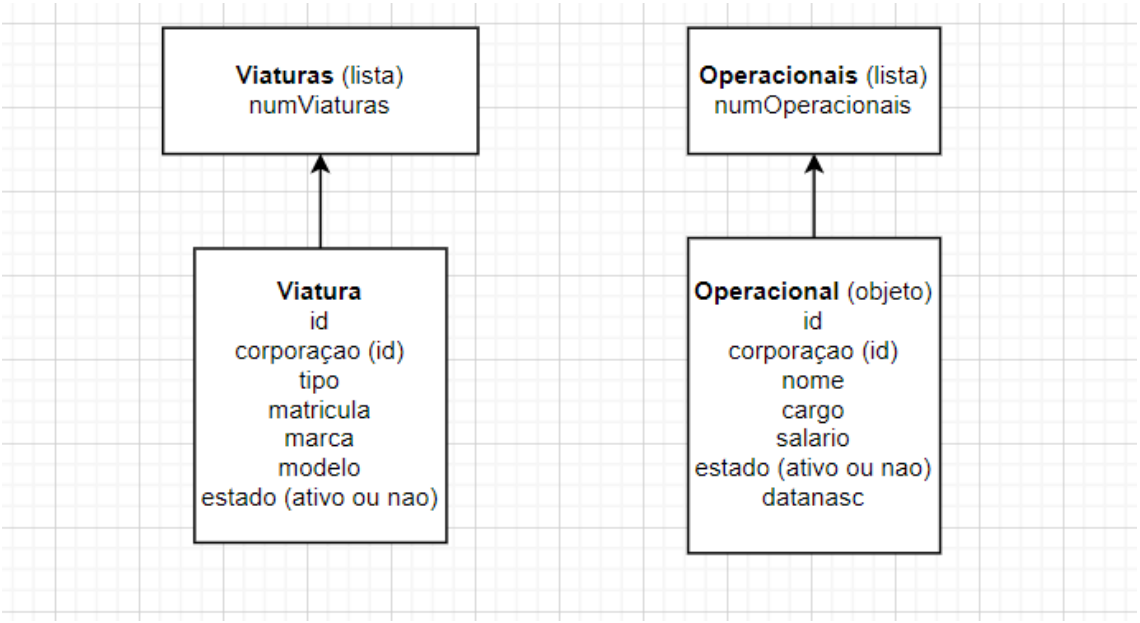
Estrutura de dados escolhida

Optei por utilizar listas no meu trabalho – em primeiro lugar porque acho mais fácil, e porque não tenho de me preocupar com a alocação de memória.

Utilizei as listas para “alojar” objetos das classes inframencionadas.

Classes criadas (primeira fase)





Classes / DLLs criadas (segunda fase)

- **Corporação**
- **Incendio**
- **Operacional**
- **Viatura**

Estas quatro classes foram divididas por camadas (cada uma tem 5 DLLs):

- Camada BO, Business Object – cria o objeto, manipula o mesmo
- Camada BR, Business Rules – regras de negócio relativas a cada objeto
- Camada Data – trata de armazenar os objetos em listas (esta é apenas acedida pela camada BR!
- Camada Input – trata da interação com o utilizador, criação de objetos com dados introduzidos pelo mesmo, etc.
- Camada Output – escreve o objeto no ecrã, caso pedido / necessário

- **Generals**
  DLL responsável por outputs "gerais".
  Possivelmente deixaria de existir caso tivesse criado Exceptions.
- **UI**

  Interação com o utlizador.

  Lê o menu do programa de um ficheiro de texto, escreve-o no ecrã e lê a opção do utilizador, a partir daí, utiliza funções das camadas Input dos objetos.

Interação entre Libraries

A DLL BO de cada objeto não utiliza nenhuma outra DLL.

A DLL BR utiliza as DLLs BO (para poder receber o objeto como parâmetro), Data (para poder chamar funções para inserção de objetos na lista) e Outputs. A BR é a única que deve ser capaz de aceder ao Data.

A DLL Data utiliza as DLLs BO (para poder ter listas deste objeto, receber o objeto como parâmetro, etc.) e a DLL Output, para poder imprimir toda a lista no ecrã (chama uma função do output).

A DLL Input utiliza as DLLs BO (para poder criar objetos dentro de funções, e "enviá-los" para a camada regras) e BR.

A DLL Output apenas utiliza a BO, para, por exemplo, receber uma Lista<ObjetoBO> como parâmetro ou fazer ObjetoBo.ToString().

O Menu utiliza as BRs e os Input.

O Main apenas utiliza o Menu.

Trabalho desenvolvido (primeira fase)

Até agora criei praticamente todas as classes (faltam as classes “operacional” e “operacionais”) – implementação essencial das classes; porém, considero que ao longo do resto do desenvolvimento do projeto e consoante as necessidades do mesmo, posso vir a precisar de fazer alterações nas mesmas.

Criei também algumas funções capazes de adicionar, remover, listar dados das classes respetivas.

Fiz alguns testes para entender se o que tinha programado estava a “agir” da forma que eu esperava.

Trabalho desenvolvido (segunda fase)

SOLID

- O projeto foi desenvolvido com vista a respeitar os princípios SOLID.

NTier

Logo depois de ter sido abordado na aula a arquitetura NTier, converti todo o projeto para esta arquitetura, dado que vi bastantes vantagens nesta conversão e o facto de ter o projeto por camadas torna-o mais fácil de manter e editar.

Libraries / DLLs

Para cada objeto, foram criadas ao todo cinco DLLs (bibliotecas): BusinessObject, BusinessRules, Data, Input e Ouput.

Desenvolvi uma DLL responsável pela interface, que lê um menu de um documento de texto.

Documentação

Melhorei também a documentação de todo o projeto, assinei documentos, documentei funções, etc.

Operadores e overrides

Defini operadores para todos os objetos e overrides (ToString e Equals).

Enums

Criei enumeradores para tipos de incêndios, tipos de corporações, tipos de viaturas, etc.

Outros

Removi as antigas classes (antes do NTier).

Dificuldades encontradas (primeira fase)

A maior dificuldade para mim até agora foi, sem dúvida, “encaixar” as heranças, os “access modifiers” (public, private, protected) – o que deve estar public, private, protected e como aceder e de onde é que deve ser possível aceder…

Fora estes dois assuntos, a criação das classes foi relativamente fácil e intuitiva, e consegui entender as vantagens da criação das mesmas.

Dificuldades encontradas (segunda fase)

Após estudo para o teste, depois de rever algumas aulas e mesmo com a implementação de NTier, consegui “encaixar” os access modifiers (o que deve ser acedido por determinada class ou library, etc.).

Nesta segunda fase, a maior dificuldade (que, por ter deixado para o fim, acabei por não superar) foi a criação de Exceptions e a sua utilização.

Conclusão

A realização deste projeto serviu, sem dúvida, para aplicar o programa lecionado durante as aulas da UC (aplicação dos princípios SOLID, NTier, OOP, etc.). Serviu também como estudo para o teste da Unidade Curricular.

Após redação deste relatório, foi notória uma grande evolução da primeira fase para a segunda.

Programas/websites utilizados

- Visual Studio Community 2019
- GitKraken / Git / GitHub - https://github.com/duartemelo/LP2
- Trello - https://trello.com/b/NP5Bhka1/lp2