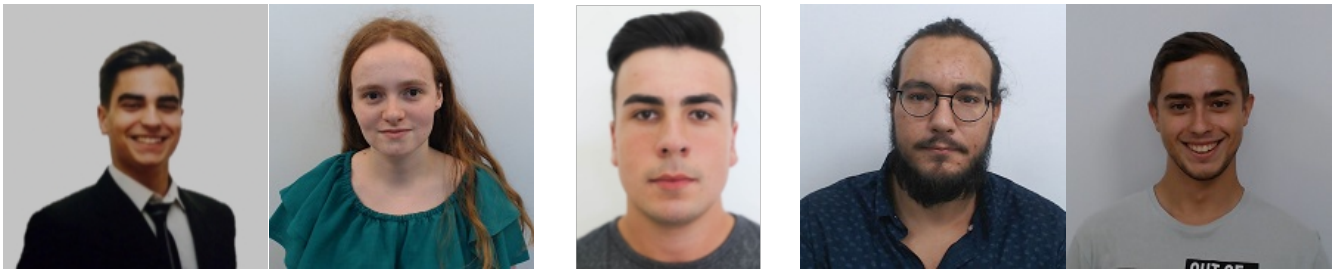
Universidade do Minho
Escola de Engenharia

Desenvolvimento de Sistemas de Software
Relatório de Projeto: FASE 3
2022/2023
Repositório do Trabalho Prático (GitHub)

Afonso Xavier Cardoso Marques 94940
Ana Filipa da Cunha Rebelo 90234
Délio Miguel Lopes Alves 94557
Pedro Miguel Duarte Araújo 70699
Ricardo Manuel Almeida Vieira De Castro 70492



Índice

Lista de Figuras

1 Introdução

O presente relatório visa apresentar a terceira e última fase do trabalho proposto no âmbito da Unidade Curricular de Desenvolvimento de Sistemas de Software.

No decorrer da primeira fase do projeto elaboraram-se o **Modelo de Domínio** e os **Modelos de Use Cases**, responsáveis por dar uma visão geral sobre a forma como as diferentes entidades do sistema se relacionam e quais as funcionalidades a serem implementadas posteriormente.

A implementação da segunda fase, foi efetuada sobre a base construida na fase anterior e, começou a descrever a aplicação a ser desenvolvida num nível de detalhe bastante superior. No decorrer desta fase produziram-se os **Diagrama de Componentes**, **Diagrama de Classes**, assim como os **Diagramas de Sequência**.

Concluídas as duas primeiras fases, a visão sobre a solução a desenvolver é bastante clara. Deste modo, a presente fase do projeto irá se focar na implementação tendo como base todo o trabalho de planeamento desenvolvido nas fases anteriores com o objetivo de produzir uma solução que cumpra com todos os requisitos especificados anteriormente.

Ao longo do presente documento serão apresentadas as alterações realizadas ás fases anteriores; os **DAOs** criados, todos as decisões tomados no decorrer da implementação da solução e por fim uma análise crítica dos resultados obtidos.

2 Alterações à 2 fase

No decorrer da presente fase, surgiu a necessidade de efetuar pequenas alterações nos diagramas desenvolvidos anteriormente com o intuito de complementar a visão da solução assim como mais completa e funcional. Esta necessidade advém de ser preciso ter um ponto de partida que reflita a aplicação que se quer modelar, e é crucial que este novo planeamento seja feito antes de se começar com a escrita do código. As alterações passaram por acrescentar no diagrama de packages a representação dos DAOs e acrescentar alguns métodos de forma a completar a implementação do código.

2.1 Diagrama de Classes

Em relação ao Diagrama de Classes desenvolvido e entregue na segunda fase, foram feitas as seguintes alterações:

2.2 Package SubCircuito

Adicionamos as Classes Tipo de Estrada,Curva,Reta e Chicane para uma melhor representação dos diferentes tipos de Estrada.

2.3 Adição de Packages

Adicionamos o package data para garantir o acesso à base de dados. Estas adições serão explicadas com mais detalhe posteriormente. O Package util contém classes auxiliares para a clareza e eficiência do código. Foi adicionado um Package Exceptions que contém todas as Exceptions utilizadas no package business.

2.4 Diagrama de Packages

Em relação à fase anterior, foi necessário efetuar alterações no diagrama de packages de modo a incluir a base de dados. O nosso projeto encontra-se organizado através de uma arquitetura em 3 camadas, visível através da existência dos packages **UI, Business e Data**, que constituem as camadas de apresentação, de lógica de negócio e de dados, respetivamente. Deste modo, obteve-se o diagrama que pode ser observado nos anexos.

3 Data Access Objects

Uma dos grandes desafios da presente fase passou pela transformação de todas as estruturas que seriam guardadas em memória para **Data Access Objects (DAOs)**.

DAOs são classes que permitem separar a lógica de negócio da lógica de persistência. A lógica de negócio conhece a *API* do **DAO**, no entanto não tem contacto com nenhuma das operações que são realizadas internamente nem conhecimento do modo como os dados são posteriormente guardados. O **DAO** funciona assim como uma *caixa negra*, providenciando todos os dados pedidos à lógica de negócio e pode ser mudado livremente. Deste modo. é possível utilizar o mesmo noutra implementação, sem por isso afetar o funcionamento da aplicação, desde que implementada a mesma *API*.

Na solução apresentada e de maneira a que fosse possível o armazenamento de informação numa Base de Dados, recorreu-se à biblioteca **JDBC**. Tendo se implementado **DAOs** para todos os dados que fossem suscetíveis de serem alterados no decorrer do funcionamento da aplicação. Deste modo, é possível manter a informação guardada mesmo que a aplicação tenha sido encerrada. Os **DAOs** desenvolvidos foram:

- Piloto;
- Administrador;
- Campeonato;
- Carro;
- Circuito;
- DAOConfig;
- Jogador.

3.1 Piloto

Corresponde aos **Pilotos**, será representado através de uma tabela composta pelas colunas **username**, **cts** e **sva**. O **username** é a chave primária.

3.2 Administrador

A tabela de **Administrador** é composta pelas seguintes colunas **username** e **password**.

3.3 Campeonato

Em **Campeonato** a composição da tabela passa por um única coluna **nome** que serve como chave primária.

3.4 Carro

Carro será representado por uma tabela composta pelas colunas **Id_Carro**, como chave primária; **Categoria**, **Marca**, **Modelo**, **Cilindrada**, **Potencia**, **Potencia_Eletrico**, **Fiabilidade**, **Tempo**, **DNF**, **Pneu** e **Pac**.

3.5 Circuito

Neste **DAO** dá-se a criação de múltiplas tabelas. Uma delas **Circuito** com as colunas **Nome**, que também é chave primária; **Distancia**, **Num_Voltas**, **Penalização** e **Recorde**. Sendo as restantes tabelas criadas referentes ao **Tipo de Estrada**, **Tempo médio por circuito** e **Recordes**.

3.6 Jogador

Em **Jogador** a tabela é composta por um **Username**, como chave primária; **Password**, *PontosJogador*, *IsAutenticado*, *PontosCorrida*.

3.7 DAOConfig

Responsável por estabelecer a conexão à base de dados (BD), os campos **Username** e Password são utilizados para autenticar o acesso à mesma. O campo **Database** especifica o nome da BD ao qual a aplicação se irá conectar. O campo **URL** especifica o local e o fuso horário a serem utilizados ao conectar-se.

4 Implementação da Aplicação

O projeto está dividido em cinco partes: SubUtilizador, SubCircuito, SubCorrida, SubCarro e SubCampeonato.

- **SubUtilizador**
  - **Administrador -** classe contém variáveis referentes ao administrador como o username, a password e seu autenticador.
  - **Jogador -** classe que contém importantes variáveis para caracterizar um jogador, como o username, a password, os totais obtidos pontos, os pontos obtidos nua corrida e a autenticação.
  - **SubUtilizadorFacade -** classe facade que implementa os métodos definidos na interface IUtilizador.
- **SubCircuito**
  - **Circuito -** classe que contém variáveis importantes para construção e descrição de um circuito como distância, voltas, tipo de estrada, record e tempos médios.
  - **Record -** classe que possui a variável tempo para fazer o record do num circuito.
  - **Piloto -** classe que incluí as variáveis referentes ao piloto, nomeadamente nome, sva, cts.
  - **TipoEstrada, Chicane, Circuito, Curva, Recta -** classes descrevem a estrutura de um circuito, suas curvas, retas e chicane.
  - **SubCircuitoFacade -** classe facade que implementa os métodos definidos na interface ISubCircuito.
- **SubCorrida**
  - **corrida -** classe que contém a função simular corrida, entre outras funções importantes para realizar a simulação como ultrapassagens, avarias e despistes. Sendo a classe responsável por determinar as possições dos jogadores de uma corrida e seus records.
  - **SubCorridaFacade -** classe facade que implementa os métodos definidos na interface ISubCorrida.
- **SubCarro**
  - **Carro -** classe abstrata que possuí variáveis de caracterização os diferentes carros, como o id, a marca, o modelo, a cilindrada, a potência, o modo do motor, a fiabilidade, o tipo do pneu, a PAC e ainda outras variáveis importantes para a simulação de uma corrida, tais como, a posição, o tempo e o DNF.
  - **GT, GTH, PC1, PC1H, PC2, PC2H, SC -** classes responsáveis pelos tipo de motores classe .
  - **SubCarroFacade -** classe facade que implementa os métodos definidos na interface ISubCarro.
- **SubCampeonato**
  - **Campeonato -** classe contém variáveis que caracterização um campeonato como o nome, a lista de corridas que integra e classificações obtidas.
  - **SubCampeonatoFacede -** classe facade que implementa os métodos definidos na interface ISubCampeonato.

ISubUtilizador - classe que representa a interface que é implementada pela secção SubUtilizador, possui funções de registo, login e autenticação de utilizadores.

ISubCircuito - classe que representa a interface que é implementada pela secção SubCircuito, possui funções de adição e remoção de circuito, adição de piloto e verificação de circuito e piloto.

ISubCarro - classe que representa a interface que é implementada pela secção SubCarro, função para obter a categoria do carro, para obter carro, para adicionar carro, para remover carro.

ISubCorrida - classe que representa a interface que é implementada pela secção SubCorrida, possui função de adicionar de piloto à lista de pilotos disponíveis, de verificar se existe um piloto na lista dos disponíveis e de verificar se existe um carro na lista dos disponíveis.

ISubCampeonato - classe que representa a interface que é implementada pela secção SubCampeonato, possui função de verificar se existe um campeonato na lista dos disponíveis, de adicionar um campeonato à lista de campeonatos disponíveis, de verificar se existe um circuito na lista dos disponíveis e de adicionar um circuito à lista de circuitos disponíveis.

4.1 Estratégias adotadas no desenvolvimento projeto

Na simular a corrida tendo em conta que os carros podiam ultrapassar, despistar e avariar, então criamos algumas condições tendo em conta as características dos carros e dos seus pilotos. Para simular uma avaria, demos percentagens diferentes a cada modo do motor (Conservador 22, Normal 33, Agressivo 45). Para simular uma ultrapassagem, tivemos em conta diversos fatores, o GDU, o clima, o SVA, o CTS, PAC e a potência. Para simular um despiste, nós tivemos em conta o PAC e a potência do carro.

5 Conclusão

Com a finalização da presente etapa é possível percecionar o impacto real que as fases anteriores tiveram. Apesar de a terceira fase se ter demonstrado extremamente desafiante, foi de extrema importância todo o trabalho desenvolvido anteriormente uma vez que permitiu o melhorar e otimizar a implementação da solução.

Apesar de não terem sido cumpridos todos os objetivos da presente fase, o trabalho desenvolvido encontra-se robusto devido à planificação realizada no decorrer das fases anteriores.

Como trabalho futuro será interessante efetuar um refinamento da solução de modo a para seguir em maior concordância os use cases, aplicar ligeiras modificações à simulação de modo a que esta corra sem erros e implementar a versão premium.

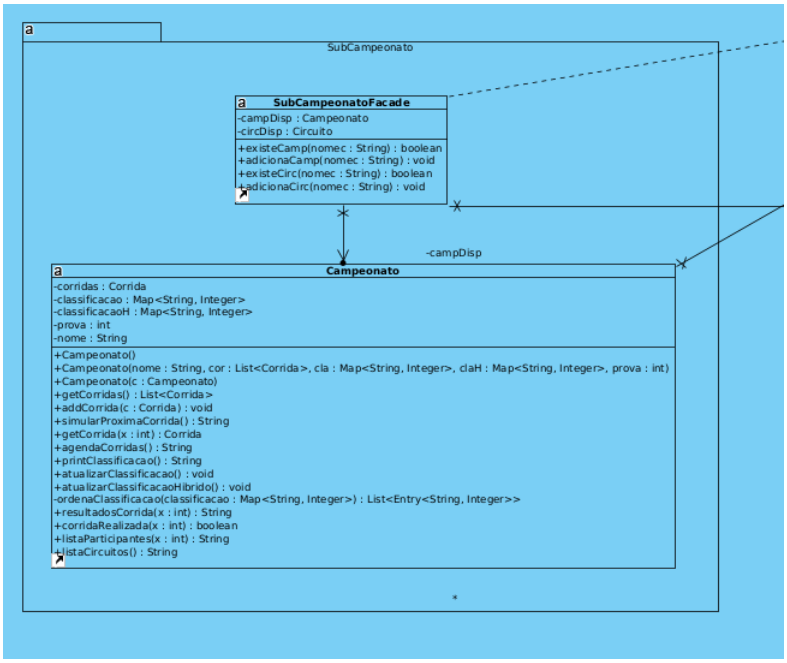


Figure 1: Package Campeonato.

6 Anexos

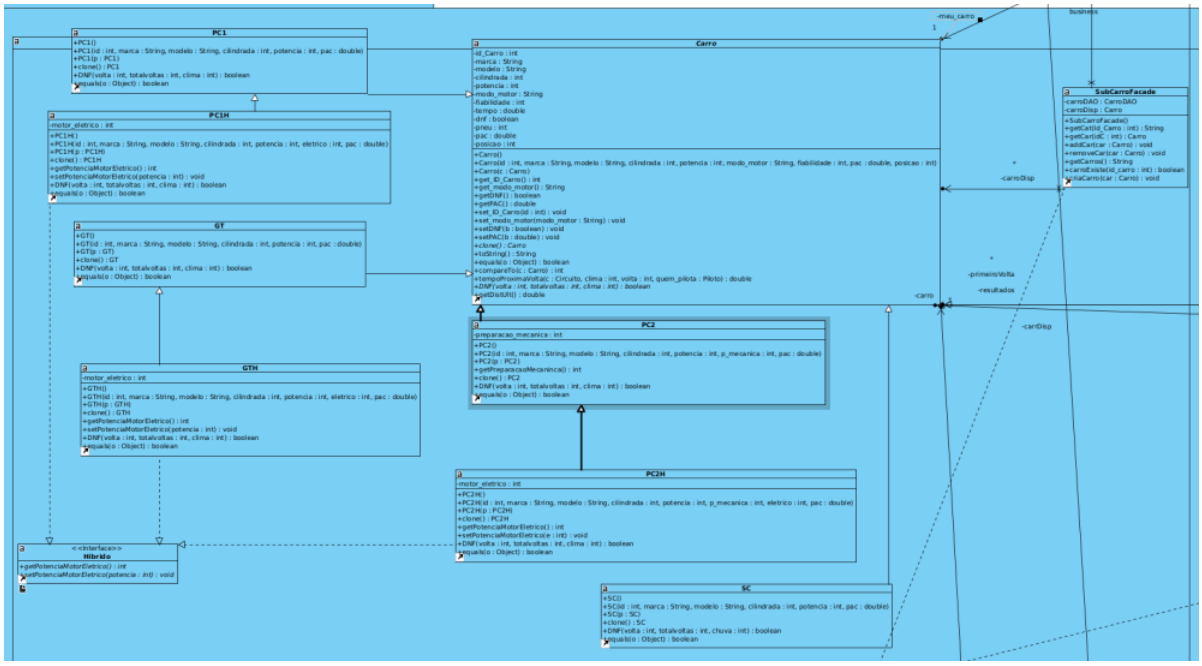


Figure 2: Package Carro.

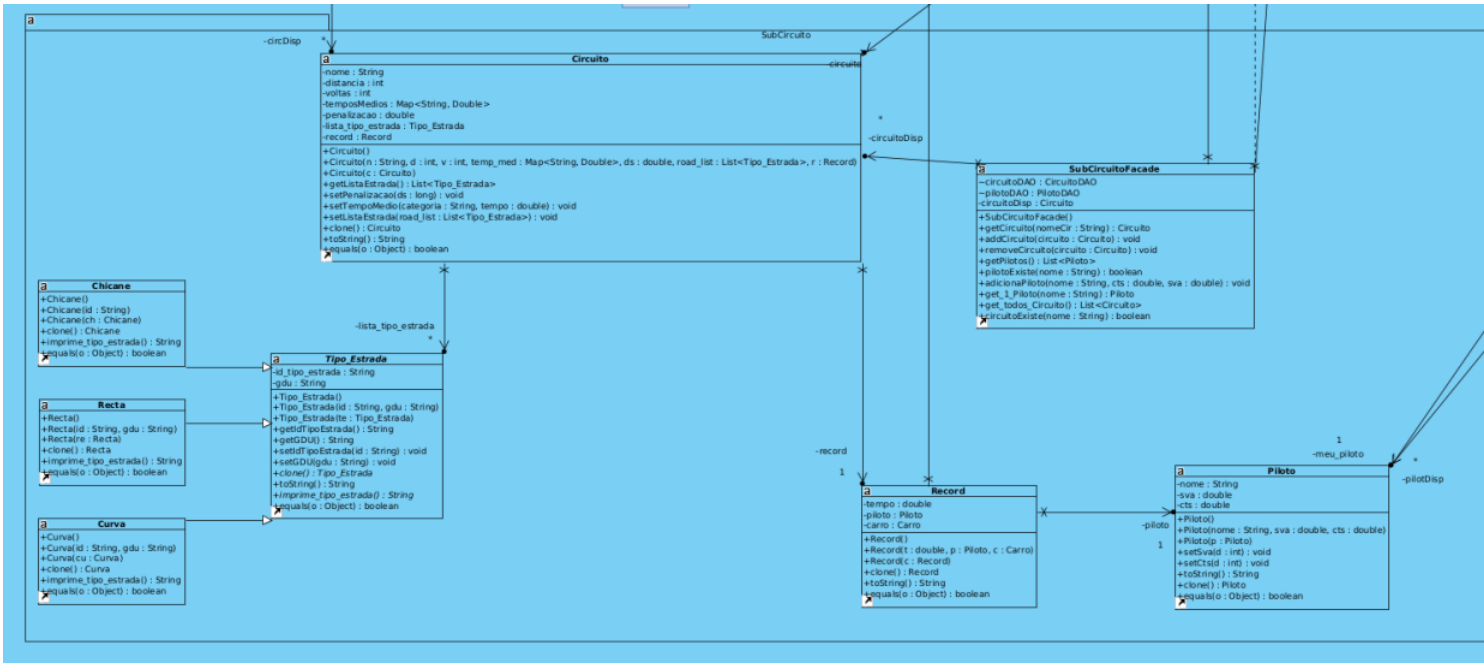


Figure 3: Package Circuito.

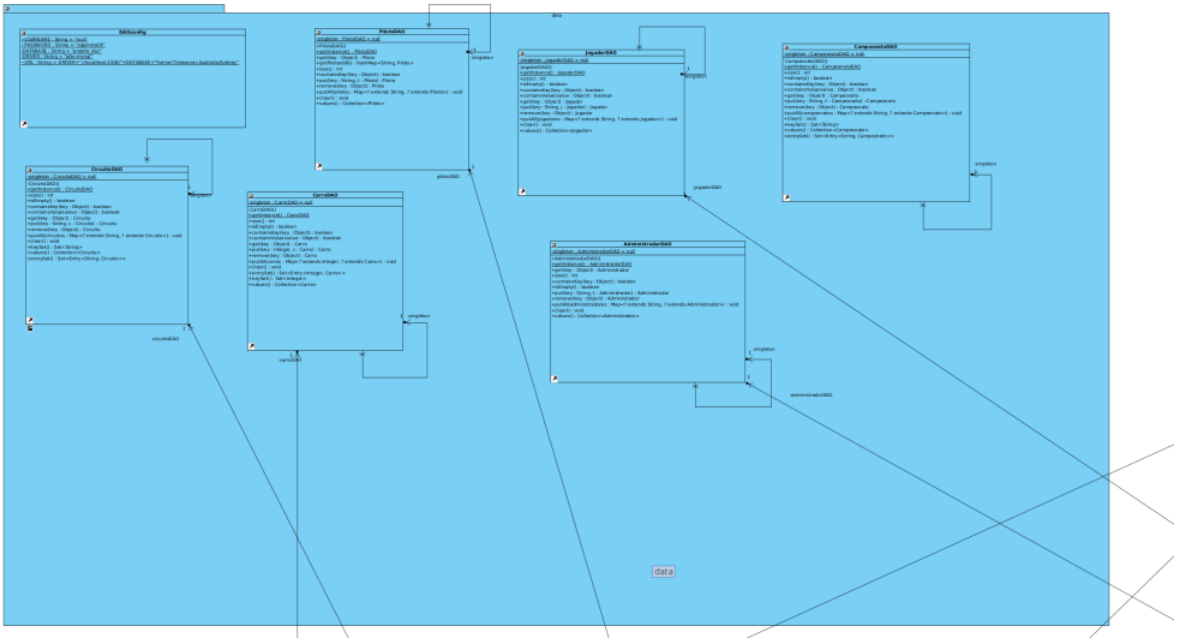


Figure 4: Package Data.

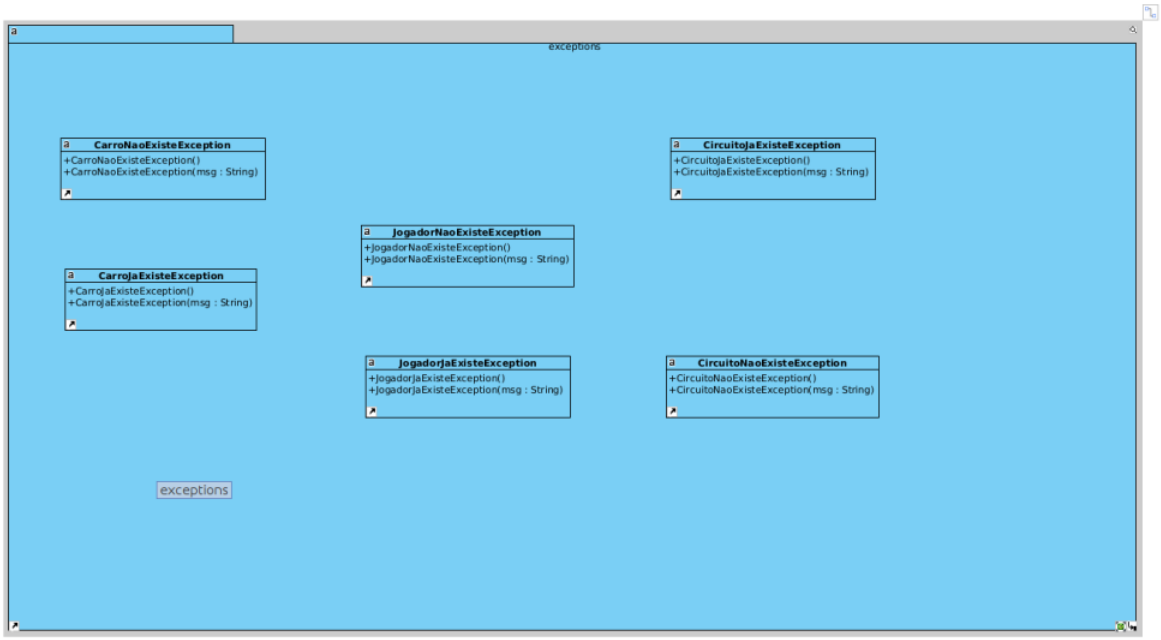


Figure 5: Package Exceptions.

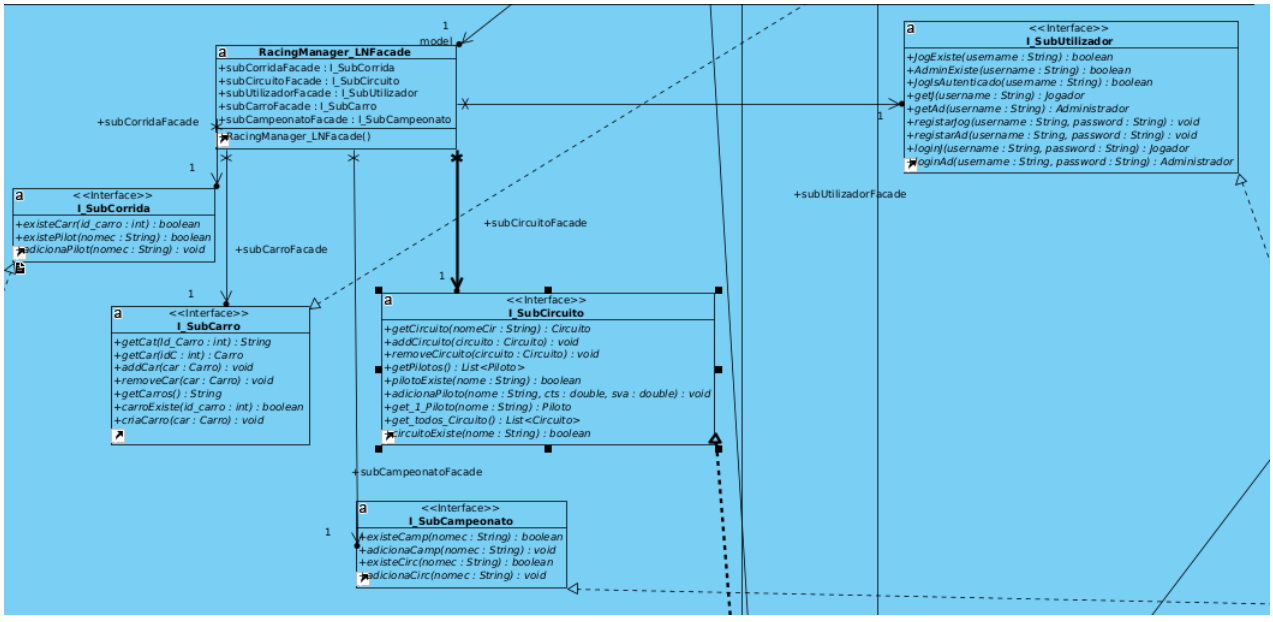


Figure 6: Package Business.

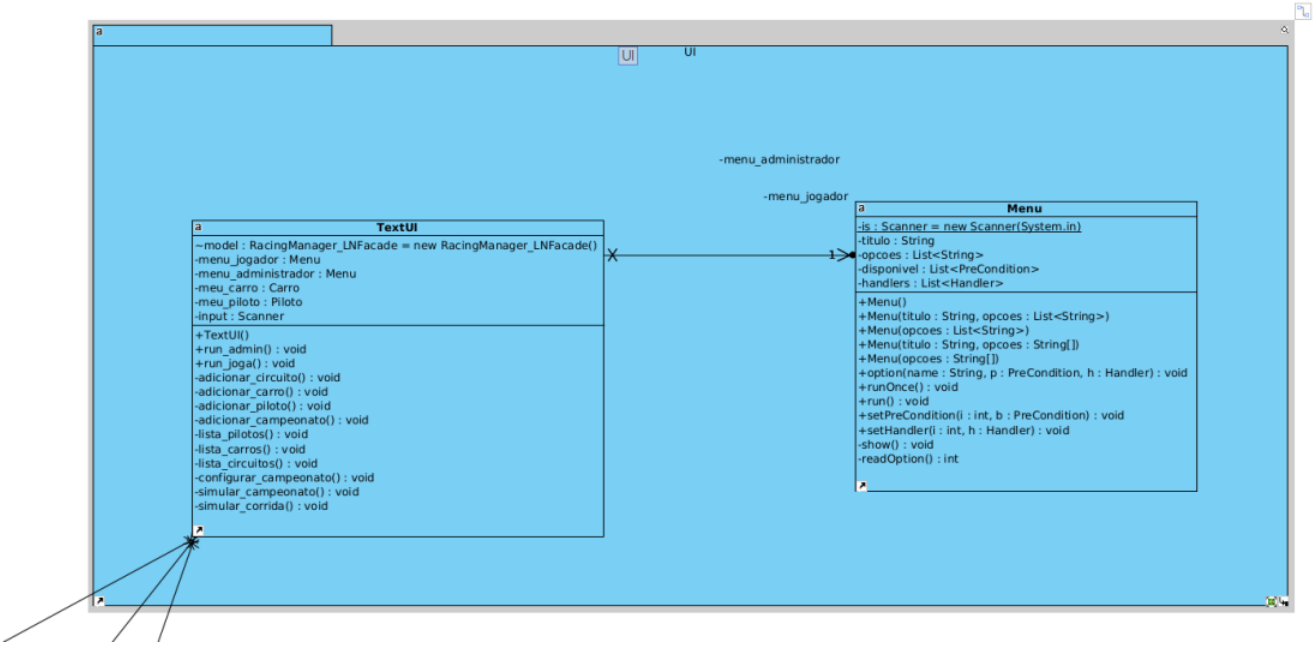


Figure 7: Package UI.

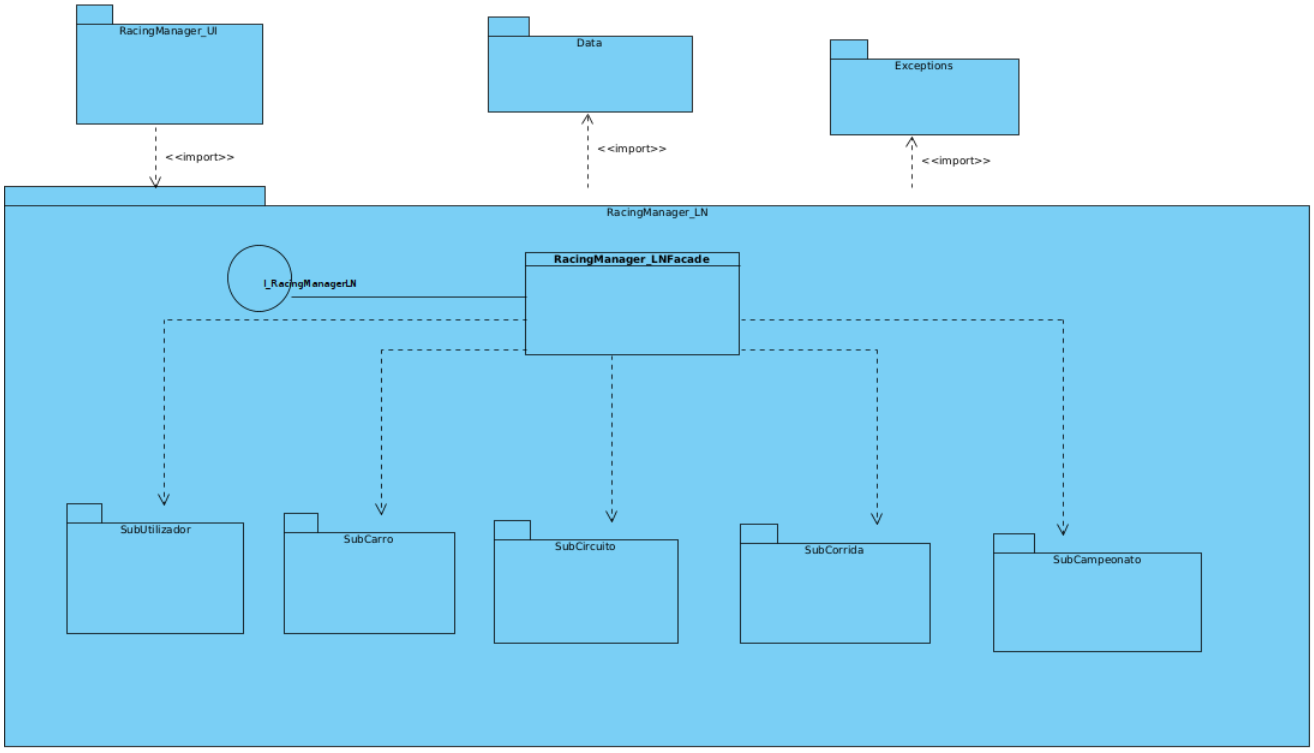


Figure 8: Diagrama de Packages Atualizado.