



Universidade do Minho





Escola de Engenharia

Licenciatura em Engenharia informática

Trabalho Prático – Entrega Final

Unidade Curricular de

Desenvolvimento de Sistemas Software

<i>A96547</i> <i>Rodrigo José Teixeira</i> <i>Freitas</i>		<i>A95076</i> <i>Pedro Marcelo</i> <i>Bogas Oliveira</i>	
<i>A92974</i> <i>José dos Santos</i> <i>Mendes</i>		<i>A93186</i> <i>Eduardo Fernando</i> <i>Cruz Henriques</i>	

Índice

1. INTRODUÇÃO	3
2. ESPECIFICAÇÕES DA IMPLEMENTAÇÃO	4
2.1. Os UTILIZADORES – TIPOS DE USER.....	4
2.2. Os UTILIZADORES – <i>LOGIN</i> E REGISTO.....	5
2.3. Os CARROS – TIPOS DE CARRO.....	5
2.4. Os CARROS – ARMAZENAMENTO.....	7
2.5. AS CORRIDAS – CIRCUITOS E SECÇÕES.....	7
2.6. AS CORRIDAS – PARTICIPANTES E PILOTOS.....	7
2.7. AS CORRIDAS – CORRIDAS E CAMPEONATOS	8
2.8. AS CORRIDAS – LOBBYS	8
3. CAMADA DE DADOS	9
4. CAMADA DE NEGÓCIO.....	11
5. CAMADA DA INTERFACE DO UTILIZADOR/DIAGRAMA COMPLETO	13
6. CONCLUSÃO	15

1. Introdução

Nesta última fase do nosso projeto procuramos aplicar o conhecimento que obtivemos na área de Desenvolvimento de Sistemas de Software ao longo deste semestre em conjunto com aquilo que apresentamos nas duas primeiras fases deste projeto para implementar uma versão robusta e otimizada de um sistema de simulação de corridas de automobilismo.

Para além disto, também iremos apresentar o processo de ligação da camada de negócio do nosso sistema à camada de dados, e avaliar a utilidade que o processo de desenvolvimento que abordamos nas aulas teve no processo de implementação do nosso trabalho, assim como a qualidade do mesmo.

Finalmente, iremos atualizar o diagrama de classes feito na segunda fase para refletir a implementação do simulador com os DAOs e com as mudanças que vimos como sendo necessárias para o sistema funcionar de acordo com aquilo que foi especificado nas fases anteriores e no enunciado.

2. Especificações da Implementação

2.1. Os Utilizadores – Tipos de User

A nossa lógica para implementar os tipos diferentes de utilizador basearam-se nas necessidades diferentes de cada utilizador. Deverá ser possível um jogador entrar num campeonato sem um username e password, mas o progresso que ele faz nessa sessão(pontos) será ignorado. Também sabemos que um administrador deve ter privilégios para alterar os dados do sistema (circuitos, carros, etc...) mas não poderá participar em campeonatos.

Logo, criamos 3 coleções diferentes para cada tipo de utilizador:

- Uma coleção para os Convidados, que apenas terão um *username* gerado com base no número de convidados criados até ao momento. Por exemplo, se já foram criados 10 convidados ao longo do funcionamento do sistema, o nome gerado para o convidado novo será “convidado 11”. Visto que este tipo de utilizador não tem password e pontos, e não pode ser premium, apenas tem o *username* como variável. Cada convidado é associado ao seu nome (“Convidado x” -> objeto Convidado).
- Uma coleção para os Jogadores, que terão de fazer login com um *username* e *password*. Os Jogadores poderão ser *premium* ou não, mas este parâmetro apenas afeta as probabilidades durante as simulações. Para além disto, os pontos que ganha com as corridas deverão ser guardados e associados á conta. Cada Jogador tem estes 4 parâmetros e cada jogador é acessado através do seu *username*(“nome_jogador” -> objeto Jogador).
- Os Administradores terão a sua própria coleção, porque apenas será possível entrar no sistema como Administrador quando forem dados privilégios ao utilizador em questão. Eles terão, então, de estar isolados dos Jogadores/Convidados por motivos de organização e porque não têm a necessidade de ser *premium* ou não ou dos pontos acumulados, visto que não é possível participarem em torneios. Contudo, ainda têm de estar registados com um *username* e *password*. Obtemos cada Administrador a partir do seu *username*(“nome_admin” -> objeto Administrador).

2.2. Os Utilizadores – *Login* e Registo

O processo de registo e *login* é semelhante para os Jogadores e Administradores e inexistente para os Convidados. Um Jogador/Administrador insere o nome que quer que seja exibido e associado á sua conta e a sua password para entrar. Caso a conta não exista, deverá registar-se, inserindo o nome e *password* que deseja. É de notar que, mesmo havendo duas coleções para Jogadores e Administradores, o Sistema não permite a um Jogador criar uma conta com um nome que já está associado a um Administrador e vice-versa.

Outro aspeto em relação aos logins é que as *passwords* têm de ter mais de oito caracteres. Caso contrário tanto a tentativa de login como de registo são inválidas.

2.3. Os Carros – Tipos de Carro

Os carros usados nas simulações serão guardados numa “garagem virtual” e cada carro terá uma performance que depende de várias características. A maioria destas é partilhada entre todos os carros, mas existem alguns tipos de carro que têm aspetos únicos que serão tidos em consideração nas simulações. Todos os carros têm:

- Marca – a marca do carro;
- Modelo – o modelo do carro. Este atributo será usado como uma chave para obter cada carro, ou seja, não deverão existir dois carros com o mesmo modelo na garagem;
- Cilindrada – atributo que influencia o cálculo da fiabilidade dos carros e é alterável durante a criação na grande maioria deles;
- Potência – atributo que influencia o cálculo da fiabilidade, assim como a chance de ocorrer uma falha no motor durante a corrida;
- Perfil Aerodinâmico (entre 0 e 1) – atributo que funciona de maneira semelhante à Downforce.
- Downforce(entre 0 e 1) – atributo que influencia a chance de ultrapassar outro carro ao custo de aumentar a probabilidade de um despiste;
- Fiabilidade (entre 0 e 1) – o atributo que indica a percentagem do motor de não falhar durante o decorrer da corrida. É ajustado com base na classe do carro, e com a maioria dos atributos que podemos atribuir ao construir os carros.

- Tipo de Pneus – o tipo de pneus é uma característica que o utilizador irá selecionar ao escolher um dos carros para o campeonato. Logo, apesar deste atributo estar ligado ao carro, não terá tipo de pneus ao início. Os tipos de pneus que estarão disponíveis são “Macio”, “Duro” e “Chuva” e, dependendo da duração e clima da corrida, um tipo de pneus pode trazer vantagens em relação aos outros, como maior fiabilidade. Os pneus macios são melhores para corridas de curta duração, os duros funcionam melhor em corridas demoradas, e os de chuva reduzem a chance de haver um despiste quando o clima é chuvoso.
- Estado do Motor – o estado do motor será introduzido, á semelhança do tipo de pneus, pelo utilizador antes de cada campeonato começar. Os estados são “Conservador”, “Agressivo” e “Normal”. Alterar o estado do motor implica aumentar a chance de ultrapassar, mas também a chance de ocorrer uma falha.

Existem quatro tipos de carros neste sistema de simulação:

- Os carros C1: têm fiabilidade alta (93-97%), cilindrada constante (6000 cm³) e não possuem atributos adicionais. A potência varia dos 300 aos 1000 cavalos.
- Os carros C2: têm menor fiabilidade (75-85%), uma cilindrada variável (3000-5000 cm³) e possuem um atributo adicional: a afinação do motor(entre 0 e 1). Este atributo apenas é alterável pelo utilizador antes da corrida, e altera (???). A potência varia dos 300 aos 900 cavalos.
- Os carros GT: têm viabilidade inicial de 75%, cilindrada variável (2000-4000 cm³) e uma taxa de viabilidade (0.5-0.75%) que é calculada em função dos outros parâmetros que todos os carros têm. A taxa é retirada da viabilidade total no final de cada volta. No pior dos casos, a fiabilidade cai para 60% caso a taxa seja o valor máximo e as corridas têm 20 voltas. A fiabilidade não pode descer dos 60% após este ponto. A potência varia entre os 300 e 800 cavalos.
- Os carros SC: têm cilindrada constante (2500 cm³) e a sua viabilidade apenas está entre 20% e 30%. Contudo, apenas uma parte da viabilidade é calculada na criação do carro. Quando o utilizador selecionar um piloto e o associar ao carro, a viabilidade aumenta entre 50% a 60%, dependendo do piloto escolhido. A potência varia entre os 100 e 600 cavalos.

Cada classe de carro tirando a classe SC pode ser híbrido. Quando é criado um carro híbrido, depois de escolher os atributos associados a essa classe de carro, é escolhida uma potência do motor elétrico (50-150 cavalos) e a fiabilidade sobre um decremento de 10% antes da criação ser finalizada.

2.4. Os Carros – Armazenamento

Apesar dos diferentes tipos de carros, como cada carro terá um modelo único e todos os carros estarem disponíveis para todos os utilizadores, decidimos usar uma coleção para guardar todos os carros no sistema("nome_modelo" -> Carro). Cada carro é clonado e alterado quando um utilizador o seleciona para um campeonato. Depois, é apagado, ficando o original na garagem para outro jogador o selecionar.

2.5. As Corridas – Circuitos e Secções

Os circuitos são constituídos por um nome, as voltas que devem ser efetuadas para completá-lo, as suas secções e a sua distância. Assumimos que cada secção tem comprimento 1, logo a distância é o tamanho da lista de secções.

As secções estão divididas em retas, curvas e chicanes. Para além da especificação do tipo de secção, cada secção também tem um de três graus de dificuldade disponíveis: possível, difícil e impossível. A combinação dos tipos de secção e dos graus de dificuldade afetam as percentagens de um jogador efetuar uma ultrapassagem com sucesso.

2.6. As Corridas – Participantes e Pilotos

Para um participante ser criado, o utilizador corrente terá de selecionar um carro e um piloto. O carro e o piloto selecionados são associados ao participante, assim como o utilizador, para conseguirmos distinguir os participantes e incrementar os pontos de um utilizador no fim de cada campeonato.

Os pilotos são constituídos por um nome do piloto, um critério que varia entre 0 e 1 e avalia a agressividade do piloto, e um critério que varia de maneira semelhante e avalia o quão efetivo este é num ambiente chuvoso comparativamente com clima seco.

2.7. As Corridas – Corridas e Campeonatos

Os campeonatos são compostos por:

- Participantes: um mapa que associa o username de um utilizador ao participante do campeonato-
- Nome da prova: usado para identificar o campeonato.
- Nº corrida: inteiro que começa a 0. É incrementado no final de cada corrida.
- Corridas: Uma coleção de corridas que vão ser usadas no campeonato.

2.8. As Corridas – Lobbys

No nosso sistema, os campeonatos são guardados numa estrutura de dados depois de serem criados pelo Administrador. Contudo, quando um Jogador ou Convidado quiser jogar, terá de criar um lobby usando uma cópia de um dos campeonatos que o Administrador criou. Depois de seleccionar o campeonato que quer usar, o Jogador recebe o número do lobby e a informação do campeonato, indicando isto que o lobby foi aberto. Quando um jogador noutra conta quiser entrar no lobby, pode seleccionar a opção “Juntar Campeonato” do menu principal e usar o número do lobby a que o host tem acesso. Devido a restrições temporais, não incluímos a habilidade de ver todos os Lobbys criados quando a opção “Juntar Campeonato” é seleccionada, então o host teria de comunicar o seu número por outros métodos aos Jogadores.

Quando o host decidir, ele pode começar o campeonato, e simular as corridas uma a uma. Sempre que uma corrida é simulada, os resultados são imprimidos no terminal.

No final, os pontos de todos os participantes que não são convidados são adicionados às suas contas.

3. Camada de Dados

Na camada de dados da nossa aplicação, usamos Data Access Objects(DAOs) para estabelecer uma ligação entre a camada de dados e a camada de negócio. Apesar de guardarmos todos os dados das classes da nossa aplicação exceto a classe **Corrida**(***), as classes criadas com os DAOs são:

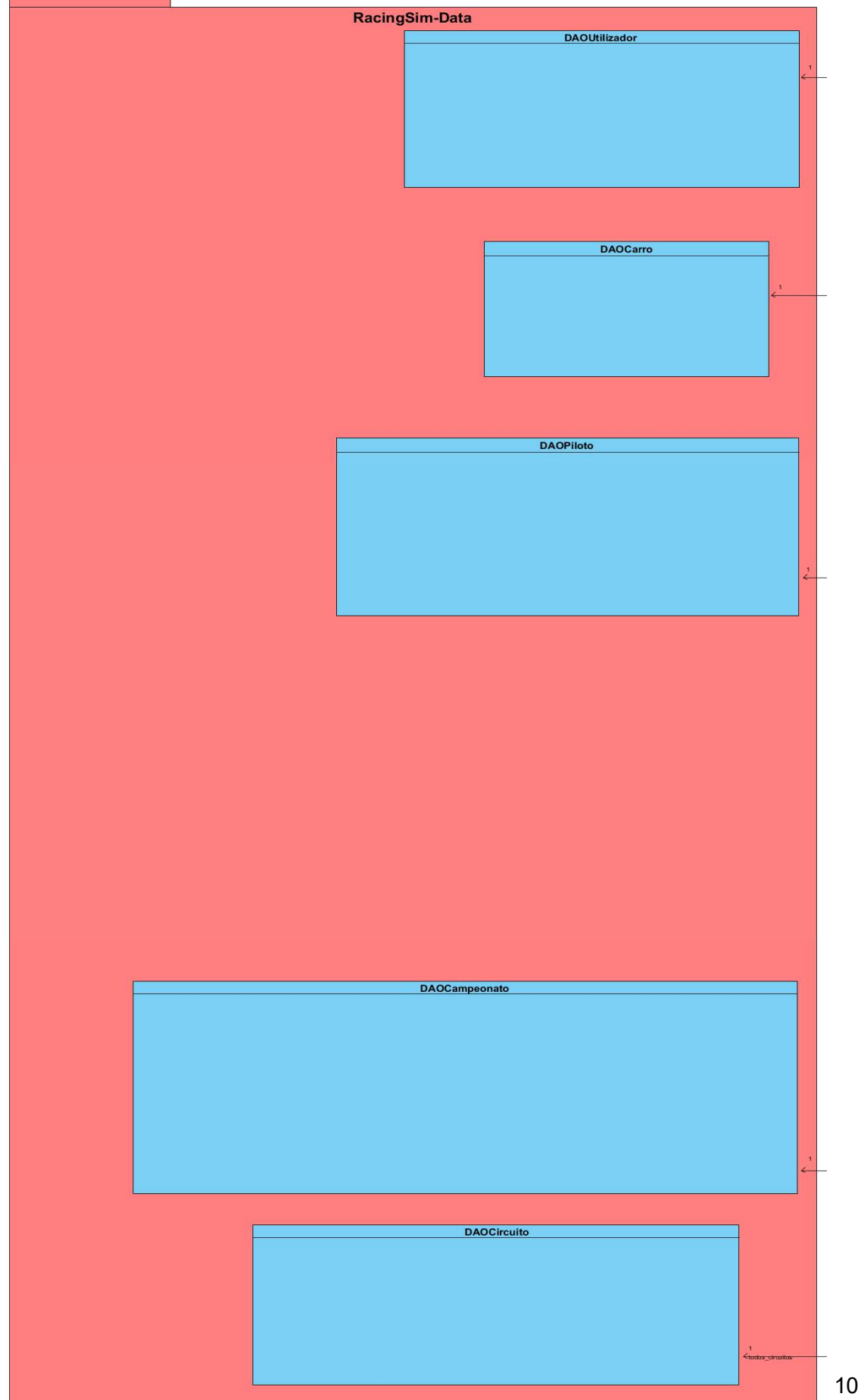
- Campeonato;
- Utilizador;
- Carro;
- Circuito;
- Piloto;

Acabamos por decidir usar DAOs para garantir, entre outros:

- Melhor escalabilidade do nosso código;
- Um maior nível de segurança, visto que existe uma camada de abstração entre as camadas de negócio e de dados;
- Manutenção mais fácil, porque temos todos os dados numa única localização, o que reduz a dificuldade de remover erros ou bugs;
- Apesar de não conseguirmos notar a diferença num trabalho desta escala, a longo prazo também conseguem aumentar o grau de performance da aplicação devido á possibilidade de efetuar queries na informação que está guardada na base de dados.

(***) As corridas têm um carácter temporário na aplicação devido ao facto de serem criadas a partir da lista dos participantes de um lobby e um circuito sempre que um lobby arranca. Depois do lobby terminar, elas são apagadas, juntamente com o lobby. Apenas existem no contexto da função que simula o campeonato.

Agora apresentamos uma imagem da camada de dados do nosso diagrama de classes, contendo os DAOs:



4. Camada de Negócio

Na camada de dados da nossa aplicação, os dados que entram nesta camada são processados, ou seja:

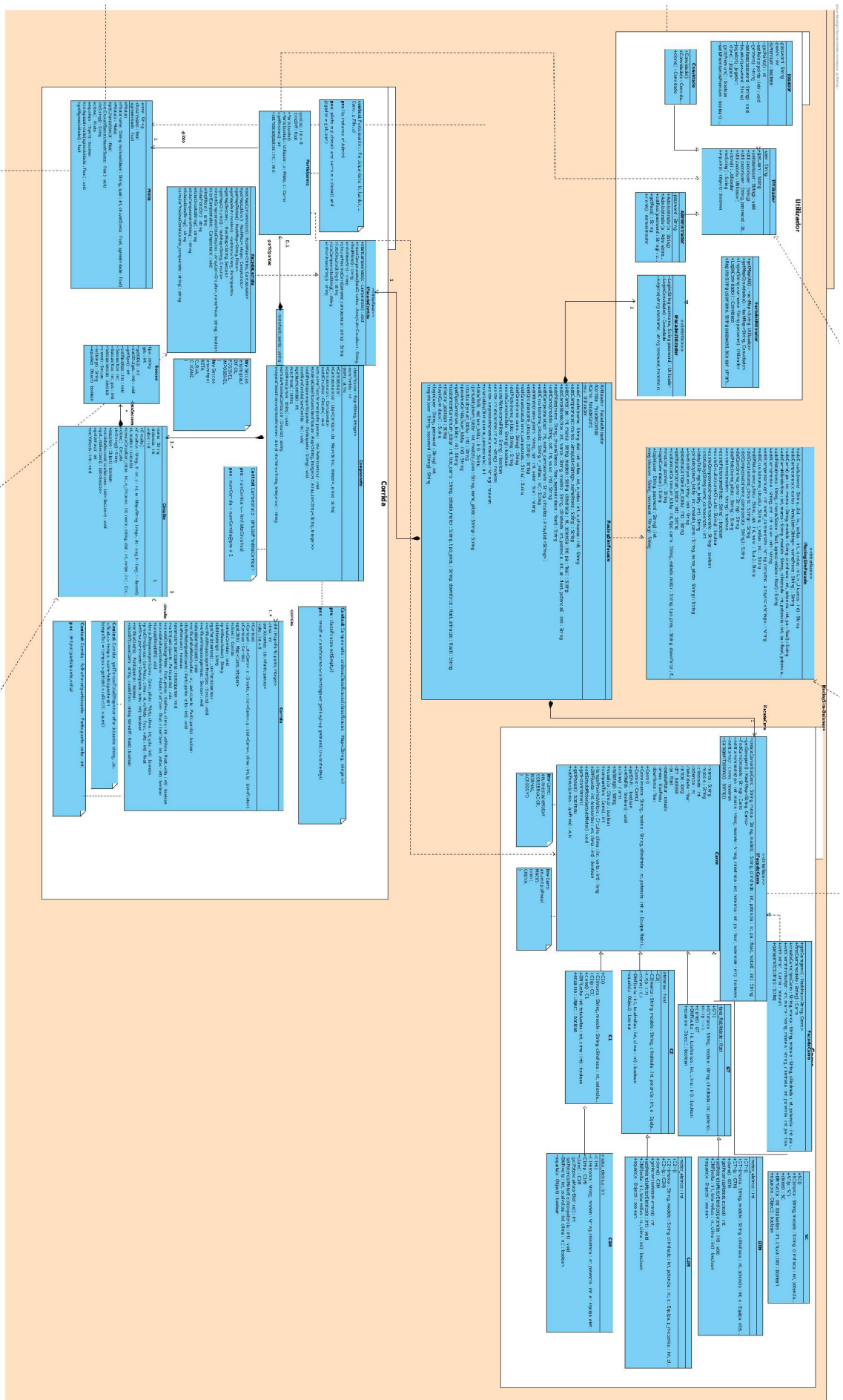
- São efetuados cálculos sobre dados(Ex: o cálculo da fiabilidade de um carro criado pelo administrador);
- A informação é organizada(Ex: as classificações de um campeonato são organizadas pela classe do carro, e depois por pontos);
- Os dados dos utilizadores são avaliados(Ex:o parsing dos valores que um administrador dá para criar um carro);

Existe uma necessidade de minimizar as relações de dependência para obedecer ao paradigma de programação orientada aos objetos. Contudo, isto não foi completamente possível devido a uma falta de tempo na realização do projeto. Logo, algumas das classes e relações poderiam ser reorganizadas de modo a reduzir as relações de dependência.

Usamos três Facades conectadas a uma Facade principal para servir como a classe onde iremos aceder a todas as classes da camada de negócios. Logo, algumas das classes e relações poderiam ser reorganizadas de modo a reduzir as relações de dependência.

De maneira a restringir e simplificar o acesso de um utilizador aos métodos desta camada, usamos Interfaces(3 Interfaces, uma para Package e uma Interface principal ligada à Facade principal).

Agora apresentamos uma imagem da camada de negócio do nosso diagrama de classes, contendo as Facades,as Interfaces e as suas respetivas classes:



5. Camada da Interface do Utilizador/Diagrama Completo

A última etapa da implementação da aplicação inclui criar uma classe que iria tratar de mostrar a informação ao Utilizador e comunicar com a camada de negócios para enviar o seu input e devolver os dados que informam o utilizador do efeito da sua ação. Para isto, desenvolvemos uma classe menu que mostra as opções ao Utilizador e comunica com uma classe TextUI, sendo esta a que dá parse á informação do input e envia-a como argumento á interface do RacingSim, de acordo com a opção escolhida no menu.



Dito isto, agora apresentamos o diagrama de classes atualizado para refletir a nossa implementação de uma maneira mais precisa:



6. Conclusão

Durante a última fase deste projeto, apercebemo-nos que o processo de desenvolvimento de um pedaço de software é algo bastante complicado sem as ferramentas/métodos adequados.

Devido a restrições temporais, juntamente com um grau de inexperiência em termos de conhecimento do processo, tivemos algumas dificuldades a concluir o projeto de maneira a condizer com todos os requisitos que foram especificados nos nossos Use Cases, e alguns dos que foram cumpridos não foram implementados da maneira que esperávamos.

Dito isto também consideramos que, perante o nosso nível de aptidão e aquilo que foi pedido e nós especificamos, conseguimos implementar este simulador de uma maneira relativamente satisfatória.