



Universidade do Minho
Escola de Engenharia
Licenciatura em Engenharia informática

Unidade Curricular de Desenvolvimento de Sistemas Software

Ano Letivo de 2022/2023
Novembro 2022

Trabalho Prático – Entrega Intermédia II

A96547
*Rodrigo José Teixeira
Freitas*



A95076
*Pedro Marcelo Bogas
Oliveira*



A92974
*José dos Santos
Mendes*



A93186
*Eduardo Fernando
Cruz Henriques*



Índice

Índice	1
Introdução	2
Arquitetura Inicial do Sistema	3
Vista dos Use Cases – Diagrama de Componentes	4
Arquitetura do Sistema – Modelação Estrutural	5
Vista Lógica do Sistema – Diagrama de Classes	5
Comportamento e Interações no Sistema – Diagramas de Sequência	11
1. Diagrama – Simulação de uma Corrida do Campeonato	12
2. Diagrama – Atualiza as posições dos jogadores numa corrida	13
3. Diagrama - Verifica se existem utilizadores Premium na corrida	14
4. Diagrama - Verifica ultrapassagem para utilizadores <i>Premium</i>	15
5. Diagrama – Verifica ultrapassagem para jogadores não- <i>Premium</i>	16
6. Diagrama – Tentativa de ultrapassagem	17
7. Diagrama – Avançar uma posição na corrida	18
8. Diagrama - Verifica se o condutor se despenhou na secção do circuito	19
9. Diagrama – Verificação de falha no motor no final de uma volta	20
10. Diagrama – Participante que se despistou vai para o fim da classificação	21
11. Diagrama – Imprimir resultados de uma Corrida	22
12. Diagrama – Imprimir classificações de um Campeonato	23
Conclusão	24

Introdução

Nesta fase do nosso trabalho prático de Desenvolvimento de Sistemas de Software, procuramos criar uma arquitetura satisfatória de um sistema de simulação de corridas.

Com este relatório tentamos dar uma descrição detalhada da arquitetura do sistema que descrevemos brevemente na fase anterior e iremos construir na fase seguinte, usando estes modelos como uma substancialização dos modelos da fase anterior e uma base para a próxima fase, em que iremos construir a aplicação baseando-nos neles.

Pensamos que as seguintes descrições da conceção do sistema nos irão providenciar um desenvolvimento mais conciso e eficiente da aplicação de simulação de campeonatos de automobilismo.

Arquitetura Inicial do Sistema

Nesta etapa da criação do modelo da arquitetura do sistema iremos desenvolver os subsistemas(componentes) principais que vão ser usados pela nossa aplicação. Concluímos que irão existir 4 componentes que irão constituir a nossa aplicação:

- o **componente Utilizador**, onde serão implementados os diferentes tipos de utilizador do sistema, e os seus componentes de login, assim como as suas funções no sistema;
- o **componente Carros**, que irá guardar informação sobre os carros que serão usados no sistema e os diferentes aspetos de cada carro, que terão depois impacto no componente da Corrida(simulações). Existe um componente Carros, mas não existe um componente Pilotos devido á complexidade da classe Carro comparativamente á classe Piloto.
A enorme variabilidade do tipo de carro e as suas consequências para a simulação é um fator que consideramos importante e que originou a criação deste subsistema.
- o **componente Corrida**, que armazena as classes relacionadas com a estrutura dos componentes principais do sistema de simulação(um campeonato e os seus participantes, circuitos, corridas, etc.) Escolhemos incluir a classe Piloto neste componente, contudo também podia estar no subcomponente Carros.
Pensamos que desta maneira iríamos reduzir o número de dependências no projeto, contudo admitimos que, caso a classe Piloto tivesse um maior grau de complexidade(á semelhança da classe Carro) devia ser devidamente colocada no componente Carros e renomeá-lo para Setup ou Loadout.
- o **componente Sistema**, que engloba os componentes anteriores e servirá como meio de unir os métodos da nossa aplicação de modo a garantir que existe uma Interface que garanta o acesso fácil á aplicação como um todo.

Vista dos Use Cases – Diagrama de Componentes

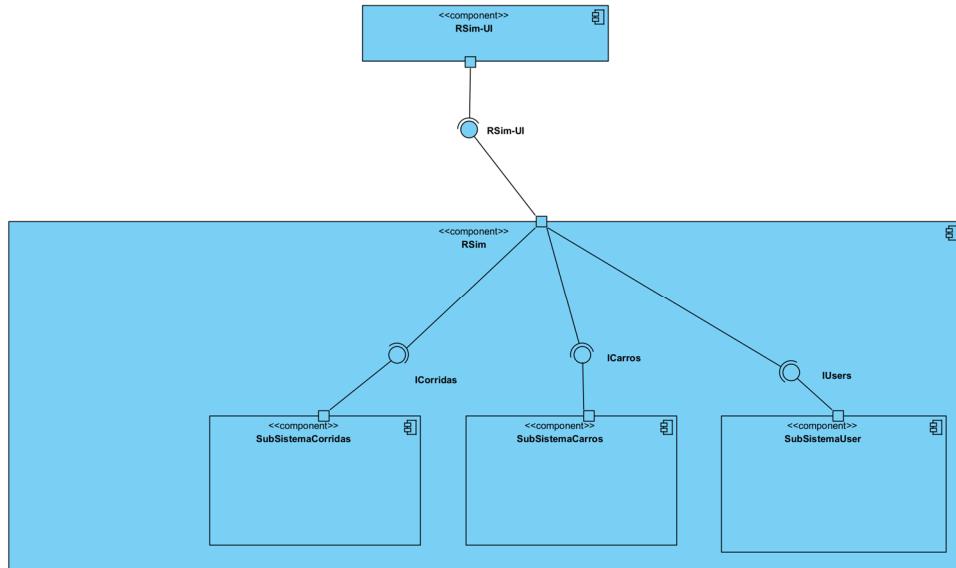


Ilustração 1 – Diagrama de Componentes

A Interface do Sistema de Simulação irá servir como a Interface principal que terá métodos de todos os subsistemas. Iremos conseguir ter acesso a métodos de todos os packages do sistema de simulação de corridas.

Cada Interface de cada componente(subsistema) terá acesso aos métodos definidos nesse package, existindo desta maneira um acesso mais controlado a cada componente do sistema, quando exista necessidade de acesso.

Arquitetura do Sistema – Modelação Estrutural

Agora iremos avançar para a criação das classes/atributos que o nosso sistema deverá ter e utilizar, assim como as suas funções, restrições, tipos e privilégios.

Também iremos descrever o comportamento de componentes do Sistema, para facilitar a construção dos métodos mais complexos que iremos implementar depois.

Vista Lógica do Sistema – Diagrama de Classes

O diagrama de classes seguinte tem como objetivo apresentar/elaborar alguns dos objetivos apresentados no tópico anterior. O aspeto do comportamento das classes abordadas neste diagrama será aprofundado na secção a seguir, dos diagramas de sequência. Também vamos referir os aspetos que não utilizamos do código legado que nos foi fornecido.

Também usamos a linguagem OCL para criar invariantes de modo a regularizar o comportamento de vários métodos das classes.

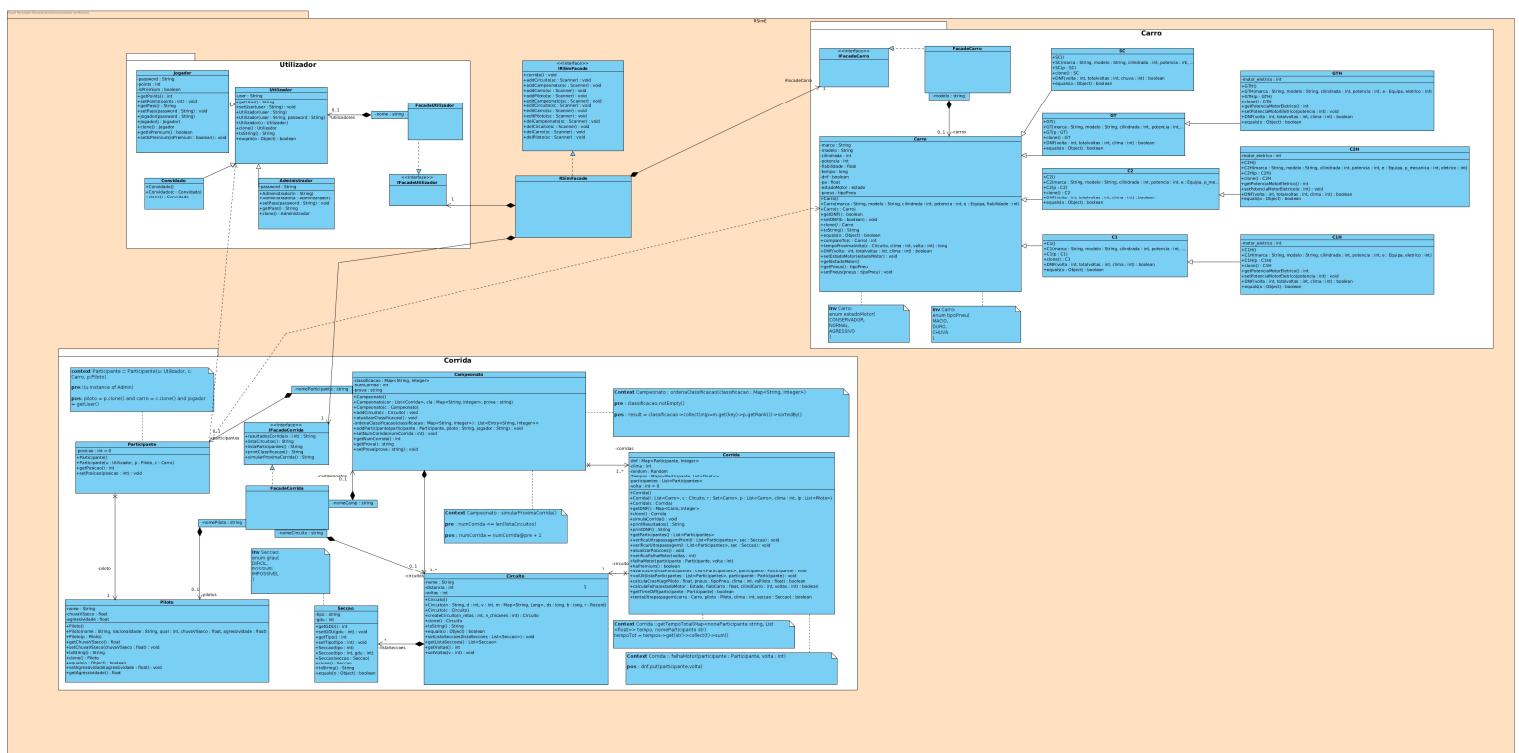


Ilustração 3 – Diagrama de Classes. É possível ver os packages a criar na próxima fase do projeto e as diferentes facades e interfaces, assim como as dependências da classe Participante

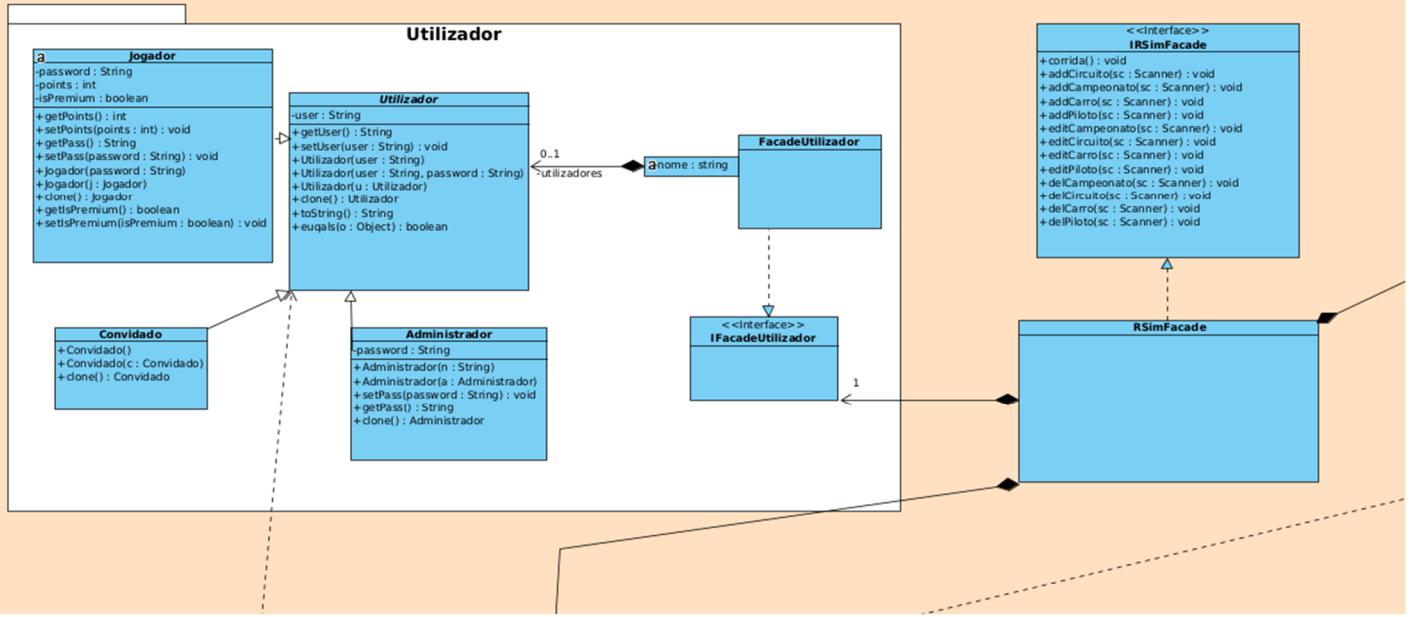


Ilustração 3 – Package Utilizador

Em relação ao código legado:

- Retiramos o conceito de apostas do simulador. Estava anteriormente associado ao Utilizador.
- Retiramos o conceito de ver o histórico de corridas de cada Utilizador do simulador.
- Foram criadas as subclasses Administrador, Convidado e Jogador.
- Introduzida a possibilidade de um Jogador ser *premium* ou não.
- Um Utilizador que não seja Convidado é guardado no sistema com associação a uma palavra-passe.

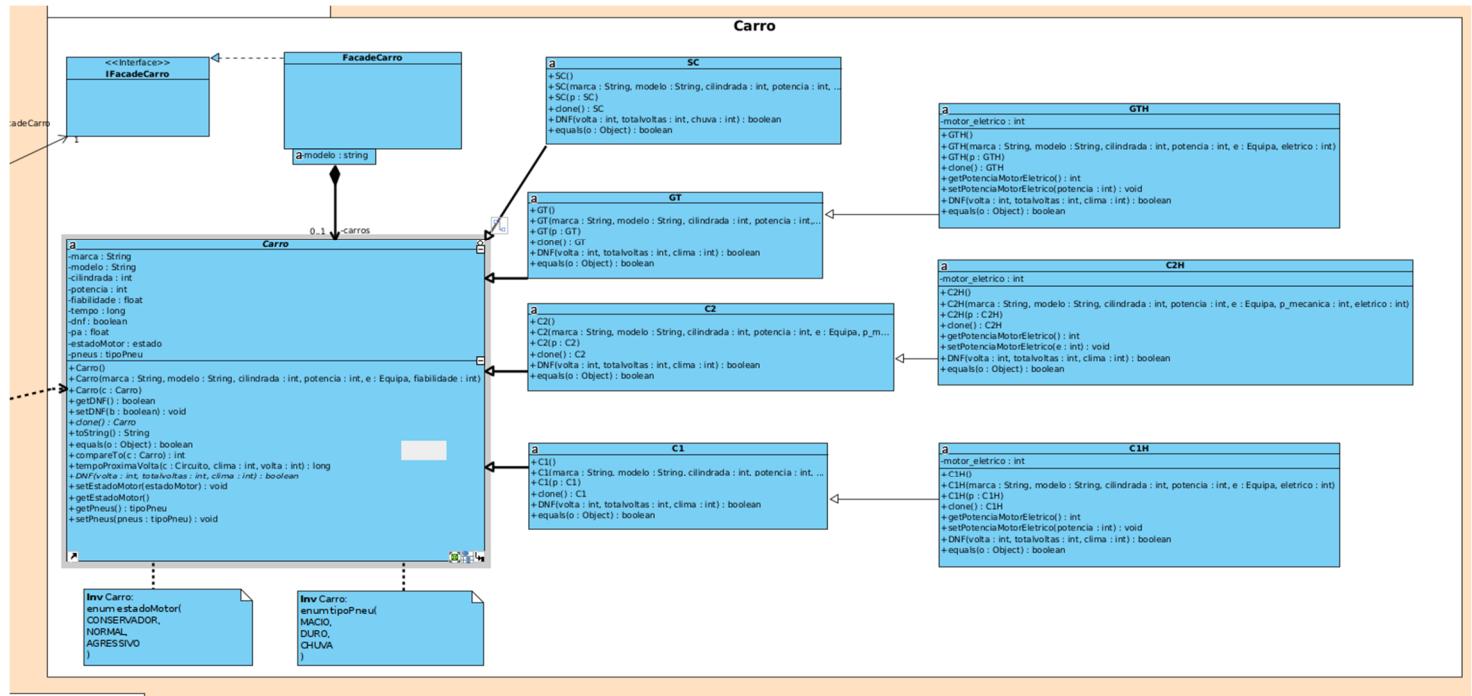


Ilustração 3 – Package Carro

Em relação ao código legado:

- Não abordamos o aspeto de um carro estar associado a uma equipa
- Introduzimos atributos para o tipo de pneu, o estado do motor e o perfil aerodinâmico de cada carro. Tal aspeto não estava presente no código legado.
- O atributo *did not finish(dnf)* foi atribuído ao participante(no package Corrida), e não ao carro.

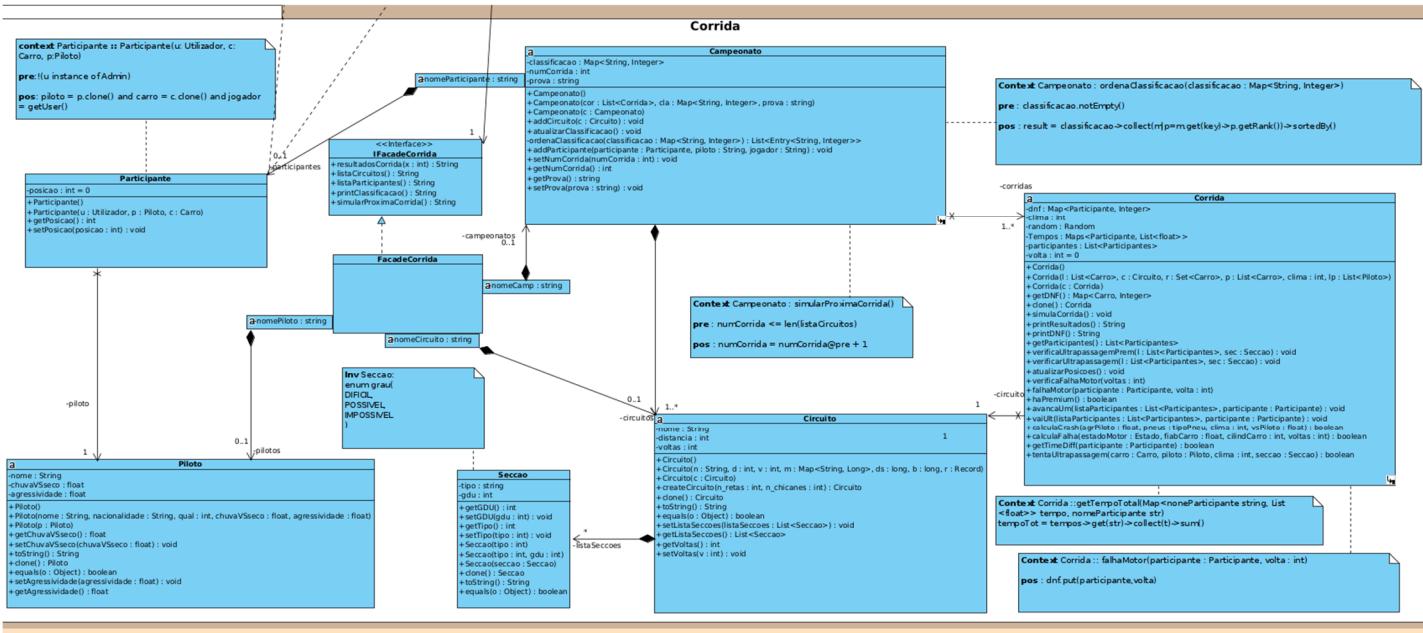


Ilustração 3 – Package Corrida

Em relação ao código legado:

- Baseamo-nos na classe Corrida do código legado para criar a classe Circuito.
 - O conceito de Recordes foi eliminado do trabalho.
 - Não é possível agendar uma corrida, porque esse aspeto não foi elaborado no trabalho prático presente.
 - As nossas classificações contêm carros híbridos e não híbridos, não existem classificações separadas para os dois tipos de carro.
 - Não consideramos necessário guardar um histórico das corridas realizadas por qualquer Jogador/Convidado.

Descrição das classes

Agora iremos fazer uma breve descrição das classes principais do diagrama que apresentamos, assim como o seu papel no funcionamento do simulador:

- **Utilizador:** A classe que identifica cada indivíduo que acede ao sistema, para conseguir interagir com as classes restantes. Existem 3 tipos de Utilizador: Administrador, Jogador(que poderá ser Premium ou não) e Convidado. Tanto os Jogadores como os Convidados irão conseguir participar num campeonato, e os pontos do Convidado não serão armazenados no seu perfil quando um campeonato termina. Caso um utilizador queira entrar no simulador com um username que já está associado a um Jogador ou Admin existente, ir-lhe-á ser pedida uma palavra-passe. Caso contrário pode registar o seu user ou entrar como convidado. O facto de um Jogador ser Premium ou não apenas afeta os cálculos durante a simulação em que participa.
- **Carro:** Representará os carros que o sistema usa nas simulações. Está dividida em diversas categorias que irão influenciar o desempenho do veículo em certos cenários de simulação. Os carros poderão ser criados, alterados e removidos do sistema(apesar disto estar restrito ao Utilizador que é Administrador).
- **Piloto:** A classe com a informação dos pilotos que o sistema usa nas simulações. Um piloto tem diversos atributos(agressividade e desempenho na chuva/tempo seco) que influenciarão o seu desempenho nas inúmeras situações que iram-lhe ser apresentadas ao longo de uma corrida.
- **Secção:** Nesta classe irão ser representados os segmentos de um circuito, onde, durante a simulação irá ser calculado se um participante irá ter um acidente ou se é capaz de ultrapassar o adversário a sua frente (dependendo de um conjunto de fatores). Esta classe é constituída pelo tipo de secção (reta, chicana, curva) e pelo grau de dificuldade da mesma (possível, difícil, impossível).

- **Círcuito:** Representa o circuito onde irão ser feitas as simulações e este é constituído pelo nome do circuito, a distância de cada volta, o número de voltas que a corrida irá ter.
- **Corrida:** Esta vai ser a classe que irá lidar com as simulações de cada corrida. Vai ter um mapa de participantes que não conseguirão acabar a corrida, um clima(o estado meteorológico ativo durante a simulação), um valor aleatório que será invocado sempre que um evento acontece(por exemplo, se uma ultrapassagem ou um despenho ocorre), uma lista que conecta cada participante a uma lista com os tempos de cada volta. Também irá conter a lista de participantes e a volta corrente da simulação.
- **Participante:** Uma classe que é criada a partir da união de um Utilizador com o seu “setup”(um carro e um piloto à sua escolha) e a posição atual no Campeonato em que decidiu participar com ele. É de notar que um podem existir vários objetos Participante a partir de um único Utilizador, visto que ele pode participar em vários Campeonatos ao mesmo tempo. No final de cada Campeonato, os objetos Participante associados a este são removidos do sistema.
- **Campeonato:** A classe que conecta os Participantes às Corridas, para fazerem parte das simulações. É composto por um mapa que interliga os participantes á sua classificação atual no campeonato, o número da corrida atual e o nome da prova(ex:”Campeonato Uminho”).

Comportamento e Interações no Sistema – Diagramas de Sequência

Os diagramas de sequência descrevem, de modo sucinto e simplificado, as interações que as classes deste simulador de corridas de automóveis têm entre si, assim como a ordem de interação que os vários objetos relacionados com um método têm quando esse mesmo método é invocado, fazendo com que o comportamento destes seja previsível e simplificado quando avançarmos para a fase da implementação concreta deste projeto em código.

Devido ao grande volume de métodos que estão associados a um sistema desta escala, e à simplicidade da maioria dos métodos que não executam tarefas complexas e por isso serão facilmente implementados, decidimos criar os diagramas que estão relacionados á parte com o maior grau de dificuldade de implementação: o processo de simulação de um campeonato do início ao fim.

Os seguintes diagramas explicam o comportamento dos métodos principais do sistema relacionados com os aspetos de simulação, abordando todos os componentes deste(as corridas dos campeonatos, a simulação de cada corrida tendo em conta o circuito e as suas secções, o clima, as escolhas de piloto/carro de cada participante, etc.).

1. Diagrama – Simulação de uma Corrida do Campeonato

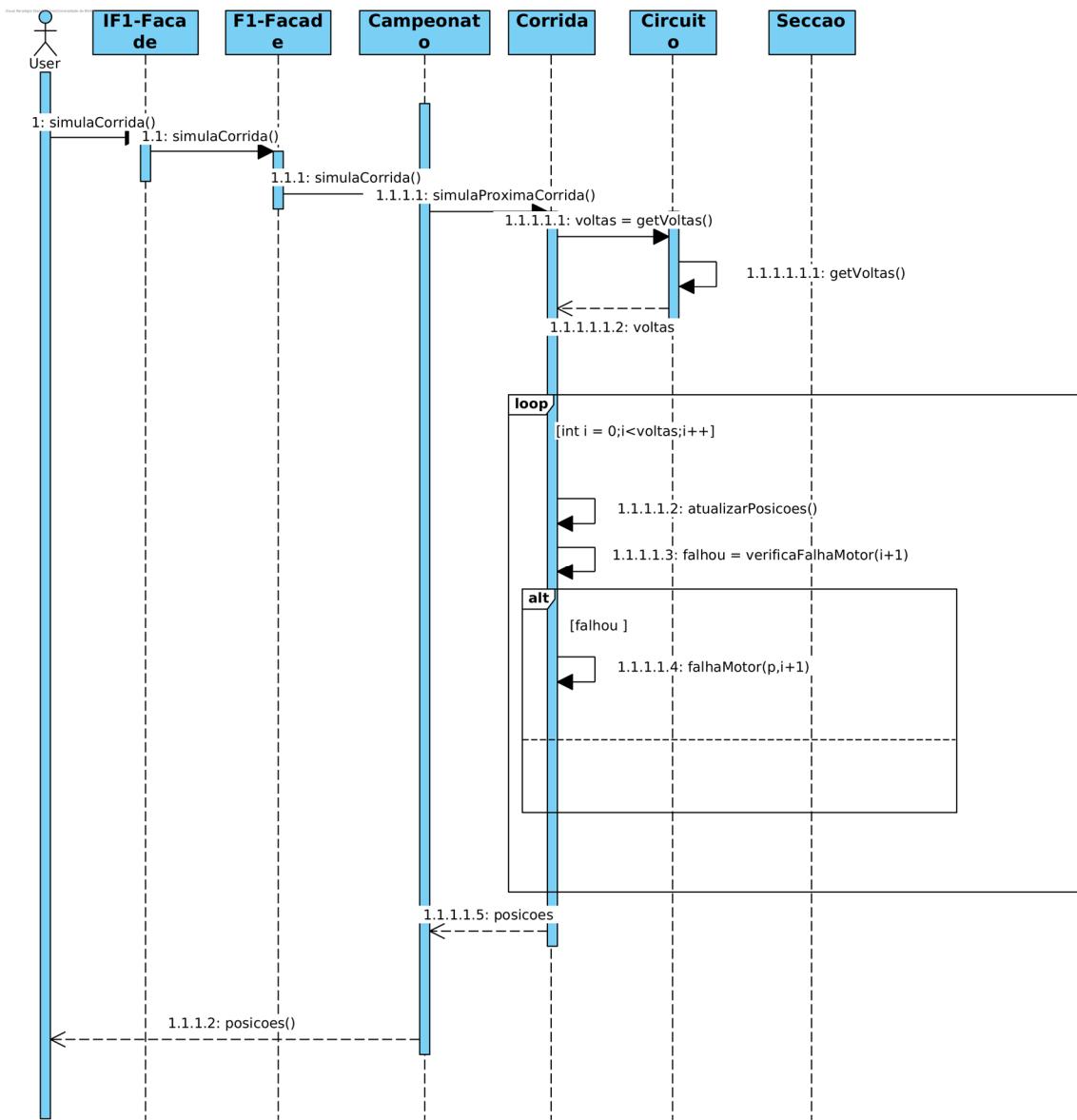


Ilustração 3 – Diagrama de Sequência da função simularCorrida

2. Diagrama – Atualiza as posições dos jogadores numa corrida

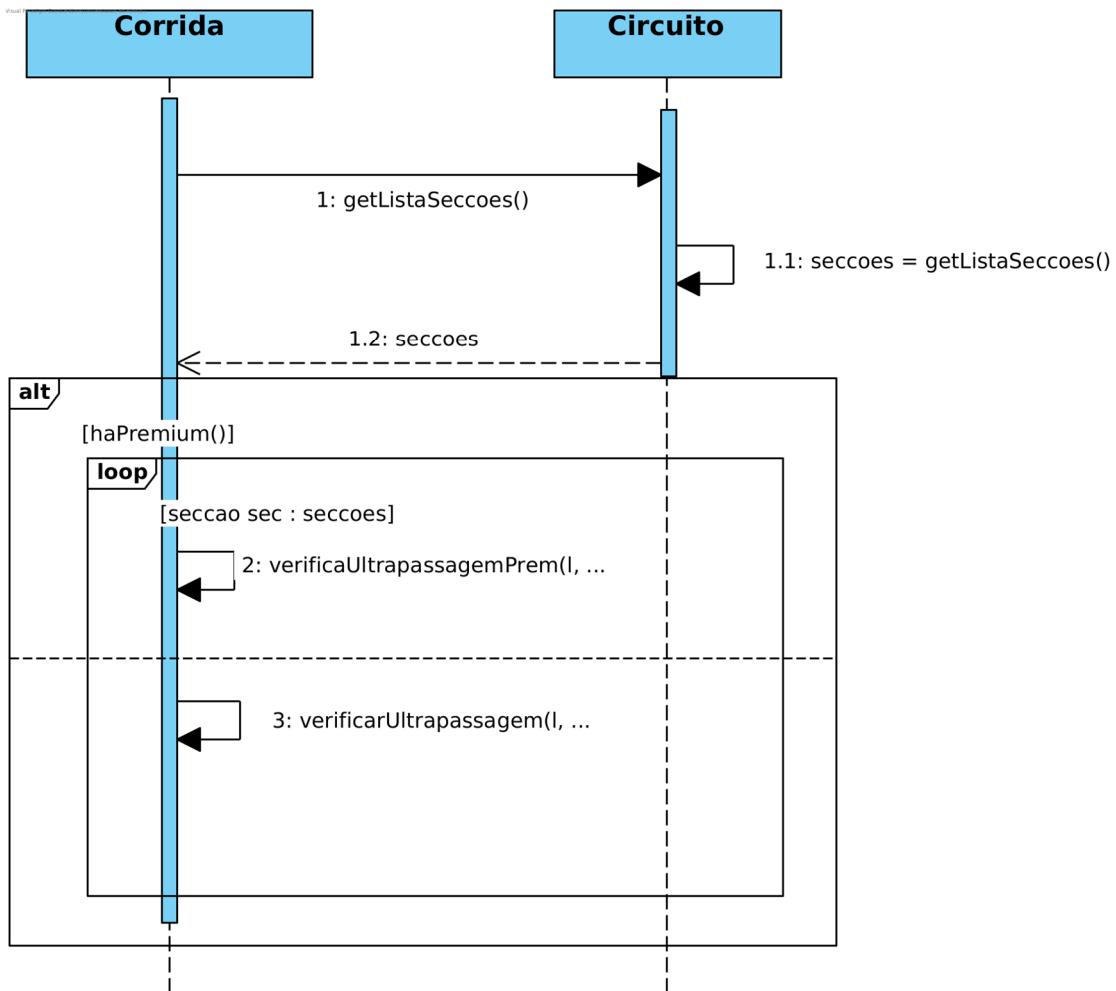


Ilustração 4 – Diagrama de Sequência da função atualizarPosicoes

3. Diagrama - Verifica se existem utilizadores Premium na corrida

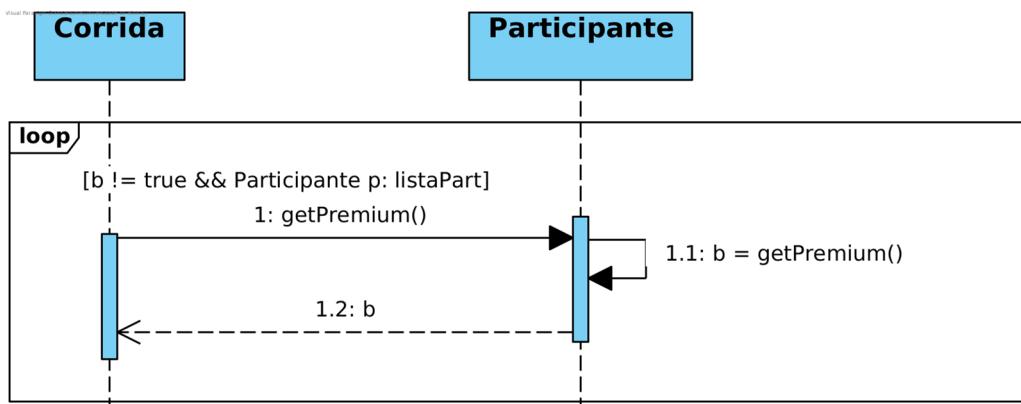


Ilustração 5 – Diagrama de Sequência da função haPremium

4. Diagrama - Verifica ultrapassagem para utilizadores

Premium

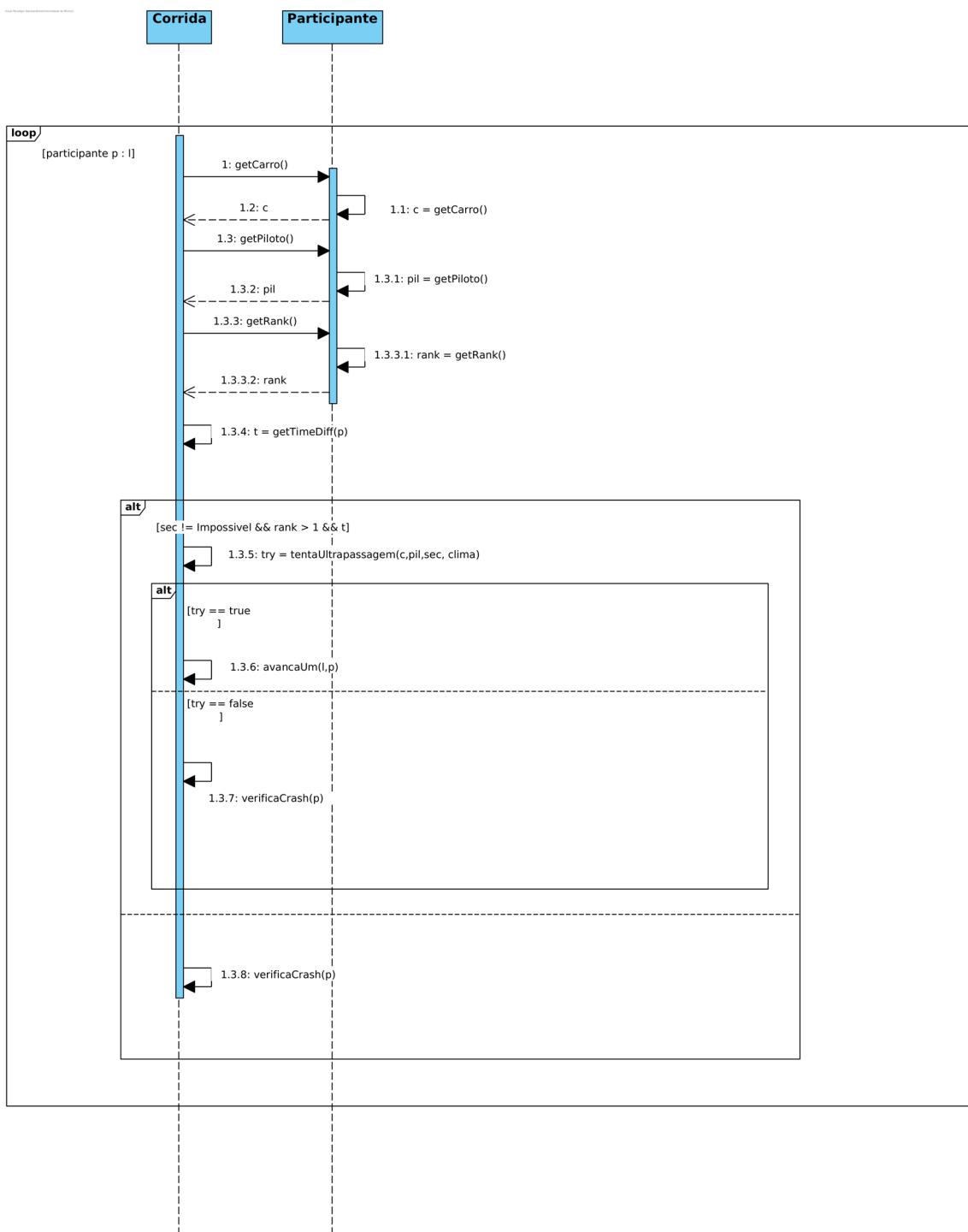


Ilustração 5 – Diagrama de Sequência da função `verificaUltrapassagemPrem`

5. Diagrama – Verifica ultrapassagem para jogadores não-Premium

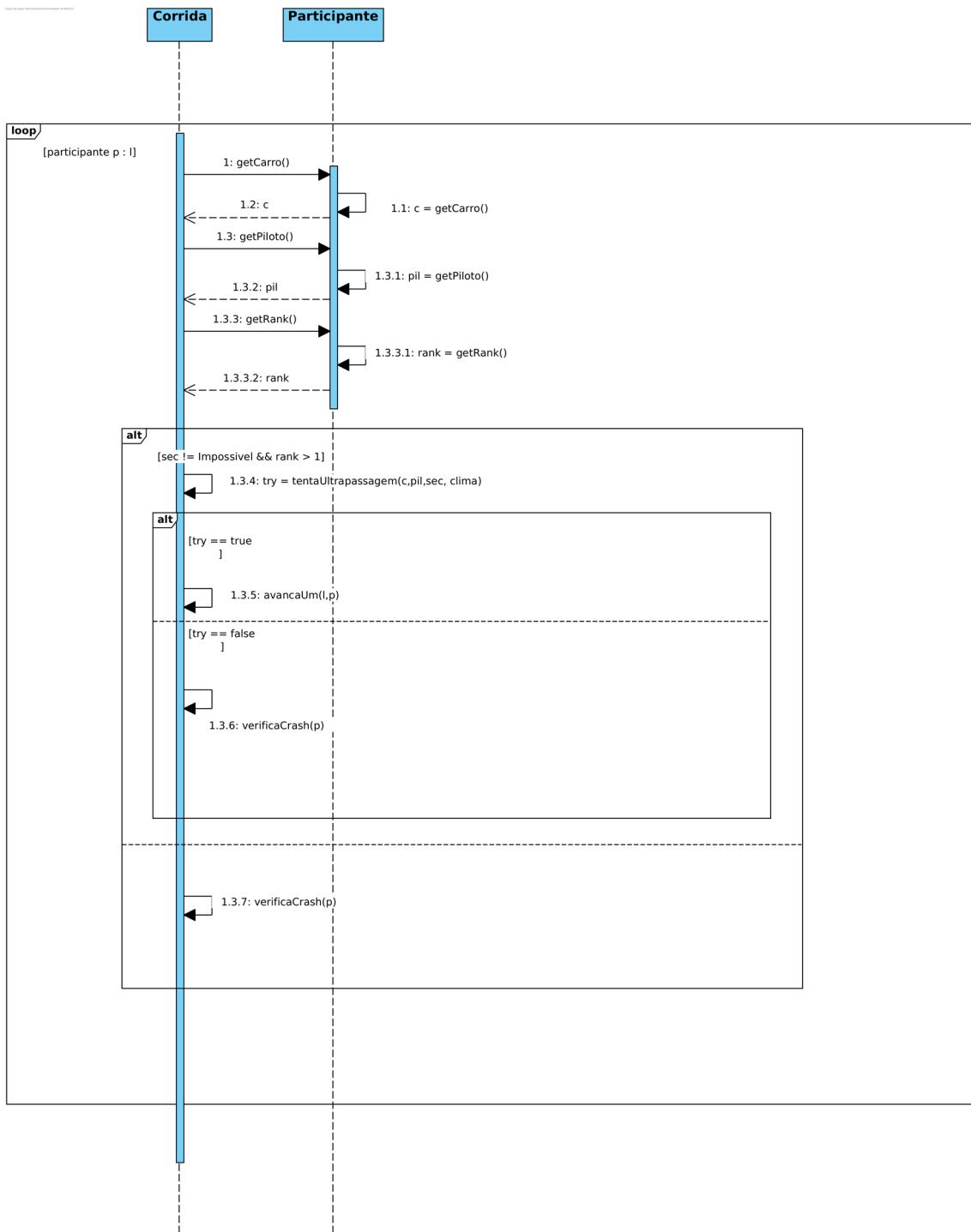


Ilustração 5 – Diagrama de Sequência da função verificaUltrapassagem

6. Diagrama – Tentativa de ultrapassagem

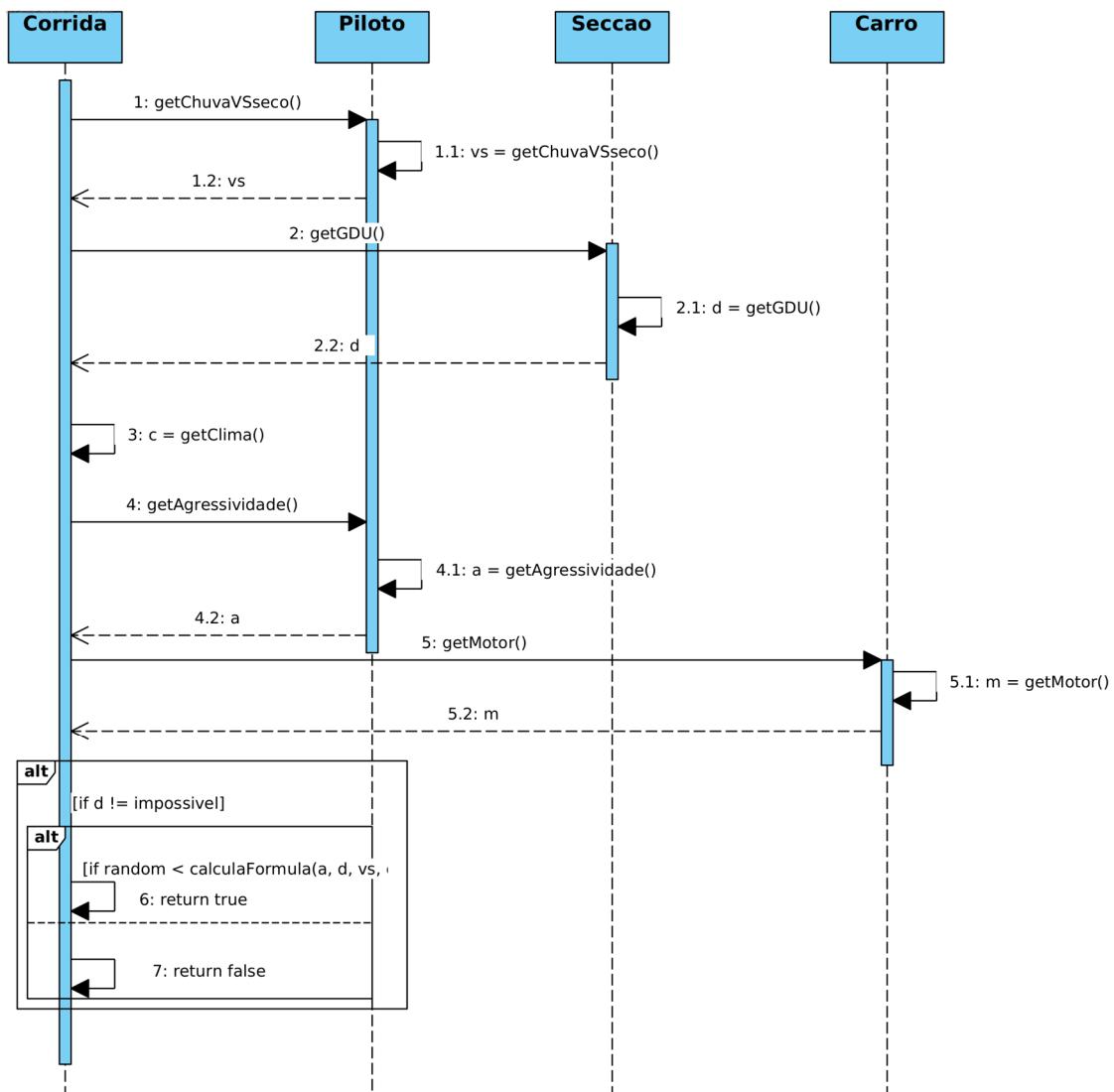


Ilustração 5 – Diagrama de Sequência da função `tentaUltrapassagem`

7. Diagrama – Avançar uma posição na corrida

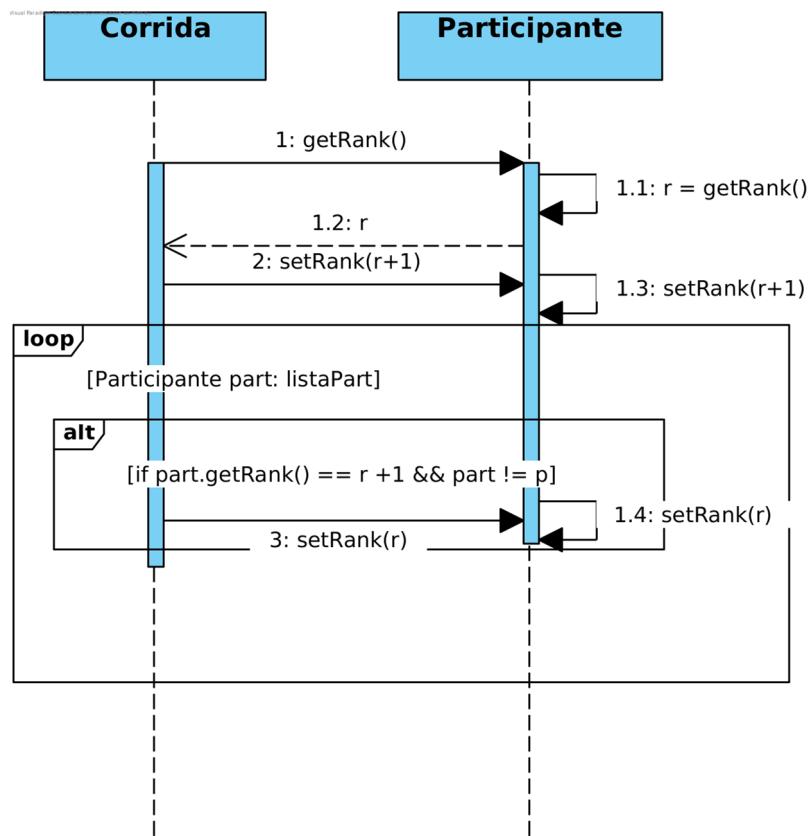


Ilustração 5 – Diagrama de Sequência da função avancaUm

8. Diagrama - Verifica se o condutor se despenhou na secção do circuito

Ilustração 5 – Diagrama de Sequência da função avancaUm

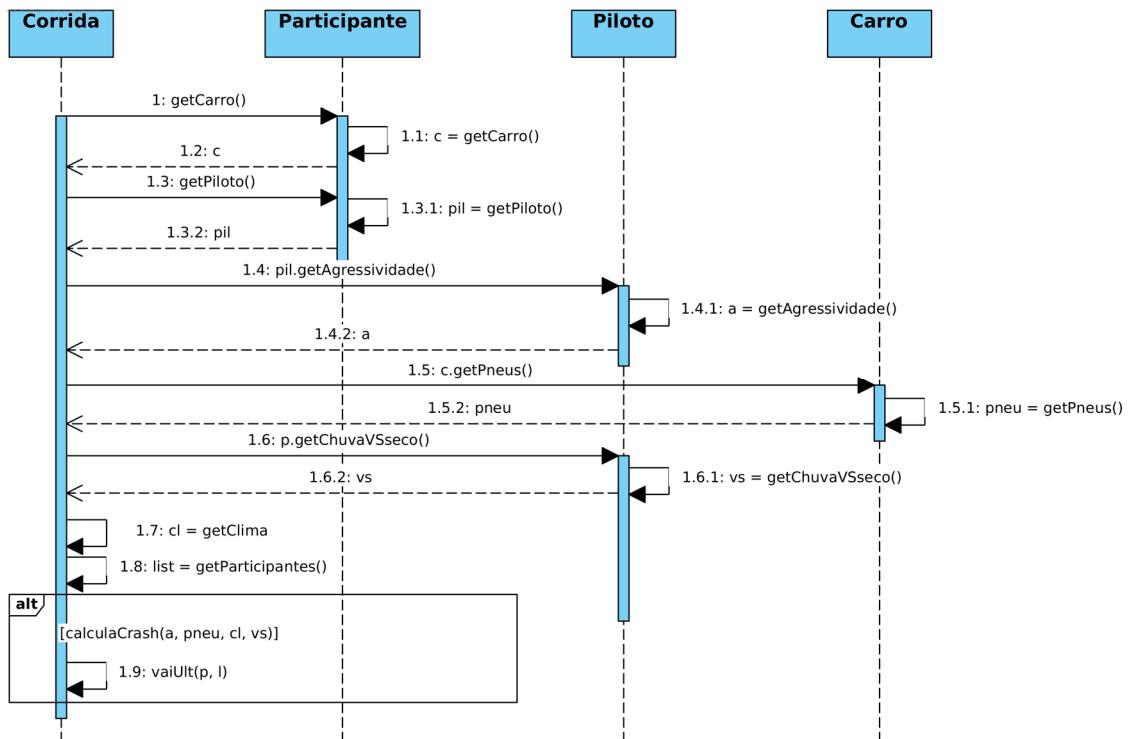


Ilustração 5 – Diagrama de Sequência da função verificaCrash

9. Diagrama – Verificação de falha no motor no final de uma volta

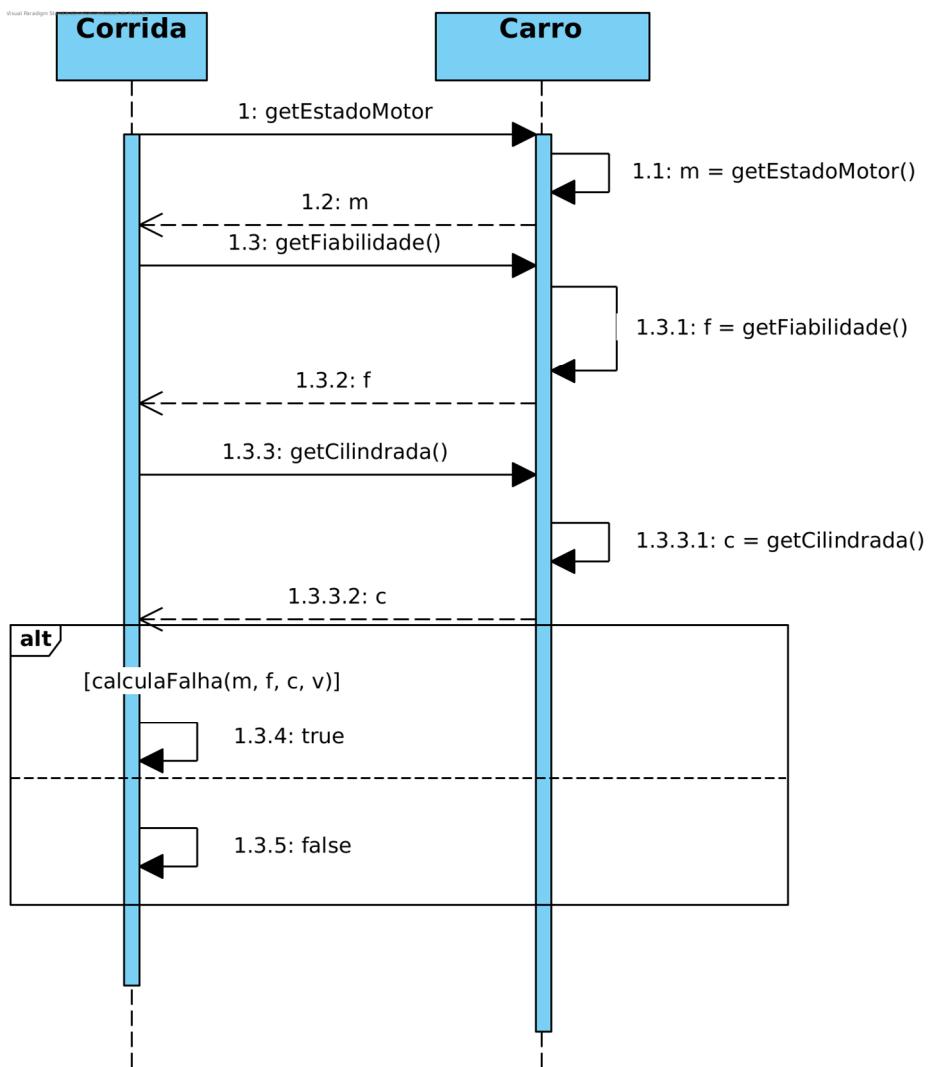


Ilustração 5 – Diagrama de Sequência da função verificaFalhaMotor

10. Diagrama – Participante que se despistou vai para o fim da classificação

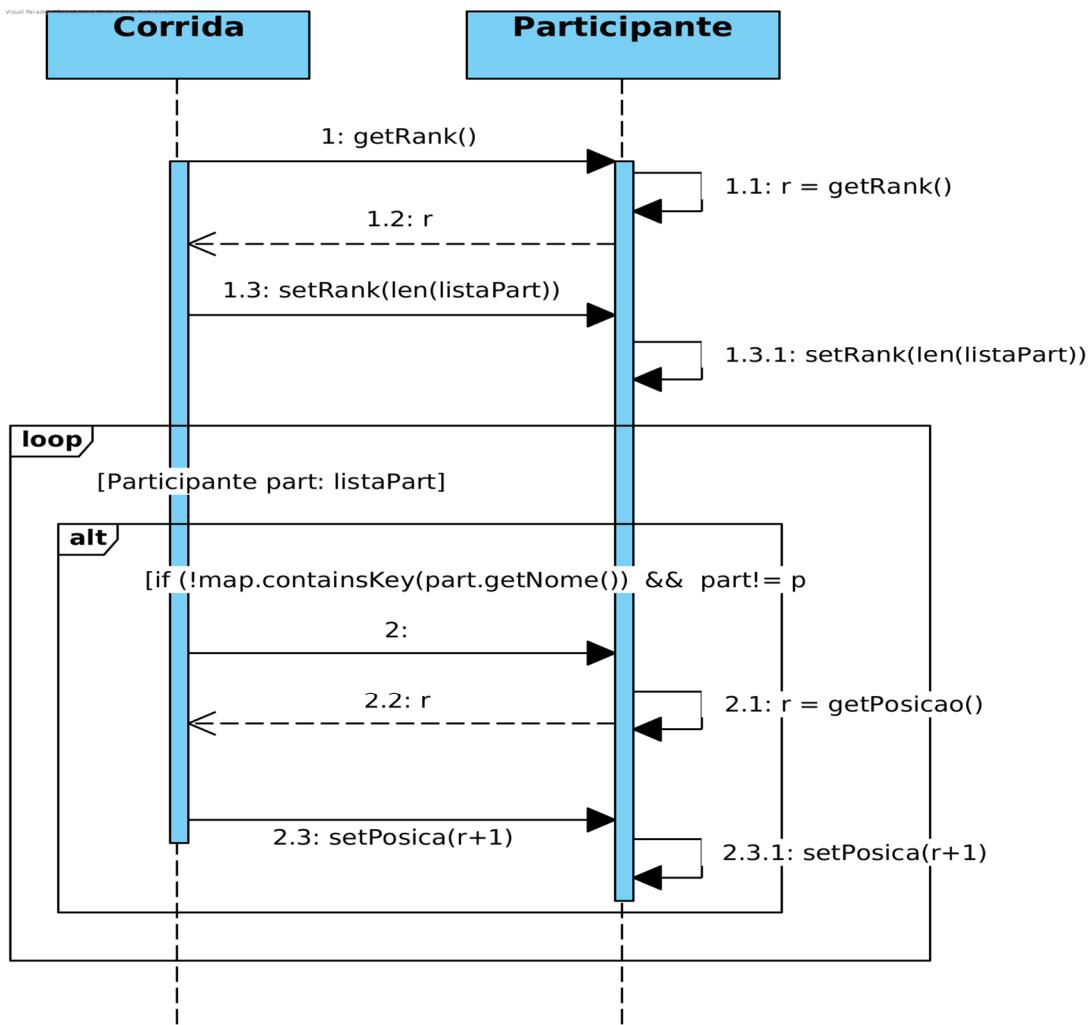


Ilustração 5 – Diagrama de Sequência da função vaiUlt

11. Diagrama – Imprimir resultados de uma Corrida

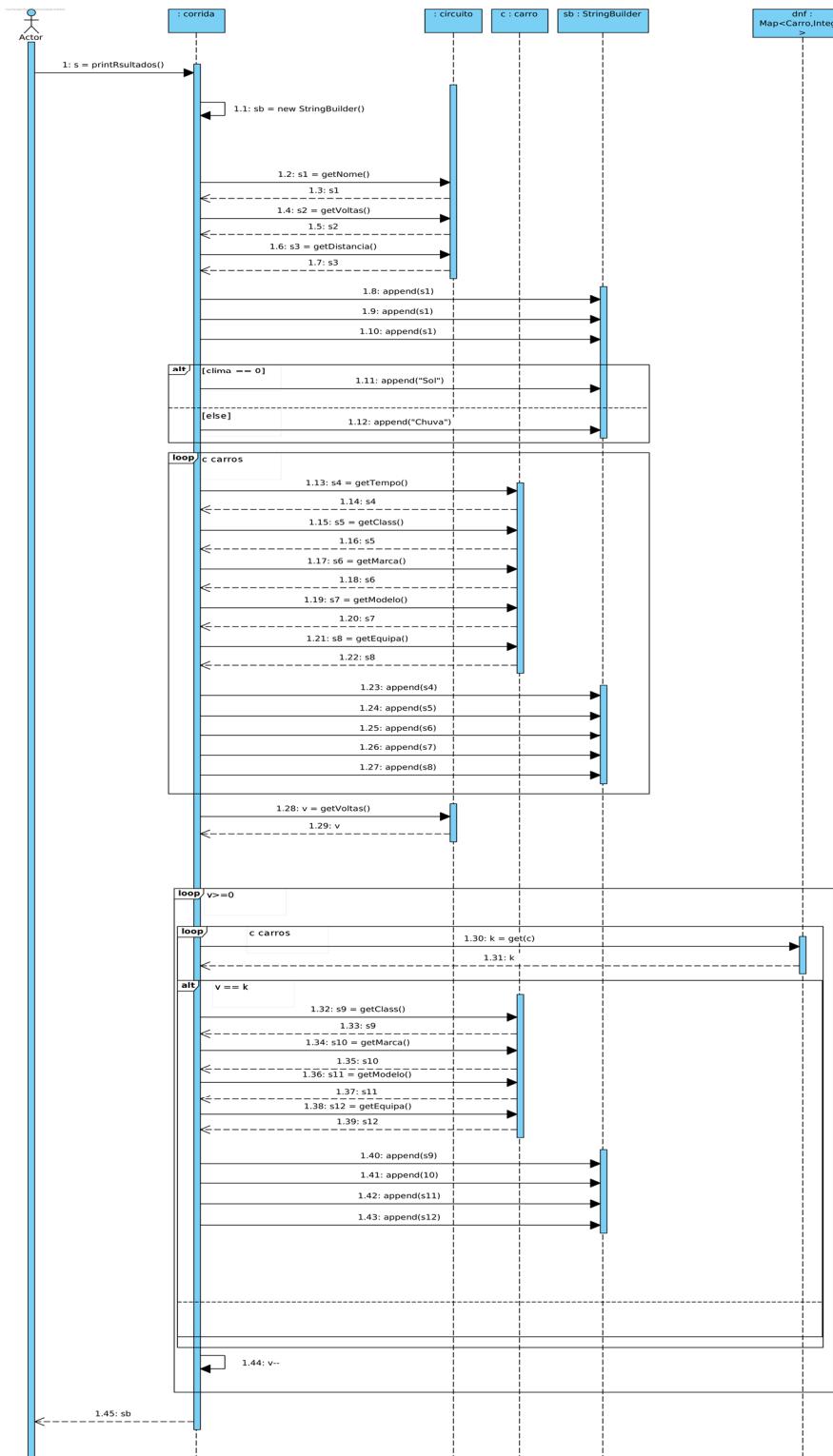


Ilustração 5 – Diagrama de Sequência da função verificaFalhaMotor

12. Diagrama – Imprimir classificações de um Campeonato

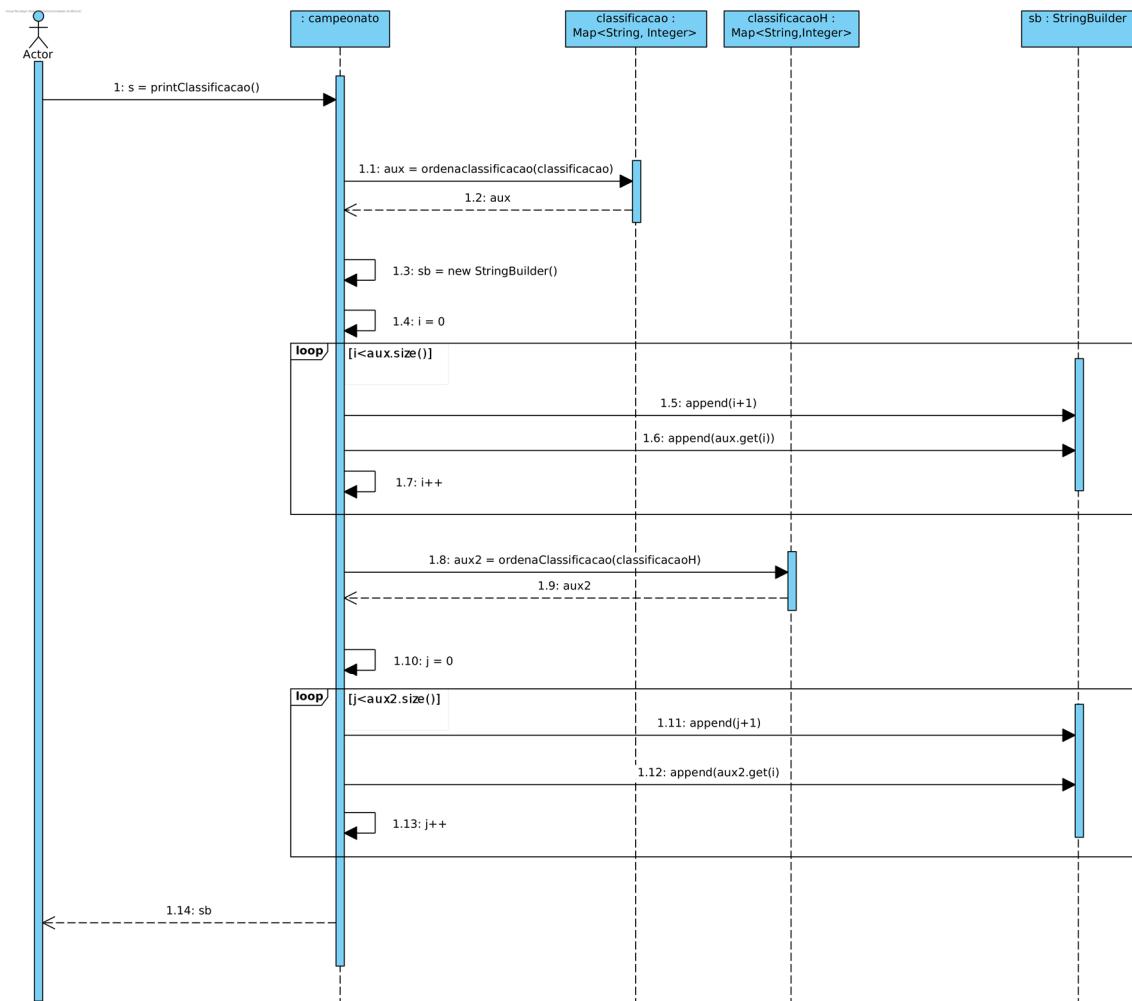


Ilustração 5 – Diagrama de Sequência da função imprimirClassificacoes

Conclusão

Com a elaboração dos modelos acima referidos, pensamos ter abordado todos os pontos chave para o desenvolvimento essencial da aplicação, tendo sempre em atenção as opiniões dos elementos do grupo e aos conselhos e dicas dadas pelos professores da disciplina.

Consideramos que a análise do código legado dado e a criação destes diagramas forneceu-nos a reconsiderar a nossa abordagem a aspetos deste projeto que não teriam sido realizados com sucesso sem este processo de pensamento.

Graças aos modelos realizados nesta fase, a arquitetura do sistema está devidamente estabelecida e temos um caminho concreto para relizar a fase final deste projeto.