António Alberto (114622), Pedro Pinto (115304), Luís Godinho (112959), Marcelo Júnior (115646)
Turma P1, v 2022-11-16.

RELATÓRIO LAB-4

Modelos de comportamentos

4.2 - Representar interações de alto-nível com diagramas de sequência

4.2.1. - Interação a nível de sistema de pagamentos online

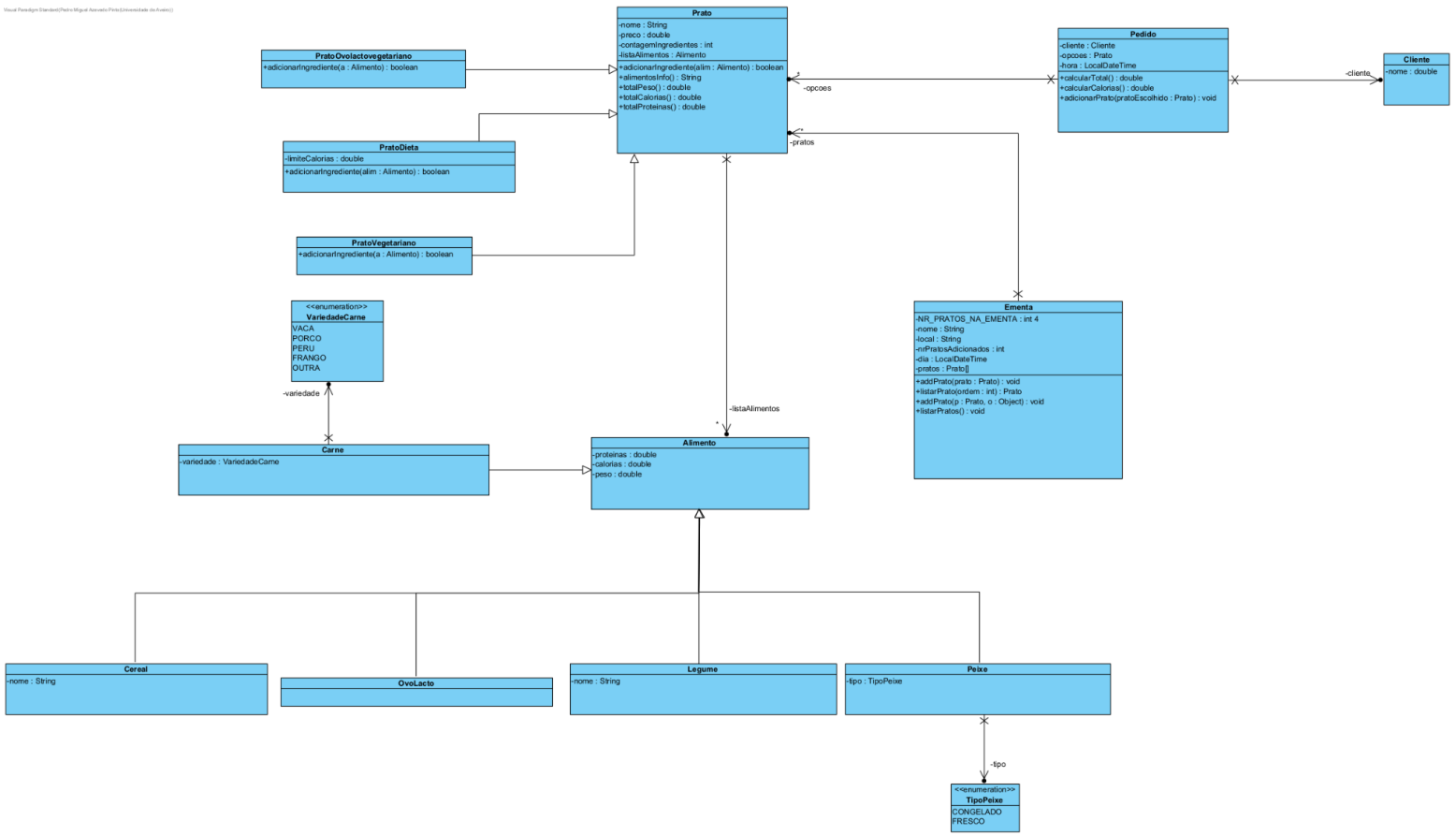
Uma API é um conjunto de regras que explicam como os computadores ou aplicações se comunicam uns com os outros. As APIs garantem segurança nas transações porque a sua posição intermédia facilita a funcionalidade entre dois sistemas distintos. Com uma API de processamento de pagamento, os clientes podem inserir os detalhes de pagamento no frontend de uma aplicação de comércio virtual, e assim, o processador de pagamento não necessita de acesso para a conta bancária do cliente. A API cria um token único para cada transação, e adiciona-o no servidor da API.

A API da EasyPay permite pedir dados de pagamento, receber notificações de pagamento e enviar notificações de pagamento. Um pagamento frequente na easypay permite que o cliente faça uma nova compra sem ter que introduzir novamente os seus dados de pagamento, facilitando a experiência de compra. Alguns exemplos de pagamento frequente existentes na easypay são a Referência Multibanco, MB Way, Cartões Visa e Mastercard e Débito Direto SEPA.

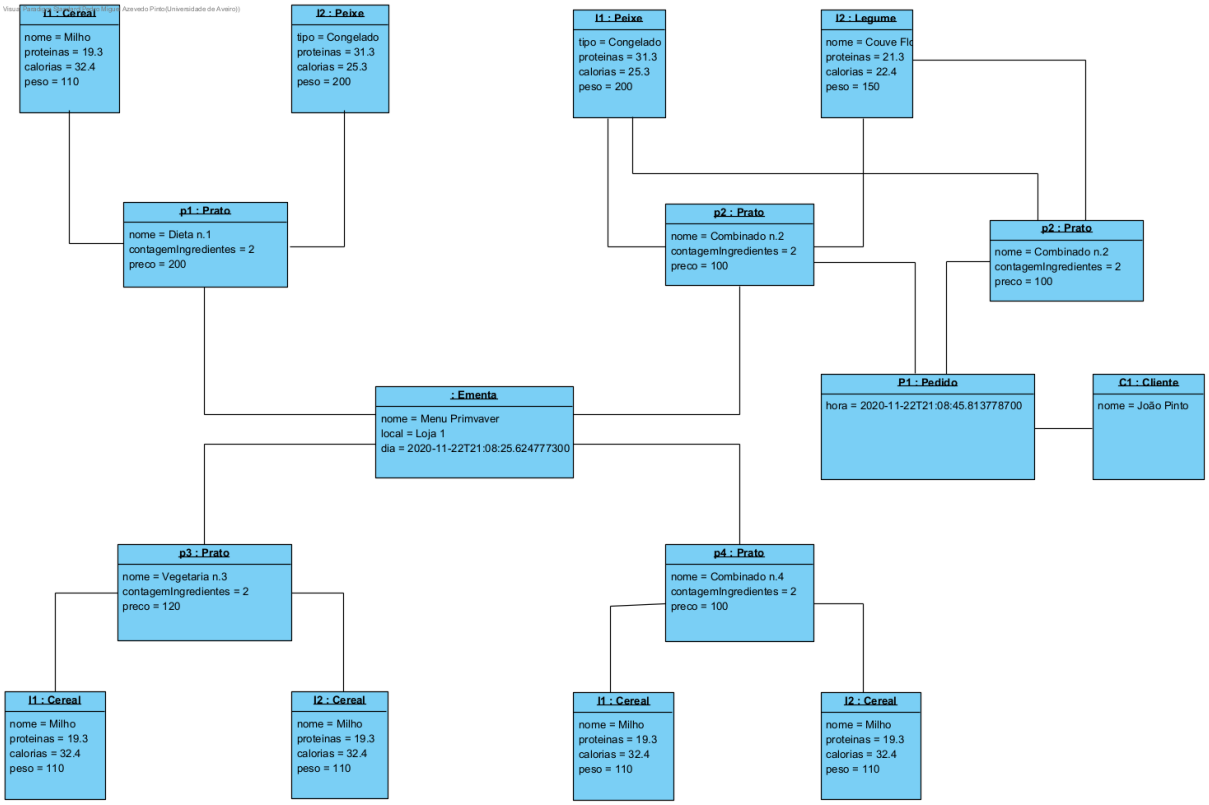
Vejamos, por exemplo, a implementação do pagamento por Referência Multibanco. Primeiramente, o Cliente pede que o pagamento seja feito por Referência Multibanco. Depois da forma de pagamento ter sido selecionada, o Comerciante pede que a EasyPay crie um pagamento frequente por Referência Multibanco. A EasyPay responde com uma Entidade e Referência Multibanco, a qual o Comerciante envia para o Cliente. O Cliente procede ao pagamento, e quando concluído, a SIBS notifica a EasyPay do sucesso do pagamento. Seguidamente, a EasyPay envia uma Notificação Genérica ao Comerciante para confirmar que o pagamento foi concluído com sucesso. O Comerciante vai, então, pedir os Detalhes do Pagamento Frequente à Easypay, de modo a atualizar os detalhes da transação e confirmar os dados recebidos. Após essa confirmação, o Comerciante confirma ao Cliente que o pagamento foi efetuado com sucesso.

4.3 - Interações no código (por objetos)

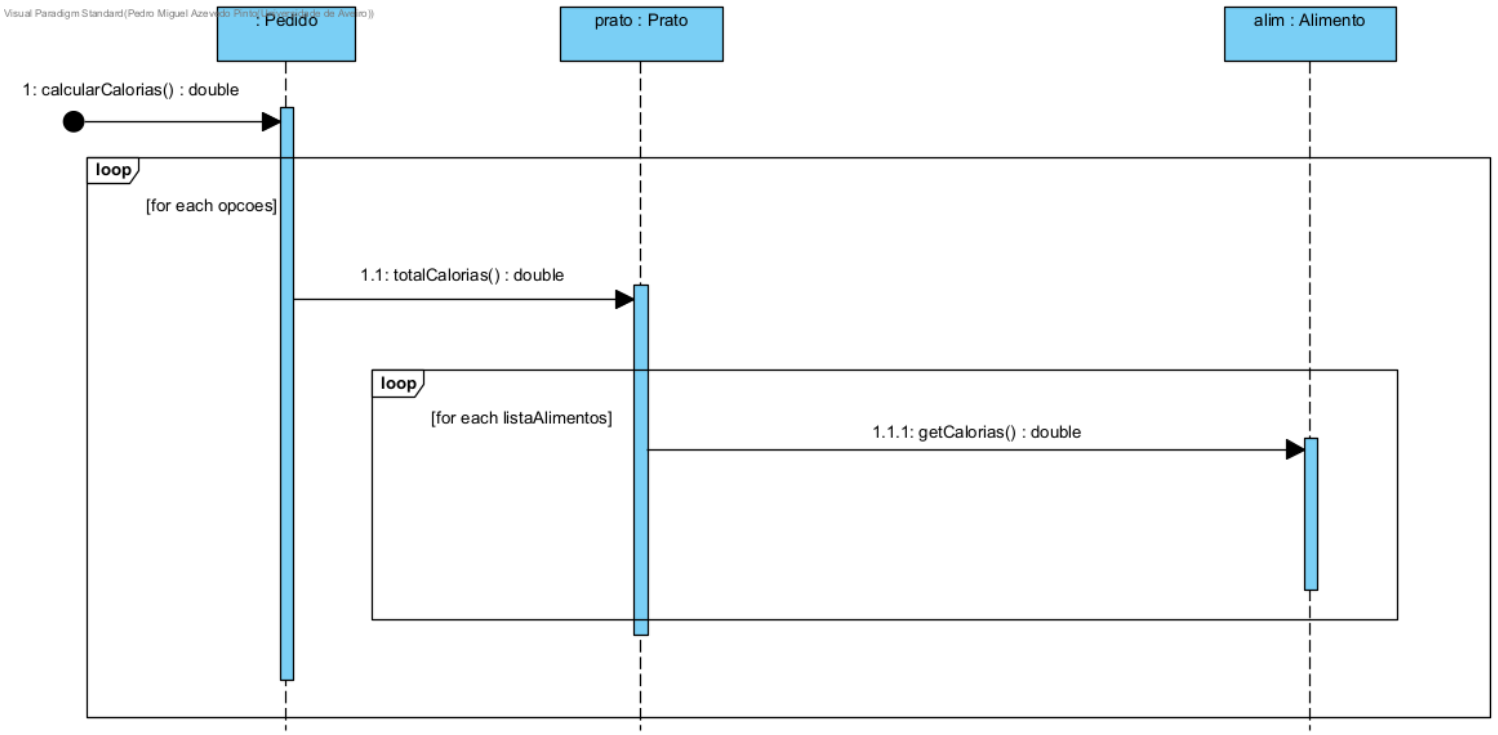
4.3.1 - Visualização da estrutura do código



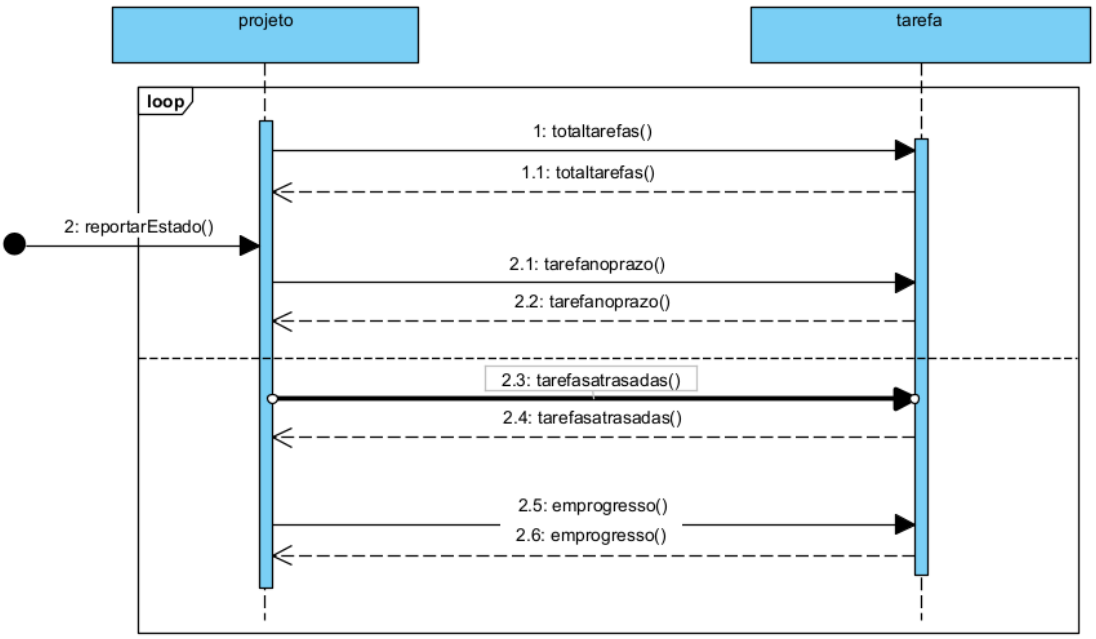
4.3.2 - Visualização das instâncias (objetos)

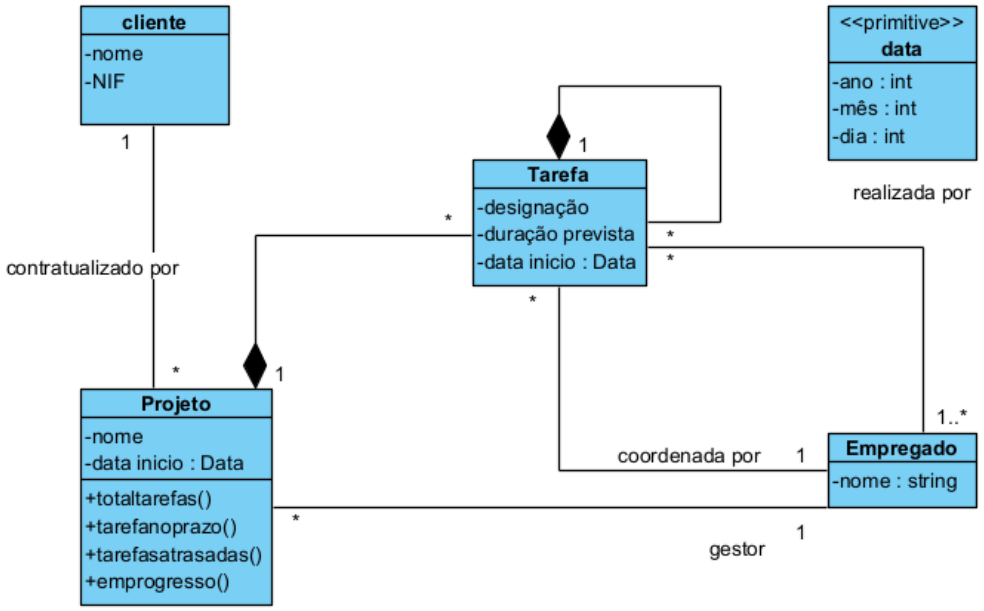


4.3.3 - Visualização da interação entre objetos de código



4.3.4 - Diagrama de sequência como instrumento de "descoberta"/planeamento





4.4 - Gestão de parcerias

a) O diagrama de atividades terá duas colunas, uma dos "Membros (os Estafetas)" e outra da "Direção da Logística". Começa na ação dos membros se candidatarem, que vai para a ação de aprovação desta candidatura por parte da Direção. A atividade pode acabar por parte da Direção, sendo suspensa ou cancelada em função das denúncias recebidas, ou acabar por parte do próprio Membro, que pede para acabar a adesão.

b) Neste caso, vão haver dois utilizadores: os Estafetas e a Direção. Os Estafetas utilizam a Plataforma para para se registarem, enviarem uma candidatura e até pedir para acabarem com a sua adesão. A Direção utiliza a Plataforma para aprovar a candidatura dos Estafetas, suspendê-los, ou até cancelar as suas adesões.

c) Com a informação presente no texto, não seria possível criar um diagrama de classes muito detalhado, já que não estão explícitos quais teriam que ser os seus atributos. Contudo, poderia fazer-se um que não oferece pertinência na sua resolução, em que uma das classes seria o Membro/Estafeta, que teria por atributos o seu nome (string), data de nascimento (data, com ano(int), mês(int) e dia(int)), morada (string), nacionalidade (string), estatuto de carta de condução (válida ou inválida (bool)), entre outros, que se ligaria à classe do Pedido de Adesão. Esta teria como atributos a data em que foi feita(data, com ano(int), mês(int), dia(int), hora(int), e minutos(int)), e o estado (aprovado ou reprovado(bool)). O Pedido de Adesão liga-se à classe Adesão, 'Caso o Pedido seja Aprovado'. A Adesão teria como atributos, por exemplo, a data em que foi aprovada (data, com ano(int), mês(int), dia(int), hora(int), e minuto(int)).

Existiria também a classe da Direção da Logística, que como atributos teria o nome(string). Esta ligar-se-ia ao Pedido de Adesão, já que o iria 'Aprovar ou Reprovar', e à Adesão, já que a pode 'Suspender ou Cancelar'. Além disso, o Membro/Estafeta também estaria ligado à Adesão, já que este pode 'Pedir para terminar'.

d) Seria possível criar um diagrama de interação simples com base na informação do texto, que poderia ter alguma pertinência de o fazer, já que apenas seriam trocadas duas mensagens: a do pedido de adesão e a do término da adesão. Este diagrama, teria dois objetos, o Membro e a Direção da Logística. O Membro enviaria a mensagem do pedido de adesão, e a Direção responderia com a sua aprovação ou reprovação. Depois, ou o Membro enviaria a mensagem que pretenderia terminar a Adesão, ou a Direção enviaria a mensagem de que a Adesão do Membro estaria suspensa ou cancelada.

e) Neste caso, poderíamos fazer um diagrama de máquina de estados do Pedido de Adesão, que passa de “Pedido de Adesão Enviado” para “Pedido Aprovado” ou “Pedido Reprovado”. No caso do pedido ser reprovado, este ligar-se-ia a um ponto de saída. Já no caso do pedido ter sido aprovado, este passaria para “Adesão”, que seguiria para “Adesão suspensa” ou “Adesão cancelada/terminada”- No caso da Adesão suspensa, esta poderia voltar para a “Adesão”, ou para a “Adesão cancelada/terminada”. No caso de “Adesão cancelada/terminada”, esta iria dar ao fim do processo.