Alunos: Eniedson Junior, Lucian Julio, Pedro Henrique, Regina Felipe

Relatório de Projeto LP2 Psquiza

Design geral:
Para o nosso design adotamos um modelo que garanta um baixo acoplamento e um maior nível de abstração, facilitando a comunicação entre as classes a partir de um controlador geral.
O controlador geral atua como um intermediário entre a facade do sistema e os controladores específicos, desta forma é possível trabalhar de forma mais eficiente.
O baixo acoplamento foi garantido através da criação de controllers especificos para as entidades principais do sistema (Atividade, Problemas e Objetivo, Pesquisa e Pesquisador), que atuam de maneira independente entre si, gerenciando apenas as necessidades de suas entidades. Suas responsabilidades são limitadas a fim de trabalhar em um escopo que permita que exerçam suas atividade principais de maneira eficiente, impedindo de interferência desnecessária entre as classes.
Para solucionar a grande quantidade de validações no sistema, optamos por fazer de uma classe Util.
Uma das abordagens para facilitar a resolução de modificações futuras no código, foi a implementação de enums, a partir de seu uso é possível diminuir a quantidade de código necessário na adição, remoção e modificação de suas categorias.
O design geral do nosso projeto se apoio em conceitos importantes que facilitam o reuso do código e facilitam modificações futuras no código.

Caso 1 (Regina Letícia):
No caso um foi necessário manipular as principais informações que caracterizam uma pesquisa, como descrição, campo de Interesse, código que a identifica unicamente e o estado da pesquisa, se era ativa, ou seja ainda em processo de elaboração, ou desativada, ou seja, encerrada. Para o desenvolvimento eficaz neste caso optei por criar um entidade Pesquisa, que possui os atributos da pesquisa e métodos que manipulam a pesquisa como método que altera seus atributos, encerra ou ativa uma pesquisa e o método que padroniza a exibição de um pesquisador. Para armazenar e gerenciar as pesquisas criei uma classe controladora que possui métodos para cadastrar, encerrar ou ativar uma pesquisa, retornar se ela é ou não ativa e por fim alterar seus atributos.

Caso 2 (Eniedson Junior):
O caso 2 era responsável pelas operações com os pesquisadores, para sua elaboração optei pela criação de uma classe “Pesquisador” que contém um nome, uma biografia, um email, um link para uma foto, uma função (representada por uma String) e um boolean que guarda se o pesquisador é ativo ou não. A partir disso criei uma classe controladora, que armazena os pesquisadores e gerencia os mesmos.

Caso 3 (Lucian Julio):
No caso 3, foi apresentado o envolvimento de problemas e objetivos que uma pesquisa pode ter, para essa parte decidi criar duas novas classes uma que representaria o problema e outra que representaria o objetivo. A entidade problema possui uma descrição e uma viabilidade, já o objetivo ele possui um tipo que identifica se ele é um objetivo geral ou específico, esse tipo é representado por uma classe enum para deixar o código mais explícito, mais legível, e menos vulnerável a erros de programação, além disso, outro atributo é a descrição, viabilidade e aderência, um caso especial é que a representação do objetivo tem um valor que é derivado da soma da viabilidade e aderência dele. Para essas novas entediantes foi decidido criar uma entidade controller que ficou responsável por gerenciar os objetivos e problemas, cadastrando problemas e objetivos, removendo e os exibindo.

Caso 4 (Pedro Henrique):
No caso 4, decidimos criar as classes Atividade, ControladorAtividade e Item, e os enums Estado e Risco. A atividade possui uma descrição, descrição de risco, período e nível de risco que é gerenciado pelo enum Risco, que pode ser BAIXO, MÉDIO ou ALTO, assim facilitando o controle caso haja a necessidade de modificar os tipos de risco. Além disso, a atividade possui uma coleção (LinkedHashMap) de itens (Item), que por sua vez possuem um nome e um estado gerenciado pelo enum Estado, que pode ser PENDENTE ou REALIZADO. O motivo da escolha de um enum para item é o mesmo do anterior (da escolha de Risco para atividade). Através desta organização é possível, cadastrar atividades, itens à atividades, desassociar um item de uma atividade, contar a quantidade de itens pendentes e realizados que uma atividade possuir e exibir uma atividade. E finalmente o controlador de atividade gerencia através de um mapa (HashMap) as atividades.

Caso 5 (nome do aluno responsável):
Para o caso 5, escolhemos usar as classes Objetivo, Problema e ControladorMetas, o controlador de metas gerencia os objetivos e problemas. Os objetivos caracterizam os objetivos de uma pesquisa e devem estar associados a uma pesquisa, fizemos isso através de um mapa (Map) localizado no controlador de metas. Como uma pesquisa pode estar associada a vários objetivos adicionamos um conjunto (HashSet) de objetivos no ao associar um objetivo a uma pesquisa. Além disso, nesta associação cadastramos uma pesquisa ao objetivo, como uma String visto que o objetivo só pode estar associado a uma pesquisa. Já para problema, optamos por armazenar também em um mapa (Map) no controlador de metas. Ao associar um problema a uma pesquisa, decidimos armazenar o próprio problema na pesquisa, para facilitar seu acesso em casos posteriores. Com esta abordagem podemos criar objetivos e problemas, associar-los e desassociar-los de

uma pesquisa, bem como listar eles, evitando assim conflitos através de nossa implementação.

Caso 6 (Regina Letícia):
Para desenvolver a funcionalidade de associar um pesquisador a uma pesquisa optei por criar dentro da entidade pesquisa uma coleção de pesquisadores, fazendo com que a pesquisa agora tenha acesso a seus pesquisadores. O método que associa ou desassocia foi criado dentro do controlador da pesquisa. Para a funcionalidade que especializa o Pesquisador eu optei por criar uma interface chamada especialidade que teria o comportamento de alteração de atributos e exibição de atributos e duas entidades que implementam essa interface que seria Aluna e Professora, e cada uma possui suas características que não são em comum com um pesquisador. Enquanto na classe de um pesquisador optei por criar uma composição com a interface especialidade, ou seja, o pesquisador agora possui um atributo especialidade que é inicializado como Aluna ou Professora quando solicitado. Por fim, foi criado também um método que lista os pesquisadores de acordo com a sua funcao, aluno (estudante), professor ou externo. Essas funcionalidades citadas que dizem respeito ao pesquisador como: especialização de pesquisador, exibição e alteração são encontradas dentro do controlador de pesquisadores, garantindo assim o baixo acoplamento.

Caso 7 (Lucian Julio):
No caso 7 é pedido que a pesquisa agora possa ser associada a diferentes atividades e possa capturar seus resultados produzidos. Dado esse problema a abordagem utilizada foi guardar as atividades na pesquisa de forma que a pesquisa agora tenha acesso direto às atividades na qual ela está associada. A pesquisa ela também pode executar suas próprias atividades, esse tempo que ela é executada é guardado na atividade sendo representado por um inteiro, outro métodos que esse caso 7 tratou foi o cadastrado dos resultados por item, esse cadastro guarda o resultado na própria atividade, sendo ele representado por uma String. Também é possível remover resultados, listar resultados e pegar o período de tempo que uma atividade foi executada. Para implementar essa caso de uso não foi necessário criar nenhuma nova entidade.

Caso 8 (Eniedson Fabiano):
O caso 8 era responsável pela busca por um termo no sistema de três formas diferentes, a primeira sendo uma busca comum, retornando todos os objetos que coincidem com a busca, a segunda uma busca específica, onde é passado o índice do resultado desejado e a terceira quer que seja retornado a quantidade de

resultados para uma determinada busca. Para a resolução desse problema, optei pela criação de um método de busca em cada um dos controladores, e um método no controlador geral que realiza a união dessas buscas, além disso, para as duas outras formas de busca, utilizei o método de busca já criado sistema e, no caso da segunda forma, apenas acessava o resultado medido pelo índice, e na terceira apenas contava o número de resultados na busca.

Caso 9 (Eniedson Junior):
O caso 9 era responsável por definir uma ordem de execução para representar uma lógica de execução das atividades, para isso optei pela adição de um atributo do tipo atividade (a próxima) dentro de cada atividade, esse atributo inicia como nulo e é alterado quando o usuário decide definir uma próxima atividade para aquela atividade, apenas uma próxima atividade pode ser definida por vez, ou seja, para que possa ser definida uma nova atividade deve-se primeiro tirar a próxima já existente, tornando o atributo nulo novamente. Como cada atividade pode ou não ter uma próxima, isso forma uma espécie de lista encadeada, e os método para obtenção das próximas atividades e de contagem foram feitos recursivamente.

Caso 10 (Lucian Julio):
O caso 10 ele trata de uma melhoria na forma de funcionamento do sistema para que o usuário tenha uma experiência melhor, foi pedido que o sistema ofereça uma sugestão da próxima atividade de acordo com uma estratégia definida pelo usuário, podendo ser, a atividade mais antiga, menos dependências, maior risco e maior duração. Dessa forma agora a entidade pesquisa armazena o tipo de estratégia que será usada para determinar a próxima atividade, foram criados dois novos métodos para o sistema, um que determina o tipo de estratégia que será usada para pegar a próxima atividade e outro para pegar a próxima atividade para o usuário. Com essas novas funcionalidades adicionadas o sistema ficou bem mais inteligente e prático para o usuário.

Caso 11 (Regina Letícia):
No caso de uso 11 foi pedido que fossem gravados arquivos de texto possuindo um resumo ou resultados de uma pesquisa, de acordo com o solicitação de um usuário. Para desenvolver essa funcionalidade eu criei dentro do controlador de pesquisa três métodos, o que recebe a solicitação do usuário para gravar o resumo, e pega o Resumo da pesquisa solicitada, o que recebe a solicitação dos resultados e pega os resultados da pesquisa solicitada e ainda o escritor, um método interno da classe que cria um arquivo de texto com o nome passado e escreve neste arquivo o resumo ou resultados. Dentro da entidade Pesquisador utilizei os pesquisadores, atividades, objetivos e problema a ela associado e criei strings que retornavam o resumo ou os resultados.

Caso 12 (Pedro Henrique):
No caso 12 foi criado a necessidade de persistir os dados do sistema em um arquivo externo. No caso para guardarmos todo o conteúdo fundamental do sistema, decidimos criar a classe GerenciadorControladores, cujo o papel é salvar em um arquivo os atributos dos ControladorAtividade, ControladorMetas, ControladorPesquisa e ControladorPesquisador. Também é possível carregar os atributos salvos de um arquivo, assim eles são salvos como atributos do próprio gerenciador de controladores e permite que sejam retornados novos controladores instanciados a partir dos atributos já carregados. Optamos por manter todas as informações em só um arquivo, mas a partir desta implementação é possível mudar para uma abordagem de uso de vários arquivos.

Link para o repositório no GitHub: https://github.com/pedrohos/p2-psquiza