Trabalho
Especificando Componentes

Componentização e Reúso de Software
Instituto de Computação
Universidade Estadual de Campinas

André Santanchè
2020

Elabore um detalhamento da modelagem baseada em componentes para o tema “Gerenciamento de Fornecedores em MarketPlace”. Para os diagramas deste projeto, está sendo disponibilizado um arquivo de modelos no endereço:
https://docs.google.com/presentation/d/1HWiTx0HU781sf3A7sdAda_LQeMHqbvIXkh-uSRDo0Js/edit?usp=sharing

As equipes podem utilizar esses modelos como base e/ou usar qualquer software de elaboração de diagramas, contanto que ele apresente informações equivalentes àquelas aqui solicitadas.

O projeto deve ser elaborado de forma top-down, partindo dos componentes de maior granularidade e indo em direção aos componentes de menor granularidade, conforme detalhado a seguir.

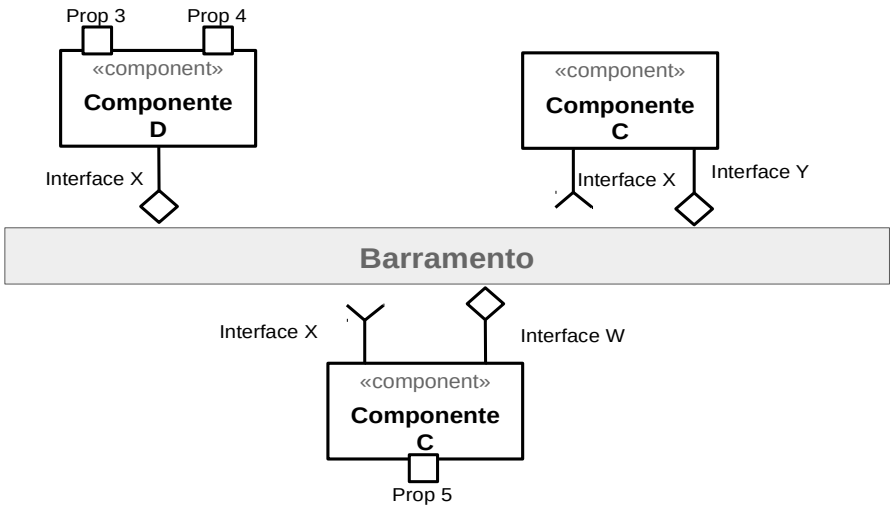
Nível 1

Neste nível de granularidade maior, devem ser considerados grandes componentes que representem módulos do sistema que gerencia o sistema de Marketplace. Esses componentes são serviços que se comunicam por um barramento.

Devem ser consideradas todas as funcionalidades do sistema descritas no documento de “Gerenciamento de Fornecedores em MarketPlace”. No processo de auxílio a tomada de decisão, descrito no referido documento, além do uso do histórico deve ser acrescentado um processo de leilão invertido, na forma de uma orquestração, detalhado a seguir:

- cada fornecedor/loja deve ser tratado como um serviço se comunicando pelo barramento;
- para a recomendação dos fornecedores de cada produto o sistema inicia um leilão informando a todos os potenciais fornecedores daquele produto sobre a demanda;
- a informação dos produtos de menor preço será utilizada como parte do auxílio de tomada a decisão.

Sugere-se a representação a seguir para esse nível de granularidade:



Os componentes nesse nível de granularidade deverão se comunicar através de publish/subscribe. Descreva a forma como os componentes interagem em alto nível conforme o exemplo:

- O componente **x** inicia o leilão publicando no barramento a mensagem de tópico

“**leilão/<número>/início**” através da interface **Gerente Leilão**, iniciando um leilão.

- O componente **Y** assina no barramento mensagens de tópico “**leilão/+/início**” através da interface **Participa Leilão**. Quando recebe uma mensagem...

Há um exemplo mais detalhado na próxima seção que também pode ser adotado como base. Note que é usada uma estrutura hierárquica de tópicos inspirada no MQTT, conforme está descrito no documento: https://github.com/santanche/component2learn/blob/master/labs/02-data-flow_messages/dcc-reference.md. Recomenda-se o uso deste padrão, inclusive com coringas (wildcards).

Deve ser descrita pelo menos duas interações: (i) um processo de compra completo envolvendo todos os componentes do marketplace; (ii) o processo de leilão.

Para cada interface deve ser detalhada conforme o template, com pelo menos as seguintes informações:

- tópico que a respectiva interface assina ou publica;
- formato da mensagem JSON associada ao objeto enviado/recebido por essa interface.

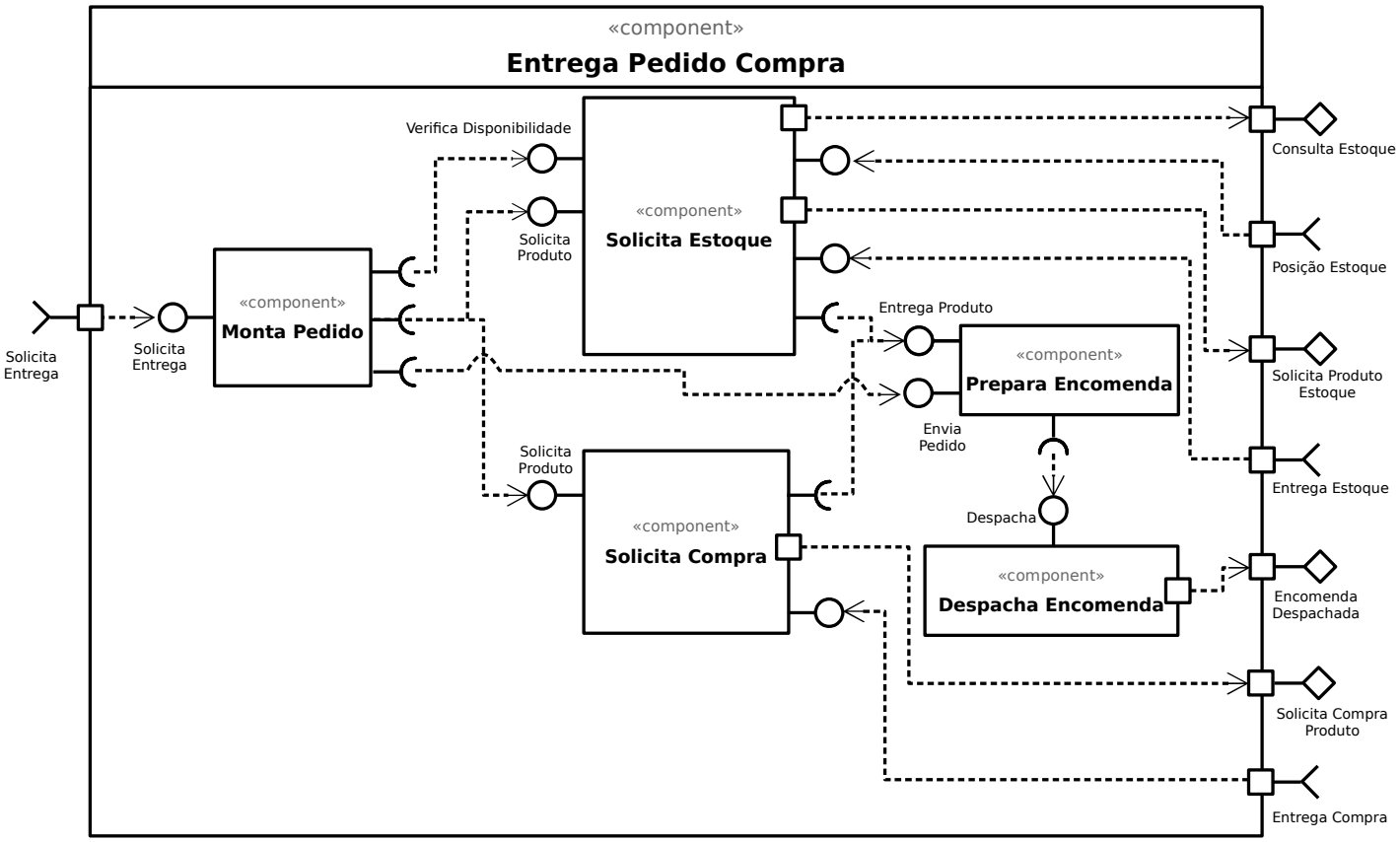
Nesse nível de detalhe a equipe decide se fará a separação MVC ou se prefere deixar para o próximo nível. Se houver separação MVC neste nível, pode haver comunicação síncrona com interface provida/requerida entre componentes para efeito do MVC.

Nível 2

Considerando que os componentes da etapa anterior são de maior granularidade e que se comunicam através de um barramento de mensagem, cada equipe deve escolher pelo menos um componente para detalhar internamente.

Se a equipe optou pela separação do MVC no diagrama anterior, deve ser escolhidos pelo menos um componente Controller e os Model/View associados para detalhamento. Se a equipe optou por realizar a separação nesse estágio, o componente escolhido deve ser separado em componentes Model/View/Controler interligados.

A estrutura interna do componente conterá subcomponentes que se comunicam através de conexões, como ilustra o exemplo na figura a seguir:



Deve ser elaborado um diagrama UML seguindo o modelo acima, com eventos especificados seguindo o modelo CORBA Component Model (CCM).

O diagrama deve ser acompanhado de uma breve explicação da interação dos subcomponentes internos para fazer operar o componente externo, conforme o exemplo de explicação do componente de **`Entrega de Pedido de Compra`**:

- O componente **`Entrega Pedido Compra`** assina no barramento mensagens de tópico "**`pedido/+/entrega`**" através da interface **`Solicita Entrega`**.
  - Ao receber uma mensagem de tópico "**`pedido/+/entrega`**", dispara o início da entrega de um conjunto de produtos.
- Internamente este evento é atendido por uma interface provida do componente **`Monta Pedido`**, que é responsável por montar o pedido para entrega.
- Esse componente verifica a disponibilidade dos produtos em estoque, acionando o componente **`Solicita Estoque`** através da interface **`Verifica Disponibilidade`**.
- Os produtos disponíveis são solicitados para o estoque (componente **`Solicita Estoque`**) e para os demais é solicitada a compra (componente **`Solicita Compra`**). Ambas as operações são acionadas através da interface **`Solicita Produto`**.
- O componente **`Monta Pedido`** notifica o componente **`Prepara Encomenda`** que uma encomenda está a caminho através da interface **`Envia Pedido`**.
- O componente **`Prepara Encomenda`** aguarda a entrega dos produtos de estoque e comprados. Ele é notificado através da interface **`Entrega Produto`**.
- Uma vez que todos os produtos estejam disponíveis, ele ativa o despacho acionando o componente

Despacha Encomenda através da interface **Despacha**.

- Uma vez que a encomenda seja despachada, o componente **Despacha Encomenda** publica no barramento uma mensagem de tópico "**pedido/<número>/despacha**" através da interface **Encomenda Despachada**.

- Os componentes **Solicita Estoque** e **Solicita Compra** se comunicam com componentes externos pelo barramento:

  - Para consultar o estoque, o componente **Solicita Estoque** publica no barramento uma mensagem de tópico "**produto/<id>/estoque/consulta**" através da interface **Consulta Estoque** e assina mensagens de tópico "**produto/<id>/estoque/status**" através da interface **Posição Estoque** que retorna a disponibilidade do produto.

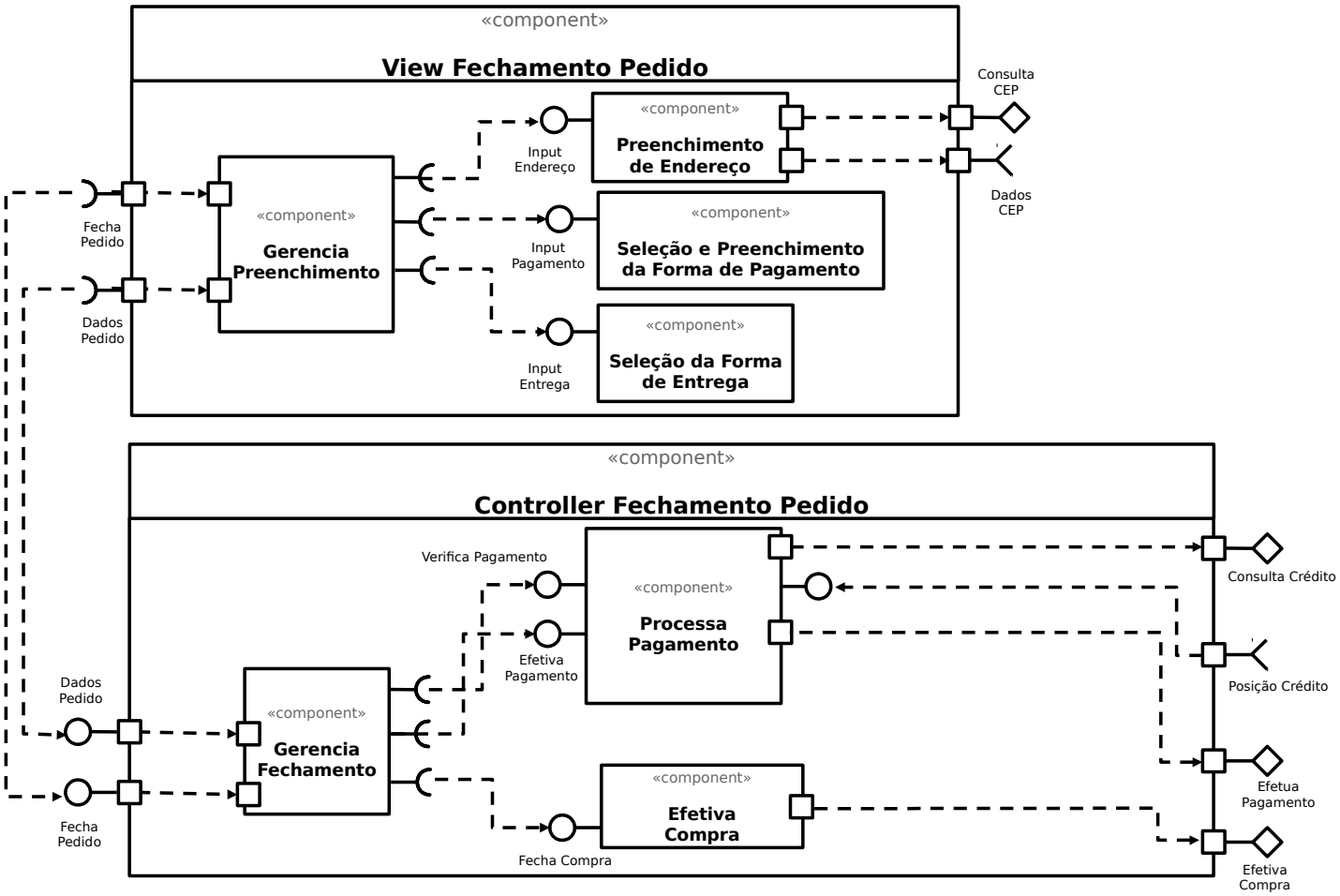
  - Para solicitar o produto em estoque, o componente **Solicita Estoque** publica no barramento uma mensagem de tópico "**produto/<id>/estoque/solicita**" através da interface **Solicita Produto Estoque** e assina mensagens de tópico "**produto/<id>/estoque/entregue**" através da interface **Entrega Estoque** que confirma a entrega da solicitação.

  - Para solicitar a compra de produtos, o componente **Solicita Compra** publica no barramento uma mensagem de tópico "**produto/<id>/compra/solicita**" através da interface **Solicita Compra Produto** e assina mensagens de tópico "**produto/<id>/compra/entregue**" através da interface **Entrega Compra** que confirma a entrega da solicitação.

Note que eventos recebidos do componente maior podem ser convertidos em acionamento de interfaces providas em componentes internos; a produção de eventos externos não é mapeada a uma interface provida/requerida, mas é associada a um componente que a produz, conforme mostra o diagrama.

Cada equipe elaborar o projeto em UML envolvendo os componentes e a conexão entre componentes . Tais componentes serão detalhados no diagrama de componentes alinhado com o diagrama de interfaces.

Considere que os componentes Model e View são independentes do Controller, ou seja, não são subcomponentes do Controller, mas se conectam com esse. A seguir é detalhado um diagrama sugerido para o detalhamento. Nesse caso, é detalhado o Controller e o View, mas o Model também deveria ser detalhado como um terceiro grande componente.



Múltiplas Interfaces

Considere que o seu sistema suportará duas interfaces distintas: uma usando a GUI nativa do Android e outra para Web a ser acessada por navegador. Escreva um texto detalhando como seus componentes podem ser preparados para que seja possível trocar de interface apenas trocando o componente View e mantendo o Model e Controller.

Detalhe quais as soluções de implementação que serão adotadas nas duas implementações, em termos de plataforma, linguagem, framework etc. Desenhe um diagrama de como a solução será implementada.

Entrega

Cada equipe deverá criar um projeto no Github específico para este projeto final em que participam todos os componentes da equipe. A apresentação deve seguir a template:

Até o dia 18/09 as equipes deverão finalizar o projeto e submeter o endereço do Github pelo Moodle. As equipes poderão levar uma prévia dos modelos no dia 12/09 para debate em sala de aula.