UCS - Universidade de Caxias do Sul

Allan Felipe Lemes

Gabriel Ricardo Tubias Rocha

Henrique Pagno de Lima

Valdir Demoliner

Relatório de Integração

Após reunião do grupo para discussão da forma de implementação, foi optado pelo uso das seguintes ferramentas/tecnologias para o desenvolvimento do projeto:

- Frontend (Interface): HTML + CSS
- Backend: PHP
- Banco de Dados: MySQL
- Versionamento: GitHub

Optamos pelo uso destas ferramentas, pois as mesmas são muito utilizadas atualmente e oferecem recursos apropriados para o desenvolvimento de projetos que envolvem as necessidades citadas neste. A escolha do banco MySql deve-se à facilidade de utilização do mesmo com PHP, visto que a ferramenta Xampp integra automaticamente os mesmos.

O uso destas tecnologias trouxe vantagens na produtividade de criação de uma interface mais amigável em relação à uma loja web em ambiente real. No entanto, como não utilizamos nenhum framework para o desenvolvimento do projeto, todas as telas/views do mesmo precisaram ser criadas manualmente, sendo desenvolvidas em código nativo. O uso de um framework teria, de fato, agregado em produtividade, porém poderia ter comprometido possíveis necessidades de manutenção ou customizações do código do projeto. Essa escolha, no entanto, foi opção do grupo por preferir criar um código nativo (*from scratch*), sem a utilização destes frameworks.

Foi utilizado também o GitHub para o controle de objetos e versões da aplicação, visto que o mesmo possibilita controle e compartilhamento através da web, permitindo que todos os integrantes do grupo manipulem a aplicação simultaneamente, controlando possíveis concorrências e conflitos dos objetos do projeto.

Definimos também as histórias que seriam necessárias para o projeto, conforme as necessidades levantadas na descrição do mesmo. As histórias foram as seguintes:

- Cadastro de Produto

- Cadastro de Pedido
- Consulta de Estoque
- Atualização/Implantação de Estoque

Estas histórias foram relacionadas e explicadas na primeira parte do projeto, entregue no webfólio, assim não foram descritas neste relatório.

A interface da aplicação foi desenvolvida baseada nas necessidades levantadas nas histórias, dessa forma:

- Cadastro de produtos
    - Permite o cadastro dos produtos utilizados no sistema (CDs e Livros). Utiliza os campos de código, nome, descrição e valor unitário.
- Atualização/implantação de estoque
    - Solicita o código do produto a ser alterado e a nova quantidade do mesmo.
- Consulta de estoque
    - Permite a consulta do estoque atual dos produtos cadastrados no sistema. Exibe em um grid os produtos cadastrados com seu respectivo estoque e permite um filtro das informações pelo código/nome do produto.
- Pedidos de venda
    - Permite a geração de pedidos de venda para os clientes (Americanas e Saraiva). O usuário fornece o código do cliente (temporariamente fixo devido à exclusividade de clientes) e os itens do pedido, informando a quantidade de cada um deles.

O banco de dados da aplicação foi criado com uma estrutura simples, atendendo apenas às necessidades básicas necessárias ao projeto. Segue abaixo a estrutura das tabelas:

- Itens
    - Nome do item: 'nm-item'
    - Código da categoria: 'cd-categ'
    - Código do item: 'cd-item'
    - Descrição do item: 'des-item'
    - Valor do item: 'val-item’
- Pedidos
    - Código do pedido: 'cd-pedido'
    - Código do item: 'cd-item'
    - Quantidade do item: 'qtd-item'

    - Código do cliente: 'cd-cliente'
- Estoque
    - Código do item: 'cd-item'
    - Quantidade estoque: 'qtd-estoque'

O uso de XP (Extreme Programming) para o desenvolvimento do trabalho exigiu o desenvolvimento de testes unitários para a aplicação. Isso trouxe um desafio maior ao projeto, visto que nenhum dos integrantes já havia trabalhado com estes. Em vista da utilização de PHP no desenvolvimento, tornou-se necessário o uso do PHPUnit para o desenvolvimento dos testes unitários. Uma dificuldade do grupo em relação a isto foi a exigência do desenvolvimento dos testes unitários antes da própria aplicação, o que não nos é comum, e assim torna-se complexo compreender como os testes devem ser realizados antes da criação da própria aplicação. Assim, infelizmente, os testes acabaram sendo desenvolvidos no fim do desenvolvimento, e não no início. A ferramenta PHPUnit também ofereceu certa dificuldade em seu uso, visto que desconhecíamos a forma apropriada para o seu uso.

Foi interessante o conhecimento do uso de testes unitários para a compreensão da forma como estes podem ser úteis durante o desenvolvimento de um projeto, prevendo assim possíveis problemas ao longo de todo o processo de criação do mesmo.

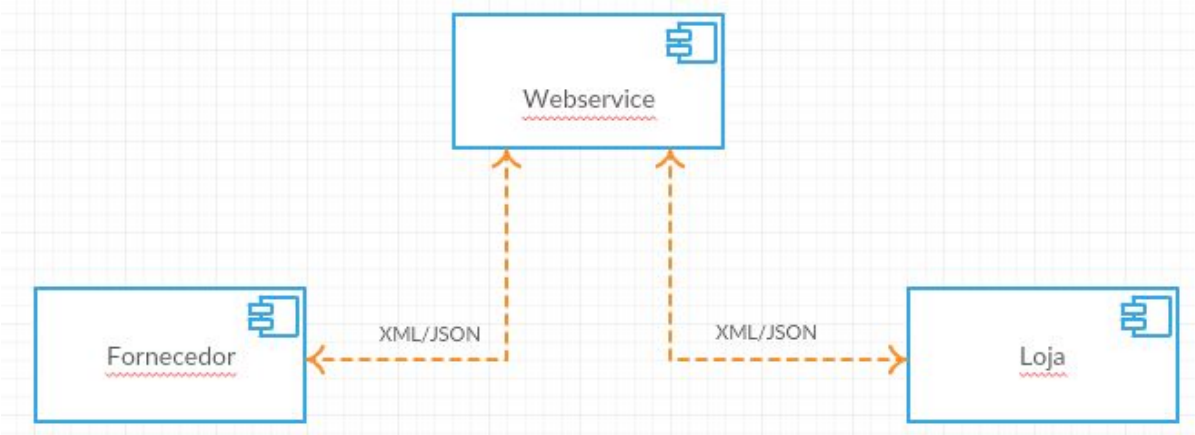
O grupo optou pela integração entre as aplicações utilizando webservices REST e conteúdo em formato JSON para a comunicação dos dados, desenvolvido utilizando o framework “Slim”. Entendemos que o uso de um webservice é mais viável para aplicações online, além de ser uma tecnologia mais usada atualmente. Foi sugerido para que os grupos de lojas (Grupos 1 e 2) optassem pela criação de um cadastro “De -> Para”, de forma a identificar quais códigos de produto dos fornecedores seriam equivalentes aos códigos cadastrados em seu sistema.

O webservice desenvolvido permite as seguintes requisições:

- Requisição de um catálogo completo de produtos do fornecedor;
    - *getAllItem*
- Requisição de produto(s) específico(s) do fornecedor;
    - *getItem*
- Envio de requisição da criação de um pedido de compra, podendo ter um ou mais itens. Nesta requisição, o webservice retorna a informação da criação do pedido com êxito ou o respectivo erro;
    - *setPedido*
    - *setItem*

- Requisição do pedido criado.
    - *getPedido*

A aplicação segue, então, a seguinte estrutura para comunicação:



Os testes de integração entre as aplicações trouxeram alguma dificuldade, em virtude da divergência de idéias dos grupos entre a forma apropriada de comunicação entre as aplicações, além de eventuais problemas de rede. Assim, ajustes foram necessários para o correto funcionamento, exigindo diálogo entre os grupos.

Concluímos observando que os principais problemas enfrentados foram a utilização dos novos conceitos tal como os testes automatizados (TDD) e o desenvolvimento dos mesmos utilizando JUnit, além da utilização da framework “Slim” e o desenvolvimento do webservice em si, processo que era desconhecido pelo grupo.