Computação Móvel como Facilitador de Aplicações Web

Análise para Aplicações de Comércio Eletrônico usando o Magento

José Eduardo de Azevedo Sousa

JUIZ DE FORA

JULHO, 2016

Computação Móvel como Facilitador de Aplicações Web

Análise para Aplicações de Comércio Eletrônico usando o Magento

José Eduardo de Azevedo Sousa

Universidade Federal de Juiz de Fora

Instituto de Ciências Exatas

Departametno de Ciência da Computação

Bacharelado em Ciência da Computação

Orientador: Romualdo Monteiro de Resende Costa

JUIZ DE FORA

JULHO, 2016

COMPUTAÇÃO MÓVEL COMO FACILITADOR DE APLICAÇÕES WEB

Análise para Aplicações de Comércio Eletrônico usando o Magento

José Eduardo de Azevedo Sousa

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE CIÊNCIAS EXATAS DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA, COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.

Aprovada por:

Romualdo Monteiro de Resende Costa
Doutor em Informática (PUC-RJ)

Marcelo Ferreira Moreno
Doutor em Informática (PUC-RJ)

Igor de Oliveira Knop
Doutor em Modelagem Computacional (UFJF)

JUIZ DE FORA

29 DE JULHO, 2016

À minha família, pelo suporte em todos os âmbitos.

Aos amigos que comigo caminharam na ardua trajetoria de aprendizado.

Aos meus professores que me proporcionaram conhecimento e desenvolvimento deste trabalho.

Resumo

O comércio de mercadorias sofreu grande mudança nos últimos anos e a modernização foi fundamental para sua evolução. Grandes empresas apostam cada vez mais em conquistar seus clientes por comodidade de compra através da Internet, evitando o deslocamento e facilitando cada vez mais o processo de compra e reduzindo a evasão. Com isso surgiu o Magento, uma plataforma e-commerce para que o estabelecimento de uma loja virtual seja mais fácil, confiável, prático e completo. Este trabalho apresenta uma aplicação móvel voltada para o Magento com o objetivo de melhorar a usabilidade administrativa, automatizando tarefas e aplicando a hipermídia adaptativa de forma a tornar seu uso mais simples e intuitivo.

Palavras-chave: Magento, e-commerce, mobile, facilitador.

Abstract

The trade of goods has suffered a huge change in the last few years and the modernization was fundamental to its evolution. Huge companies concentrate even more their efforts in obtaining and retaining clients through online shopping facilities, avoiding the need to go on-site and turning the shopping process much easier preventing from possible loss of clients. Given this scenario, Magento came up as a new e-commerce platform for an easier, more reliable, practical and complete commerce setup. This work presents a mobile application focusing Magento and improving it's usability over administrative side, automatizing tasks and using adaptive hypermedia to become easier and more intuitive to use.

Keywords: Magento, e-commerce, mobile, ease.

Agradecimentos

Agradeço aos meus pais por me incentivarem e cobrarem desempenho visando meu bem.

Ao professor Romualdo Monteiro de Resende Costa pela orientação, enorme paciência e dedicação para que este trabalho pudesse ser desenvolvido com êxito. Muito além disso, agradeço sua grande didática que me fez admirar esta área da computação.

Aos professores do Departamento de Ciência da Computação pelos seus ensinamentos e aos funcionários do curso, que durante esses anos, contribuíram de algum modo para o nosso enriquecimento pessoal e profissional.

Aos meus amigos e colegas de trabalho por compreenderem esta fase tão conturbada e, de alguma forma, me ajudarem no desenvolvimento deste trabalho.

"When it's over, so they say

It will rain a sunny day."

John Fogerty (Creedence)

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviações

API	Application Programming Interface
DCC	Departamento de Ciência da Computação
PHP	PHP: Hypertext Preprocessor
UFJF	Universidade Federal de Juiz de Fora
REST	Representational State Transfer
SOAP	Simple Object Access Protocol
QoS	Quality of Service
SKU	Stock Keeping Unit
XSD	XML Schema Definition
HTTP	Hypertext Transfer Protocol
JSON	JavaScript Object Notation
XML	eXtensible Markup Language

1 Introdução

O comércio é uma atividade relevante para a economia. Seu funcionamento, no entanto, não é estático, pois é necessário adequá-lo ao estilo de vida dos potenciais compradores. Com a inserção da Internet no estilo de vida das pessoas, as lojas, que antes eram somente físicas, hoje em dia necessitam estar on-line.

Para facilitar o oferecimento do comércio on-line, plataformas de comércio eletrônico (1) podem ser utilizadas. Essas plataformas oferecem vantagens em relação ao desenvolvimento de aplicações específicas, podendo ser destacadas o tratamento de vulnerabilidades existentes no funcionamento desse tipo de aplicação, o aprimoramento do desempenho devido a sua melhoria contínua e a agilidade no oferecimento do serviço.

Atualmente, com mais de 150 mil lojas em uso em diferentes países e várias aplicações (2), o Magento [1] é a plataforma mais popular de sua categoria. Entre os benefícios conhecidos dessa ferramenta estão o conjunto de técnicas e métodos para melhorar o posicionamento da página em mecanismos de busca e uma gama muito rica e diversificada de *plug-ins* para uso. Graças as suas funcionalidades, até mesmo grandes empresas, de diferentes ramos de atividade, que poderiam desenvolver ferramentas específicas, incluindo a Samsung[2], Nokia[3], GoodYear[4], Nike[5], Olympus[6], Ford[7], utilizam essa ferramenta. Apesar de oferecer uma licença de uso gratuito[8], desde 2011 pertence ao eBay[9].

Ao contrário do *software* desenvolvido sob demanda, a utilização de plataformas de comércio eletrônico como modelo genérico de solução poderia não oferecer aos usuários todos os requisitos desejáveis. Para resolver esse problema, essas ferramentas, incluindo o Magento, oferecem diversas funcionalidades, para atender usuários dos mais diferentes ramos de atividades. Essa proposta, no entanto, traz um outro problema que é a grande

[1] https://magento.com/
[2] http://www.samsung.com/
[3] http://www.nokia.com/
[4] http://www.goodyear.com.br/
[5] http://www.nike.com.br
[6] http://www.olympus-global.com/
[7] http://www.ford.com.br/
[8] https://magento.com/legal/licensing
[9] http://www.ebay.com/

quantidade de operações que esses usuários podem precisar realizar para configurar o seu comércio virtual, devido à grande quantidade de campos para preencher, atividades extensas, entre outros. Nesse cenário, o uso de facilitadores mostra-se interessante como forma de agilizar o real oferecimento do serviço por meio de várias ferramentas que serão apresentadas.

A utilização de dispositivos móveis pode ser um facilitador importante, ao permitir aspectos de mobilidade a essa tarefa, garantindo, entre outros ganhos, a agilidade no processo. No entanto, esses dispositivos apresentam limitações importantes já conhecidas, como a interface reduzida, restrições ou ausência de recursos e quedas de conectividade. Essas limitações se tornam mais evidentes em aplicações que exigem uma grande quantidade de operações, como é o caso, justamente, do ambiente administrativo do Magento.

1.1 Apresentação do tema

A computação móvel envolve a interação entre homens e equipamentos na qual é esperado que o computador seja transportado durante seu uso normal e envolve a comunicação, *hardware* e *software* móveis. Nela é permitida a transferência de dados, voz e vídeo por meio de computadores sem a necessidade de conexões físicas. Esse paradigma de uso dos computadores se faz presente em quase todos os eventos na atualidade, sejam eles para informação, entretenimento ou mesmo para o trabalho. Além disso, devido à grande praticidade conferida pela mobilidade, a inclusão de serviços ainda não disponíveis no meio móvel acontece frequentemente.

No começo da computação, as aplicações, em sua grande maioria, funcionavam de forma isolada, sem a necessidade de acesso a servidores, comunicação com outras máquinas para troca de informações ou qualquer tipo de interação entre clientes. Com a forte disseminação do uso da Internet nas últimas décadas, várias aplicações migraram para a *Web*, descentralizando o uso, permitindo novas experiências, oferecendo praticidade e o compartilhamento de informações, entre outras vantagens.

O Magento é um *e-commerce* que representa uma importante facilidade na Web, pois permite estabelecer uma loja virtual sem grandes conhecimentos em programação devido à sua interface interativa e instaladores prontos. Associando-se a um estabeleci-

mento responsável por pagamentos é possível realizar vendas por meio da Internet através dessa ferramenta até mesmo sem uma loja física. Esse cenário mostra-se adequado a uma nova demanda de mercado aberta por clientes que buscam praticidade, agilidade e, em alguns casos, evitar contato com vendedores.

Na última década, uma tecnologia que se tornou bastante popular no mercado e se difundiu com facilidade entre os usuários foi o *smartphone*(3). Os sistemas operacionais de smartphones mais renomados, tais como Android e iOS possuem em suas lojas[1] um número significativo de aplicativos que funcionam como facilitadores do uso de aplicações direcionadas à Web. Novas ferramentas, no entanto, nem sempre oferecem ganhos em termos de facilidade e agilidade de uso aos seus usuários. A deficiência de algumas ferramentas ou ainda a inexistência de facilitadores em algumas áreas proporcionam oportunidades de desenvolvimento.

1.2 Problema

Em uma sociedade com conhecimentos cada vez mais específicos, usuários podem enfrentar dificuldades quando o uso de uma ferramenta não tange o seu conhecimento técnico. Surge então a necessidade de se construir uma interface para oferecer aos usuários uma ferramenta final de forma simples e intuitiva.

Ferramentas como o e-commerce Magento podem ser utilizadas por comerciantes, sem conhecimentos técnicos específicos. Surge, então, a necessidade de pesquisa com enfoque na construção de facilitadores desse e-commerce buscando minimizar os problemas de configuração enfrentados pelos seus usuários.

Em (4) o autor afirma:

> "Produtos cada vez mais complexos são mais sensíveis a necessidade de serviços especializados."

Portanto, a dificuldade do uso está diretamente relacionada à necessidade de suporte especializado em um sistema. Essas dificuldades devem ser vistas como problemas a serem resolvidos, a fim de evitar a não aderência dos usuários. Principalmente quando se

[1] Play Store e Apple Store respectivamente

trata de facilitadores em ambiente móvel, há a necessidade de contextualizar a ferramenta, de forma a permitir que o seu funcionamento seja sensível ao contexto, sendo necessário definir quais campos e áreas são, de fato, interessantes ao usuário, além de ser capaz de oferecer informações úteis. Um exemplo consiste em analisar a situação e sugerir ações, tais como ao identificar que se encontra em época natalina propor a realização de uma promoção, sugerindo mudança de preços e até mesmo de categoria (produtos em promoção, por exemplo).

1.3 Justificativa

Muitas aplicações, atualmente, funcionam na *Web*, empregando o uso de tecnologias tais como HTML(5), CSS(6), Javascript(7), PHP(8), ASP.NET(9) entre várias outras, que voltam sua dedicação, primariamente, ao acesso através de computadores de mesa, onde há disponibilidade de dispositivos de entrada, como teclados físicos e mouse, facilitando a inserção de dados.

Ao mesmo tempo, essas tecnologias não tendem a favorecer o uso de recursos que podem não estar ao alcance do usuário, tais como câmera, alto-falantes e microfones. Devido ao aumento do uso de smartphones pelos usuários essas aplicações web deixaram de atender satisfatoriamente, pois deixam de usar todos os recursos possíveis que aumentam o nível de interação entre usuário e aplicação.

Análises recentes de mercado apontam que as aplicações nativas têm maior desempenho se comparadas àquelas desenvolvidas diretamente para Web(10) quando usadas em um smartphone, além de proporcionar maior possibilidade de lucro. Em contrapartida, o custo de desenvolvimento de aplicações e a grande dificuldade de abranger todas as plataformas possíveis é maior.

Os smartphones têm um grande potencial para tornar simples e efetiva a forma de uso de aplicações para Web visto que contém vários dispositivos reunidos em um só. O uso de dispositivos móveis podem conferir maior agilidade na inserção de um produto no e-commerce devido, por exemplo, a presença da câmera, permitindo ao usuário que a foto seja tirada de qualquer lugar, sem a necessidade posterior de uma transferência para computador de mesa e assim, então, inserí-la no Magento.

O smartphone confere também uma maior mobilidade em relação ao notebook, visto que há grandes facilidades de porta-lo em praticamente todas as situações. Diferentemente dos outros dispositivos, através do uso do smartphone, até mesmo curtos períodos de tempo disponíveis pelo usuário, em quaisquer lugares, podem ser aproveitados para atualizar uma loja virtual.

1.4 Objetivos

O objetivo geral deste trabalho é desenvolver uma aplicação móvel capaz de facilitar o uso da parte administrativa do *Magento*, seja por meio da aplicação de automatização ou da hipermídia adaptativa juntamente com inferências.

A plataforma Magento é utilizada como aplicação Web foco do facilitador objeto deste trabalho. Esse facilitador, por sua vez, é implementado como uma aplicação do sistema Android para dispositivos móveis.

Apesar das plataformas escolhidas servirem de meios para a análise das funcionalidades propostas, não existe uma dependência total dessas funcionalidades com as respectivas plataformas. Um dos objetivos deste trabalho é permitir que as idéias apresentadas possam ser implementadas em outras plataformas e soluções. Portanto, não é objetivo deste trabalho analisar aspectos da usabilidade associados a, por exemplo, modificações do leiaute do Magento. Ao contrário, o objetivo principal busca simplesmente trazer a facilidade da computação móvel à administração de aplicações Web. Adicionalmente, considerando aspectos limitadores da computação móvel, este trabalho busca explorar alternativas para diminuir a quantidade de informações necessárias às operações, automatizar operações realizadas e, adicionalmente, explorar mecanismos que auxiliem o usuário, indicando as operações a serem realizadas baseado em inferências sobre o contexto. Para alcançar os objetivos mencionados, são explorados:

- A utilização de arquétipos que permitam categorizar a inclusão de produtos no sistema, buscando diminuir a quantidade de informações preenchidas.

- A utilização de estruturas de grande alcance simultâneo para operações de atualização, permitindo modificar um conjunto de produtos, simultaneamente. É possível,

por exemplo, alterar o preço de um conjunto de produtos, ao invés de precisar, individualmente, realizar as modificações.

- Definir informações contextuadas, sobretudo, no tempo, a fim de indicar ao administrador operações que possam melhorar o sistema. Caso essas informações não existissem, seria necessário ao usuário realizar múltiplas navegações a fim de tomar a decisão. Um exemplo é sugerir promoções em datas especiais.

O capítulo a seguir descreve os conceitos que foram usados no trabalho e que são necessários para a compreensão de seu conteúdo.

2 Revisão Conceitual

Alguns conceitos precisam ser explorados para que as propostas deste trabalho possam ser alcançadas. A própria computação móvel, com destaque para as tecnologias encontradas nas principais plataformas, engloba os principais conceitos que necessitam ser compreendidos. Além desse modelo de computação, conceitos de customização e detalhes da plataforma Magento constituem itens importantes para a elaboração deste trabalho e que, por isso, são apresentados nas seções a seguir.

2.1 Computação Móvel

Mudanças no ambiente de execução dos sistemas trazem prováveis necessidades de propostas para resolver ou, ao menos minimizar, as diferenças e, principalmente, as limitações que possam ser impostas por esse novo ambiente. Na mudança da computação desktop para a computação móvel essas necessidades podem ser observadas em vários aspectos.

Um desses aspectos envolve as limitações das redes de comunicação (wireless ou rede de dados da operadora) em determinados espaços (11; 12). A computação móvel, como paradigma computacional, exige, além do dispositivo móvel, a necessidade de um canal de comunicação que, no entanto, não precisa estar disponível durante toda a execução da aplicação. Existe, concomitantemente, a necessidade de uma aplicação que funcione sem grandes processamentos por parte do dispositivo e que não faça constantemente requisições de hardware de forma a o exaurir.

Um outro aspecto de limitação corresponde à disponibilidade de memória, que deve ser planejada, ou pode gerar uma experiência negativa ou insatisfatória por parte do usuário. A implementação deste trabalho é baseada no sistema operacional Android 6.0 (Marshmallow), lançado pela Google em 28 de maio de 2015(13). Os requisitos de uso de hardware são providos por esse sistema operacional, cabendo ao facilitador, objeto deste trabalho, utilizar, de maneira apropriada, os recursos oferecidos por essa plataforma.

Entre os recursos de comunicação utilizados, o protocolo HTTP merece destaque

por sua utilização na comunicação entre o facilitador e o Magento. Nas versões anteriores do Android, as classes do projeto Apache HttpComponents (https://hc.apache.org/) eram amplamente utilizadas na implementação da comunicação através do HTTP. No entanto, por dificuldades relacionadas à evolução e a manutenção da compatibilidade, o Android na sua versão 6.0 não oferece suporte às classes desse projeto, recomendando o uso das classes encontradas no pacote java.net. As classes desse pacote, com destaque para a HttpURLConnection, oferecem facilidades como o uso de sessões criptografadas (através do SSL/TLS), transmissão de streaming e o uso do IPV6. Essas facilidades, no entanto, tornam a sua utilização mais complexa, quando comparada a utilização da implementação oferecida pelo Apache HttpComponents.

Os métodos contidos no HTTP (GET, POST, PUT, PATCH, DELETE)(14) são utilizados, como mencionado, para a comunicação com a plataforma Magento, que oferece serviços no formato REST (Representational State Transfer). Nesse formato, as requisições são realizadas diretamente através dos métodos do HTTP. As respostas, usualmente, são construídas no formato JSON (Javascript Object Notation - http://www.json.org), um formato leve proposto para a troca de dados. Nesse formato os dados são empacotados em dois tipos de estruturas: mapeamento de pares chave/valores (objetos) e lista ordenada de valores (arrays).

2.2 Customização

Dado um sistema com com várias possibilidades, a existência da customização pode ser útil a fim de garantir uma boa experiência por parte do usuário. Aplicativos que oferecem uma plataforma para comércio eletrônico, incluindo o Magento, usualmente proporcionam uma grande variedade de opções para o usuário interagir com o sistema, através de menus extensos. No entanto, essa complexidade torna a utilização da aplicação pouco objetiva.

Uma boa parte das opções permitidas pelo sistema não faz parte do uso diário de um usuário comum, muita das vezes gerando até mesmo confusão na hora de navegar pelo caminho das funções. Deve-se observar então as funcionalidades fundamentais ao uso da ferramenta e, então, garantir um acesso direto e fácil a elas.

As Figuras 2.1 e 2.2 ilustram como a inclusão de um produto ao catálogo pode

ser extensa, demasiadamente complexa e confusa, dependendo do nível de afinidade do usuário com a área e sistema.

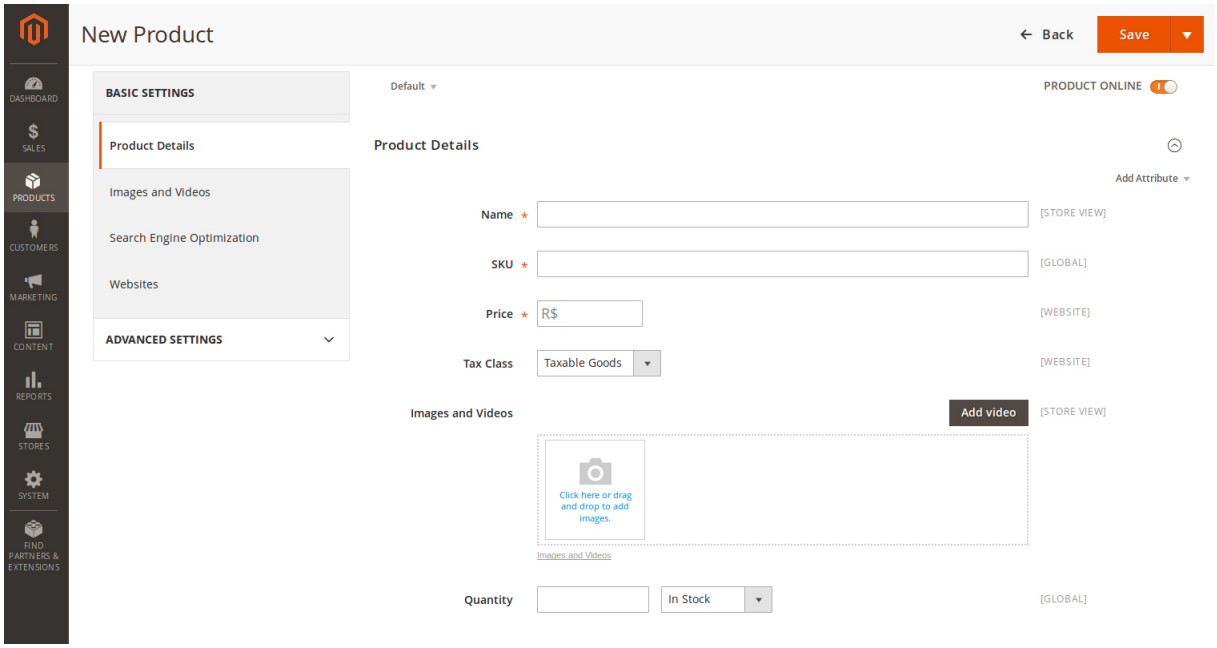

Figura 2.1: Inclusão de produto via interface de administrador web, parte 1

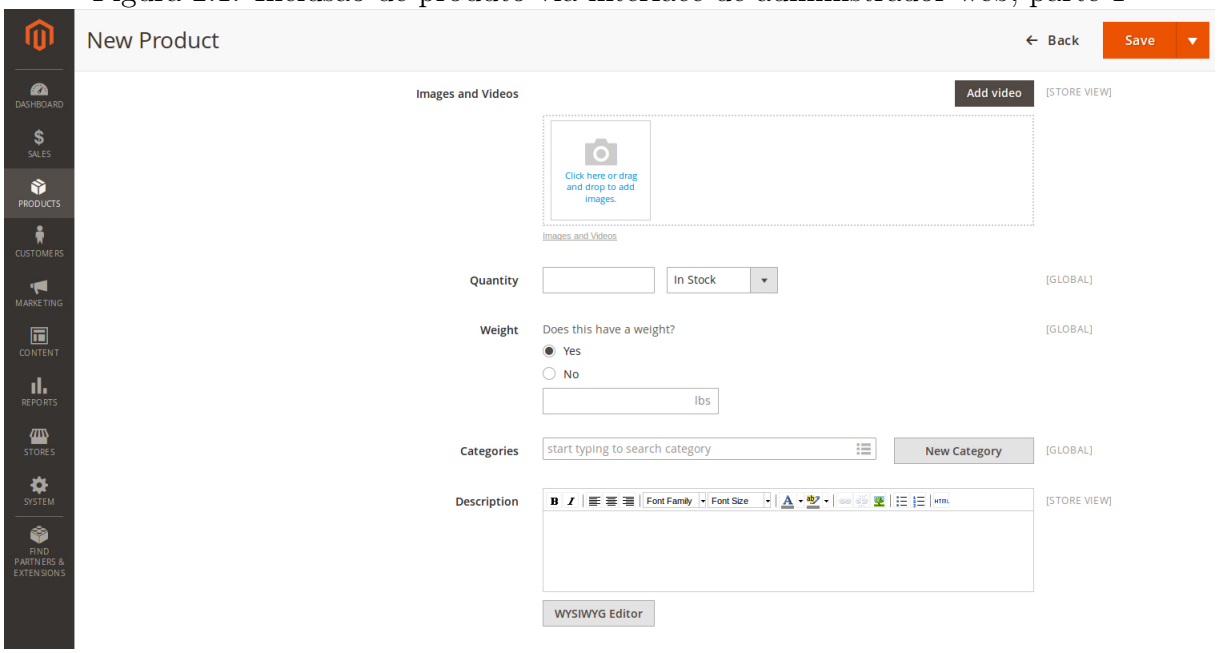

Figura 2.2: Inclusão de produto via interface de administrador web, parte 2

A operação de inclusão, como pode ser visto nas Figuras 2.1 e 2.2, além de envolver um número significativo de campos, contém campos com informações não cotidianas para muitas pessoas, tais como up-sell e cross-sell, definição de produtos relacionados, entre outros. Além das informações particulares de um produto, a operação de inclusão envolve outras partes como o "Search Engine Optimization"(15), área de otimização de buscas,

que pode ser de difícil configuração para usuários menos habilitados.

A inclusão no Magento envolve, portanto, diversos campos básicos, diretamente relacionados ao produto e outros campos com opções avançadas. Individualmente, o conteúdo desses campos pode confundir o usuário com menos experiência e, em conjunto, o preenchimento de todos esses campos podem ser uma tarefa complexa.

Não é possível, de forma fácil, eliminar grande parte das opções de um sistema com o objetivo de simplificá-lo. No entanto, é possível especificar as opções menos usuais, colocá-las em separado, e reutilizá-las a cada operação. Se necessário, essas opções menos usuais podem ser modificadas. Assim, é possível separar as opções de um sistema entre as partes comuns, das partes específicas, pouco modificadas.

Para implementar a customização de forma eficiente, uma opção consiste em definir perfis que contenham cada conjunto diferente das opções. Dessa forma, é possível substituir um conjunto de informações pela simples escolha do perfil. Cada perfil pode ser atribuído de forma direta, por exemplo, pelo tipo do produto ou ainda ser escolhido pelo usuário. Para a utilização dos perfis é necessária a sua construção. Embora perfis possam ser construídos em tempo de execução, este trabalho propõe uma especificação em duas etapas. Os perfis podem ser previamente construídos, através de uma linguagem proposta para esse fim.

2.3 Magento

O Magento, como anteriormente mencionado, é uma plataforma madura para aplicações de comércio eletrônico. Sua instalação ocorre de forma razoavelmente simples e intuitiva. Entre suas vantagens estão o gerenciamento do catálogo de produtos e a grande escalabilidade do software, de forma a atender tanto pequenos quanto grandes projetos. Caso uma condição qualquer não seja satisfeita, ainda há a possibilidade de recorrer a extensões já desenvolvidas e testadas pela comunidade (16) ou mesmo desenvolver sua própria solução.

A plataforma disponibiliza um conjunto de APIs de comunicação entre a aplicação e o sistema, em padrões REST e SOAP(17). Para a realização do trabalho, como anteriormente mencionado, é utilizada a comunicação via REST, em razão da facilidade que a plataforma Android oferece para a comunicação através desse formato.

Para estabelecer a comunicação, inicialmente, é necessário uma credencial de comunicação para evitar o acesso indevido. O método escolhido foi o de geração de um token para validar a transferência de dados. Inicialmente, é cadastrado um usuário administrador do Magento via interface Web para que a geração do token possa acontecer, limitando suas permissões de acordo com os campos selecionados na hora de escolher o nível de privilégio do utilizador.

A Figura 2.3 apresenta um exempo de uma solicitação de sessão usando comandos cURL[1] ao Magento de forma a obter a autenticação. A resposta é enviada em formato JSON contendo o token a ser incorporado à sessão da aplicação, caso a solicitação tenha sucesso.

Figura 2.3: Requisição de Token usando cURL

```
curl -X POST "https://magento.host/index.php/rest/V1/integration/admin/
    token" \
      -H "Content-Type:application/json" \
      -d '{"username":"cliente@exemplo.com", "password":"123123q"}'
```

Em caso de sucesso na obtenção do token, o seu valor deve ser anexado no cabeçalho dos comandos HTTP. Atualmente, a plataforma Magento oferece acesso a diversas operações, que incluem consultas, inserções, modificações e remoção de produtos, clientes, estoque, categoria, pagamentos, entre várias outras finalidades diretas e indiretas. Os comandos para essas operações são padronizados em uma API Swagger [2] descrita na seção a seguir.

2.4 Estratégias de Venda

As estratégias de venda são muito importantes para o funcionamento e sucesso de um programa de comércio eletrônico. No Magento algumas delas são adotadas e são mencionadas no trabalho(18).

[1]cURL é um software que provêm ferramentas de biblioteca e linha de comando para transferir dados usando vários protocolos.

[2]REST API do Magento: http://devdocs.magento.com/swagger/index.html

O “Up-sell” é a exposição de produtos mais caros à clientes que possuem perfis de compradores de produtos com valor mais elevado, no intuito de fazê-lo despender uma quantia maior de dinheiro em suas compras.

Outras técnica utilizada é o “Cross-sell”, no qual o *software* exibe para o cliente em uma determinada página produtos complementares aos que estão sendo adquiridos.

A aplicação de ambas estratégias tem o intuito de promover o aumento de vendas e são amplamente utilizadas por empresas tais como McDonald’s(19), como por exemplo quando oferecem batatas fritas no momento em que um cliente faza o pedido de um hambúrguer.

Este trabalho leva em conta também o lead-time, que é o tempo necessário pra que um produto ou matéria-prima chegue, ao ser feito o pedido, incluindo o tempo de processamento e de fila.

2.5 Adaptabilidade e hipermídia adaptativa

A adaptabilidade é a característica que se confere às aplicações a capacidade de manter o seu funcionamento mesmo em situações diferentes daquelas para a qual ela foi projetada.

Aplicações hipermídia, por exemplo, usualmente necessitam de processos adaptativos a fim de manter o seu funcionamento. Esse tipo de aplicação é definida em (20) da seguinte forma:

> "(...) denomina-se sistema de hipermídia adaptativa todo sistema de hipertexto e hipermídia que reflita algumas características de seus diferentes usuários em modelos e aplique tais modelos na adaptações de diversos aspectos visíveis do sistema às necessidades e desejos de cada usuário"

Tem-se também em (21) que sistemas hipermídia adaptativos tentam antecipar as necessidades, preferências e desejos de seus usuários a partir de modelos representando seu perfil, nível de conhecimento, preferências, etc, como visto no trecho:

> “O objetivo geral dos sistemas e modelos de hipermídia adaptativa é, portanto, prover seus usuários com informação atualizada, subjetivamente interessante, com a ilustração multimídia pertinente, num tamanho e profundidade

adequados ao contexto e em correspondência direta com o modelo do usuário. Este funciona como uma referência para o sistema, que busca adaptar seu ambiente – um hiperespaço, por vezes caótico – às expectativas particulares de seus usuários. ”

2.6 Arquétipos

No almejo de alcançar os objetivos propostos há necessidade de usar ferramentas as quais possibilitam o desenvolvimento do trabalho e permitir a concretização do facilitador. Um dos componentes fundamentais para tal é o arquétipo.

Arquétipo é uma designação de modelo ou padrão, um protótipo que serve de exemplo para se elaborar uma obra(22). No contexto deste trabalho, arquétipo tem conceito próximo ao de template, sendo um modelo no qual os dados podem ser representados em um arquivo e tem o intuito de guardar valores iniciais para campos.

Devido a sua semelhança, definiremos os templates futuramente citados como arquétipos, seguindo os conceitos previamente apresentados concomitantemente com os posteriormente inseridos.

2.7 Swagger: API REST

Com uma conexão estabelecida e autenticada pode-se fazer uso do Swagger Magento (API REST) para acessar diversas funcionalidades do Magento. Essa API especifica o formato JSON com campos padronizados para as requisições.

A Figura 2.4 define como é o comportamento dos módulos envolvidos no trabalho, sendo o enfoque na comunicação entre eles.

A maioria dos casos estudados neste trabalho utilizam o objeto catalogProduct-RepositoryV1, que oferece acesso direto ao catálogo de produtos. Suas ações possíveis são o GET, POST, DELETE, para obter, submeter e apagar produtos, respectivamente.

Com exemplo, os métodos GET e POST podem utilizar a URL “/V1/products” e fazem uso da estrutura JSON representado nas Figuras 2.5, 2.6,2.7 ,2.8 e 2.9 para representar um produto no sistema.

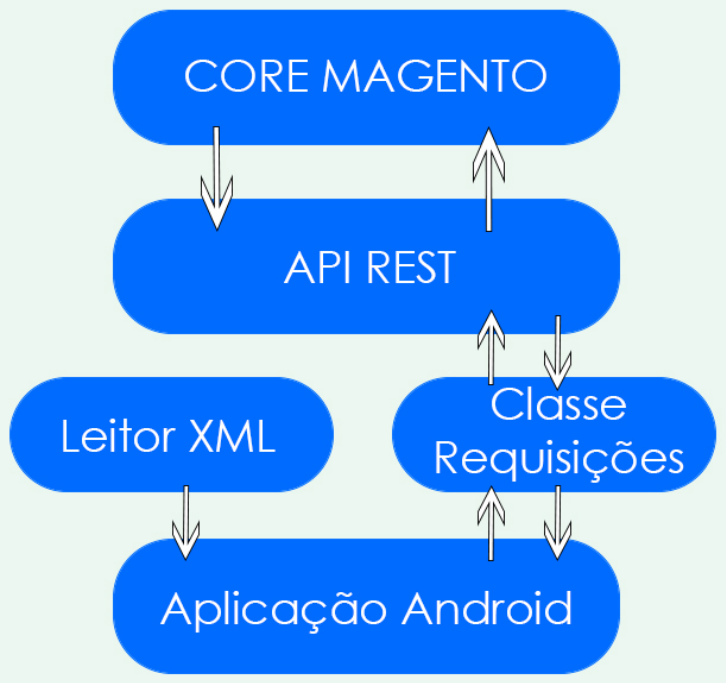

Figura 2.4: Arquitetura de comunicação de módulos do facilitador

As respostas usuais de uma chamada à API REST Magento retornam um JSON, como apresentado nas Figuras 2.5, 2.6, 2.7, 2.8 e 2.9. Nas figuras da Estrutura de dados JSON para manipulação de produtos (A-E) uma lista de itens, com conjuntos de objetos do tipo par e valor que representam as propriedades de cada item, são apresentados. As propriedades principais de cada item, fundamentais na implementação do facilitador são descritas no Capítulo 3.

```
{"items": [ {
    "id": 0,
    "sku": "string",
    "name": "string",
    "attributeSetId": 0,
    "price": 0,
    "status": 0,
    "visibility": 0,
    "typeId": "string",
    "createdAt": "string",
    "updatedAt": "string",
    "weight": 0,
    "extensionAttributes": {
      "bundleProductOptions": [
        {
          "optionId": 0,
          "title": "string",
          "required": true,
          "type": "string",
          "position": 0,
          "sku": "string",
          "productLinks": [
            {
              "id": "string",
              "sku": "string",
              "optionId": 0,
              "qty": 0,
              "position": 0,
              "isDefault": true,
              "price": 0,
              "priceType": 0,
              "canChangeQuantity": 0,
              "extensionAttributes": {}
```


Figura 2.5: Estrutura de dados JSON para manipulação de produtos- A

```
    }
  ],
  "downloadableProductLinks": [
    {
      "id": 0,
      "title": "string",
      "sortOrder": 0,
      "isShareable": 0,
      "price": 0,
      "numberOfDownloads": 0,
      "linkType": "string",
      "linkFile": "string",
      "linkFileContent": {
        "fileData": "string",
        "name": "string",
        "extensionAttributes": {}
      },
      "linkUrl": "string",
      "sampleType": "string",
      "sampleFile": "string",
      "sampleFileContent": {
        "fileData": "string",
        "name": "string",
        "extensionAttributes": {}
      },
      "sampleUrl": "string",
      "extensionAttributes": {}
    }
  ],
  "downloadableProductSamples": [
    {
      "id": 0,
      "title": "string",
      "sortOrder": 0,
      "sampleType": "string",
      "sampleFile": "string",
      "sampleFileContent": {
        "fileData": "string",
        "name": "string",
        "extensionAttributes": {}
      },
      "sampleUrl": "string",
      "extensionAttributes": {}
    }
  ],
  "stockItem": {
    "itemId": 0,
    "productId": 0,
    "stockId": 0,
    "qty": 0,
    "isInStock": true,
    "isQtyDecimal": true,
    "showDefaultNotificationMessage": true,
    "useConfigMinQty": true,
    "minQty": 0,
```


Figura 2.6: Estrutura de dados JSON para manipulação de produtos- B

```
      "useConfigMinSaleQty": 0,
      "minSaleQty": 0,
      "useConfigMaxSaleQty": true,
      "maxSaleQty": 0,
      "useConfigBackorders": true,
      "backorders": 0,
      "useConfigNotifyStockQty": true,
      "notifyStockQty": 0,
      "useConfigQtyIncrements": true,
      "qtyIncrements": 0,
      "useConfigEnableQtyInc": true,
      "enableQtyIncrements": true,
      "useConfigManageStock": true,
      "manageStock": true,
      "lowStockDate": "string",
      "isDecimalDivided": true,
      "stockStatusChangedAuto": 0,
      "extensionAttributes": {}
    },
    "configurableProductOptions": [
      {
        "id": 0,
        "attributeId": "string",
        "label": "string",
        "position": 0,
        "isUseDefault": true,
        "values": [
          {
            "valueIndex": 0,
            "extensionAttributes": {}
          }
        ],
        "extensionAttributes": {},
        "productId": 0
      }
    ],
    "configurableProductLinks": [
      0
    ]
  },
  "productLinks": [
    {
      "sku": "string",
      "linkType": "string",
      "linkedProductSku": "string",
      "linkedProductType": "string",
      "position": 0,
      "extensionAttributes": {
        "qty": 0
      }
    }
  ],
  "options": [
    {
      "productSku": "string",
```


Figura 2.7: Estrutura de dados JSON para manipulação de produtos- C

```
      "optionId": 0,
      "title": "string",
      "type": "string",
      "sortOrder": 0,
      "isRequire": true,
      "price": 0,
      "priceType": "string",
      "sku": "string",
      "fileExtension": "string",
      "maxCharacters": 0,
      "imageSizeX": 0,
      "imageSizeY": 0,
      "values": [
        {
          "title": "string",
          "sortOrder": 0,
          "price": 0,
          "priceType": "string",
          "sku": "string",
          "optionTypeId": 0
        }
      ],
      "extensionAttributes": {}
    }
  ],
  "mediaGalleryEntries": [
    {
      "id": 0,
      "mediaType": "string",
      "label": "string",
      "position": 0,
      "disabled": true,
      "types": [
        "string"
      ],
      "file": "string",
      "content": {
        "base64EncodedData": "string",
        "type": "string",
        "name": "string"
      },
      "extensionAttributes": {
        "videoContent": {
          "mediaType": "string",
          "videoProvider": "string",
          "videoUrl": "string",
          "videoTitle": "string",
          "videoDescription": "string",
          "videoMetadata": "string"
        }
      }
    }
  ],
  "tierPrices": [
    {
```


Figura 2.8: Estrutura de dados JSON para manipulação de produtos- D

```
            "customerGroupId": 0,
            "qty": 0,
            "value": 0,
            "extensionAttributes": {}
          }
        ],
        "customAttributes": [
          {
            "attributeCode": "string",
            "value": "string"
          }
        ]
      }
    ],
    "searchCriteria": {
      "filterGroups": [
        {
          "filters": [
            {
              "field": "string",
              "value": "string",
              "conditionType": "string"
            } ]  }
      ],
      "sortOrders": [
        {
          "field": "string",
          "direction": "string"
        }
      ],
      "pageSize": 0,
      "currentPage": 0
    },
    "totalCount": 0
}
```


Figura 2.9: Estrutura de dados JSON para manipulação de produtos- E

3 Templates

Muitas das aplicações que os usuários têm acesso na atualidade proporcionam uma enorme gama de opções de uso, comportamentos e funcionalidades. Funcionalidades são adicionadas e oferecidas em demasia para o usuário final de forma a suprir toda e qualquer necessidade relacionada ao tema.

Apesar da grande quantidade de recursos oferecer praticidade ao usuário por sua capacidade, ao mesmo tempo é incrementada a complexidade de uso da interface, exigindo, muitas vezes, a inserção de vários campos nem sempre necessários para particularizar uma solicitação.

No âmbito das lojas de comércio eletrônico, os campos de baixa relevância ficam evidentes. No Magento, por exemplo, produtos podem ser inicializados no sistema com três campos (nome, valor e quantidade), porém o usuário no ambiente administrativo precisa preencher dezenas de peculiaridades que, se por um lado abrangem todos os possíveis detalhes, por outro lado podem tornar a tarefa difícil.

Como não é possível o não preenchimento dos campos obrigatórios, este trabalho propõe o uso de templates que contenham o conteúdo geral de um produto. Esses templates seguem a definição de (23), "páginas pré-configuradas que podem ser editadas através de um formulário. São úteis para publicação de conteúdo de forma rápida".

No entanto estes estão limitados a definição de arquétipos, e têm o papel de definir valores iniciais para campos que pertencem a estrutura de dados de produto do Magento.

No contexto de páginas de comércio eletrônico, este trabalho propõe arquétipos com características definidas de acordo com a categoria do produto, pois refere-se a um grupo de produtos que possuem características similares ou próximas.

A designação das características similiares ou próximas fica a cargo do usuário, embora um conjunto pré-definido, oferecido pelo facilitador possa ser oferecido. O ideal, no entanto, é que usuários mais avançados possam utilizar as instruções fornecidas a fim de construir os templates necessários.

Como exemplo de especificação de contextos que poderiam gerar diferentes templates, está a possibilidade da venda fracionada ou unitária dos produtos oferecidos. Suponha, por exemplo, que em um comércio de eletrônicos há cabos de rede que possam ser vendidos em quantidades tais como 2,5 metros ou 5 metros, atendendo a fracionabilidade. Esse mesmo comércio pode possuir também monitores, os quais não podem ser vendidos em frações, assumindo assim características e categorias diferentes.

Continuando com o exemplo da diferença devido a categorização, suponha que esse mesmo comércio realize a venda de abraçadeiras plásticas, que podem ser vendidas somente a partir de uma quantidade mínima de unidades, seja devido ao seu valor ínfimo ou por política da empresa. Isso ocorre diferentemente de, por exemplo, um mouse que pode ser vendido tanto em uma unidade ou em milhares, de acordo com a demanda do comprador.

Os templates devem então ser preenchidos de acordo com a semântica apresentada ao contexto, sejam dados pertinentes e que se repetem na categoria ou eventos que se mostram importantes para a loja, que podem ser aprimoradas por profissionais que tenham conhecimento de ferramentas de comércio eletrônico e que normalmente fazem a configuração inicial do sistema. A partir da configuração inicial, o usuário com menos conhecimento na área pode fazer a adição de novos produtos de forma sucinta e eficaz sem alterar a qualidade das informações.

Para a construção dos modelos é necessário algum modo e/ou linguagem com intuito de descrever as idéias. Em (24) o autor sugere o uso do XML (25) para a descrição dos produtos em um comércio eletrônico:

> "À medida que o e-commerce e outras aplicações se tornam mais comuns, há uma necessidade crescente de estruturar páginas da Web e de separar o conteúdo da formatação. Por exemplo, um programa que pesquisa a Web em busca do melhor preço para algum livro ou CD precisa analisar muitas páginas da Web procurando pelo título e preço do item. Com páginas da Web em HTML, é muito difícil um programa descobrir onde está o título e onde está o preço. Por essa razão, o W3C desenvolveu um aperfeiçoamento para a HTML, a fim de permitir que as páginas da Web sejam estruturadas para pro-

> cessamento automatizado. Foram desenvolvidas duas novas linguagens para esse propósito. A primeira, chamada XML (eXtensible Markup Language) descreve o conteúdo da Web de uma forma estruturada."

Assim, o XML é adotado neste trabalho para a especificação dos templates. Seu uso é eficaz para o propósito, além da maneabilidade e facilidade para o entendimento. Outras opções para o armazenamento e o transporte de informações, tais como o JSON, poderiam ser utilizadas. No entanto, o XML se destaca pela possibilidade da validação da formatação, através da especificação de um esquema explícito que determina o tipo dos dados e a relação entre eles. Essa característica é importante se os dados são produzidos por pessoas e interpretados por máquinas, a fim de evitar falhas na construção da especificação. Como esse é justamente o cenário deste trabalho, a escolha do XML permite aos usuários especificarem os templates a serem interpretados pelo facilitador. Por fim, cabe destacar a utilização dessa solução em outros trabalhos para a mesma finalidade (templates) como pode ser observado em (26) e (27).

A Seção a seguir descreve o formato da linguagem proposta para a especificação dos templates. A utilização dessa linguagem instaura um uso dinâmico na aplicação, ao evitar que as facilitades sejam fixadas no código. Obtém-se, assim, vantagens tais como o ganho de dinamicidade e adaptação da aplicação.

3.1 Templates no Magento: Categorias

A principal aplicação proposta para os templates no Magento é formada por dados de produtos relacionados às suas categorias, descrevendo campos que provavelmente se repetiriam no preenchimento de produtos que pertencem a um mesmo grupo. Para cada produto, os únicos campos que não fazem parte do template são o nome, a quantidade e o valor, por serem dados fundamentais e únicos para cada produto. Adicionalmente, o SKU (Stock Keeping Unit), código único de um produto no sistema é gerado de forma automática pelo facilitador, combinando o seu nome com a data e hora atuais, evitando colisões.

Considerando a possibilidade de mudanças nos nomes dos campos do Magento,

principalmente em versões futuras, o formato proposto para o template não associa diretamente os campos do Magento a elementos ou atributos do XML. Ao contrário, são definidos apenas dois elementos (category e field), cada um com dois atributos (name e value). O elemento categoria (category) corresponde a um conjunto de produtos que terão valores de campos em comum, cada um representado pelo elemento campo (field). Para cada um desses elementos, o atributo nome (name), corresponde a sua descrição, enquanto o atributo valor (value), contém, como esperado, o valor da categoria (seu identificador) ou do campo. A Figura 3.1 apresenta o XML Schema para a especificação dos templates.

Os valores dos atributos nome e identificador da categoria identificam cada categoria. Os mesmos atributos para os elementos campos representam cada um dos campos do Magento com o preenchimento automático, para a categoria em questão. A Figura 3.2 apresenta a descrição das categorias "periferico"e "monitor", construídas para exemplificar o template, com os respectivos campos.

No Magento, o campo “attribute_set_id” define o tipo de atributos que a estrutura de dados deve receber, adquirindo um formato para cada modo de inserção de dados,como por exemplo, se contém algumas entidades ou não presentes, deixando a estrutura maior ou menor de acordo com o valor. Convencionalmente pelo sistema, admite-se o ID 4 como padrão, e este valor será usado nos exemplos adotados neste trabalho.

O campo denominado “status” refere-se ao estado do produto na loja. Sendo o valor "0"atribuindo a ele, o produto assume a configuração de “offline”, ou seja, não disponível para visualização na loja. Complementarmente, o valor "1"o define como “online”. Portanto, nos exemplos, será utilizado como padrão o valor "1", pois assume-se que ao inserir o produto o administrador tem a intenção de disponibilizá-lo aos clientes.

O campo “visibility” refere-se à visibilidade do produto no comércio eletrônico: contendo o valor “1” o produto é dado como individualizado, ou seja, visível somente via link direto. O valor “2” torna-o acessível pela lista de catálogo de produtos, já o valor “3” determina que o artefato pode ser visualizado somente por método de pesquisa. Finalmente, o valor “4” é o padrão do sistema, o valor que garante que o produto pode ser encontrado tanto por através do catálogo quanto através da pesquisa.

O campo “type_id” define se o produto é do tipo composto ou simples. Se for

Figura 3.1: XML Schema de especificação dos templates

```
<schema xmlns="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified" attributeFormDefault="unqualified" >

  <complexType name="fieldPrototype">
    <sequence minOccurs="0" maxOccurs="unbounded">
      <element ref="field"/>
    </sequence>
    <attribute name="name" type="string" use="required" />
    <attribute name="value" type="string" use="optional" />
  </complexType>

  <complexType name="categoryPrototype">
    <sequence minOccurs="0" maxOccurs="unbounded">
      <element ref="field"/>
    </sequence>
    <attribute name="name" type="string" use="required" />
    <attribute name="value" type="string" use="optional" />
  </complexType>

  <element name="field" type="fieldPrototype"/>

  <element name="category" type="categoryPrototype"/>
```

do tipo composto, um produto é formado pela associação com outros produtos, sendo necessário, portanto, fazer o link entre eles. Um exemplo é uma promoção que associa três produtos juntos com um outro produto específico por um valor 10% menor. Para o uso cotidiano pode ser definido, como padrão, a inserção de produtos simples.

O campo “weight”, por sua vez, tem como representação semântica o peso do produto. Muitas vezes seu valor não é de grande relevância para a inserção imediata, sendo então definido como “1” seu valor padrão neste modelo. Pode-se defini-lo como

Figura 3.2: XML representando as categorias periferico e monitor

```
<template>
    <category name="periferico" id="3">
        <field name="attribute_set_id" value="4"/>
        <field name="status" value="1"/>
        <field name="visibility" value="4"/>
        <field name="type_id" value="simple"/>
        <field name="weight" value="1"/>
        <field name="extensionAttributes">
            <field name="stockId" value="1"/>
            <field name="qty" value="1"/>
            <field name="isInStock" value="true"/>
            <field name="isQtyDecimal" value="false"/>
            <field name="useConfigMinQty" value="true"/>
            <field name="minQty" value="0"/>
            <field name="useConfigMaxSaleQty" value="true"/>
            <field name="maxSaleQty" value="3"/>
            <field name="useConfigBackorders" value="false" />
            <field name="backdorders" value="0" />
            <field name="useConfigQtyIncrements" value="true"/>
            <field name="qtyIncrements" value="0"/>
            <field name="useConfigManageStock" value="true"/>
            <field name="manageStock" value="true"/>
            <field name="isDecimalDivided" value="true"/>
            <field name="stockStatusChangedAuto" value="0"/>
        </field>
        <field name="saveOptions" value="true"/>
    </category>
    <category name="monitor" id="4">
        <field name="attribute_set_id" value="4"/>
        <field name="status" value="1"/>
        <field name="visibility" value="4"/>
        <field name="type_id" value="simple"/>
        <field name="weight" value="1"/>
        <field name="extensionAttributes">
            <field name="stockId" value="1"/>
            <field name="qty" value="1"/>
            <field name="isInStock" value="true"/>
            <field name="isQtyDecimal" value="false"/>
            <field name="useConfigMinQty" value="true"/>
            <field name="minQty" value="0"/>
            <field name="useConfigMaxSaleQty" value="true"/>
            <field name="maxSaleQty" value="3"/>
            <field name="useConfigBackorders" value="false" />
            <field name="backdorders" value="0" />
            <field name="useConfigQtyIncrements" value="true"/>
            <field name="qtyIncrements" value="0"/>
            <field name="useConfigManageStock" value="true"/>
            <field name="manageStock" value="true"/>
            <field name="stockStatusChangedAuto" value="0"/>
        </field>
        <field name="saveOptions" value="true"/>
    </category>
</template>
```

qualquer valor fracionado ou, ainda, valores inferiores a 0.

Alguns campos da API do Magento são compostos, isto é, são formados por outros campos. Um exemplo é o campo “extensionAttributes” que pode possuir vários campos

opcionais e alguns obrigatórios para o POST. Nos exemplos utilizados neste trabalho os seguintes dados foram utilizados para compor esse campo:

O campo “stockId” que corresponde ao código de estoque do produto. Pode assumir os seguintes valores: "1"para estoque local, "2"para estoque indireto e "3"para compra de outra empresa.

O campo “isInStock” refere-se à semântica do produto estar ou não em estoque. O valor “true” define que há produtos em estoque e “false” a negativa da afirmação.

O campo “qty” especifica a quantidade de produtos em estoque. Pode assumir qualquer valor natural, sendo que o valor "0"transforma automaticamente o atributo “isInStock” para false.

O campo “isQtyDecimal” especifica se o objeto em questão pode ser fracionado, por exemplo no caso de venda em metros, se é possível realizar a venda de 3.5 metros. O valor “true” assume a fracionabilidade e “false” define que somente valores naturais podem ser vendidos.

O campo “useConfigMinQty” define se o sistema trabalhará com uma quantidade mínima em estoque, ou seja, ao atingir esta quantidade no estoque do sistema não podem ser vendidas mais unidades do produto. Pode assumir o valor binário “true” ou “false”.

O campo “minQty” está relacionado com o campo “useConfigMinQty”. Caso o segundo esteja definido como “false” é desnecessária a especificação deste campo. Caso contrário, este campo define a quantidade mínima de produtos que deve existir no estoque.

O campo “useConfigMaxSaleQty” pode assumir os valores “true” ou “false” e sua semântica permite definir uma quantidade máxima de unidades do produto a ser vendida para um cliente através do campo “maxSaleQty” que faz referência a quantidade máxima em unidades que o produto pode ser vendido. Assume valores inteiros ou fracionados de acordo com o campo “isQtyDecimal”.

O campo “useConfigBackorders” refere-se a permissão ou não de backorders. Um backorder é um produto da loja que está temporariamente fora de estoque no fornecedor. Admite valores “true” ou “false”. Pode funcionar como pré-venda. Por padrão define-se como falso essa opção nos exemplos, a fim de evitar riscos de possíveis falhas na entrega dos produto.

O campo “backorders” refere-se ao campo que descreve a backorder. Pode conter o tipo, sendo pagamento parcial, adiantado ou somente após a disponibilidade em estoque, percentual de adiantamento, caso seja escolhido, a mensagem que será exibida ao cliente, a permissão de envio automático de mensagens ao cliente e o e-mail remetente. Definida por padrão para o modelo como falso.

O campo “useConfigQtyIncrements” é um campo binário que define a permissão do uso de uma configuração diferente de incrementos. Pode assumir os valores “true” ou “false”. Um exemplo seria a de venda métrica que incrementa de 2,5 metros em 2,5 metros para alcançar seu objetivo funcional. Usualmente definido como "false".

O campo “qtyIncrements” assume valores reais ou inteiros que definem, ao se apertar no botão de mais quantidade, qual a quantidade de incremento.

O campo “useConfigManageStock” permite que o produto usar propriedades de gerência de estoque ou não, decidindo o uso do campo “manageStock”. Pode assumir valores de “true” ou “false”. Definida no modelo por padrão como "true".

O campo “manageStock” define quais campos podem ser gerenciados, como por exemplo aceitar pré-venda ou não. Para permitir que todos os campos sejam modificados dessa forma usa-se somente o valor “true”. Definido nos exemplos como verdadeiro para todas as opções possíveis.

O campo “stockStatusChangedAuto” permite que o produto mude o seu estado de estoque automaticamente ao se atingir uma quantidade mínima especificada. Assume valores inteiros ou fracionários. Estabeleceu-se para os exemplos o valor “0” para evitar quaisquer problemas de continuidade de fornecimento. Com isso encerram-se os campos de “extensionAttributes”.

O último campo a ser adicionado para os atributos do produto é o “saveOptions” que permite o produto seja salvo com as opções escolhidas pelo cliente para acesso futuro. Um exemplo, no caso da venda de uma camisa, caso o usuário prefira a camisa verde, ao invés da tradicional camisa azul essa opção será salva e definida como preferencial. Assume valores “true” ou “false”.

3.2 Templates no Magento: Eventos

Além do template para a manipulação dos produtos, uma outra facilidade oferecida pela ferramenta envolve o tratamento de produtos em datas especiais. De forma similar ao template para os produtos, a escolha do template para a especificação de eventos envolve o uso de um arquivo XML. Esse arquivo é formado pelo elemento “eventTemplate” que possui como filhos eventos (elementos “event”) para descrever um evento temporal.

Um evento anual possui uma descrição, especificada pelo atributo "name"e um identificador único (representado pelo atributo "id"). Cada evento temporal corresponde a uma data do ano, sendo representada, portanto, pelo dia e pelo mês de sua ocorrência.

Para cada evento temporal uma ação é especificada. Essa ação corresponde à semântica que deve ser implementada pelo facilitador para o evento temporal em questão. Por fim, um evento temporal pode ter um intervalo de duração, que define um período, anterior, ou posterior onde a ação especificada deve ser mantida.

O arquivo XML criado no trabalho para suprir necessidades e que não são disponíveis no Magento é apresentado na Figura 3.3, com os campos “discountPercentage”, “date” e “howEarly”, descreve exemplos de datas especiais, através do campo data (Natal, Aniversário da Loja e Ano novo). O campo número descreve o número de dias que antecede o evento a partir do qual o usuário administrador deve ser notificado. Por fim, o campo percentual descreve o percentual de desconto básico a ser implementado na geração dos cupons de desconto para o evento considerado. Esse desconto, antes da emissão dos cupons, pode ser alterado pelo administrador.

De forma similar ao que acontece com a proposta dos templates para os produtos, no primeiro uso da aplicação, o usuário deve preencher o documento com as datas que considera de relevância para sua loja virtual. O sistema fará a leitura deste arquivo ao início da aplicação para verificar se a data atual pertence a um período descrito previamente e em caso positivo disparar uma mensagem alertando sobre a ação associada ao contexto da data. A Figura 3.4 descreve o XML Schema associado a esse outro template.

```xml
<eventTemplate>
    <event name="Natal" id="1">
        <attribute name="discountPercentage" value="10%"/>
        <attribute name="date" value="25/12"/>
        <attribute name="howEarly" value="5"/>
    </event>
    <event name="Aniversário da Loja" id="2">
        <attribute name="discountPercentage" value="20%"/>
        <attribute name="date" value="23/05/"/>
        <attribute name="howEarly" value="10"/>
    </event>
    <event name="Ano novo" id="3">
        <attribute name="discountPercentage" value="5%"/>
        <attribute name="date" value="01/01/"/>
        <attribute name="howEarly" value="3"/>
    </event>
</eventTemplate>
```

Figura 3.3: XML para descrever datas especiais

```xml
<schema xmlns="https://www.w3.org/2001/XMLSchema"
        elementFormDefault="qualified" attributeFormDefault="unqualified">
    <complexType name="eventPrototype">
        <sequence minOccurs="0" maxOccurs="unbounded">
            <element ref="field"/>
        </sequence>
        <attribute name="name" type="string" use="required"/>
        <attribute name="value" type="string" use="required"/>
    </complexType>
        <complexType name="eventTemplatePrototype">
        <sequence minOccurs="0" maxOccurs="unbounded">
            <element ref="field"/>
        </sequence>
        <attribute name="name" type="string" use="required"/>
        <attribute name="value" type="string" use="required"/>
    </complexType>

    <element name="event" type="eventPrototype"/>

    <element name="eventTemplate" type="eventTemplatePrototype"/>
</schema>
```

Figura 3.4: XML Schema para descrever o XML de eventos

4 Automatização

Atualmente, é comum que as empresas tenham cada vez mais atividades e processos envolvidos no seu negócio. Esses procedimentos tendem a ganhar complexidades cada vez maiores, gerando demandas, inclusive humanas, para que os objetivos e metas sejam alcançados com êxito. No entanto, o aumento da mão-de-obra nos processos pode não alcançar um desempenho satisfatório, por fatores que incluem desde a maior possibilidade de inserção de erros até a dificuldade de coordenação na excecução das tarefas. Tendo em vista maximizar performance dos empreendimentos e até mesmo aumentar a capacidade de produção surge a necessidade de implementação de processos automatizados em vários meios de produção. A automatização de processos procura reproduzir de forma sintética a capacidade humana de executar uma tarefa e, em seguida, executá-la com auxílio da computação. (28)

A implementação de um sistema automatizado gera vários benefícios, dentre eles está a busca pela redução de erros que podem ser gerados por distração, imperícia, entre outros fatores. Além disso, normalmente após a implantação do sistema, os custos de manutenção tendem a ser nulos ou baixos se comparados com o cumprimento de todas as exigências para a manutenção de um colaborador. Outro fator que favorece a automatização é a necessidade prévia de uma análise precisa das atividades realizadas, possibilitando otimizações das atividades. Por último têm-se a garantia de uma maior padronização dos processos.

Quando aplicado ao cenário do comércio eletrônico há enorme possibilidade de aplicação da automatização. Através, por exemplo, do preenchimento automático de vários campos, interfaces limpas, mais intuitivas e diretas podem ser alcançadas. Consequentemente, o aumento na objetividade das interfaces pode reduzir os erros inseridos no sistema, que poderiam gerar informações inconsistentes e consequentemente o retrabalho. Assim, podem, eventualmente, serem alcançados ganhos na redução de horas trabalhadas em determinadas atividades, obtendo maior produção e menor necessidade de mão-de-obra.

4.1 Automatização aplicada ao Magento

No Magento, a automatização pode evitar a inserção informações de forma análoga à artesanal, com as vantagens citadas que incluem a diminuição do trabalho e a eficiência das operações, minimizando o tempo gasto, bem como os erros cometidos.

O Magento permite dezesseis campos para detalhes de produtos, um campo para imagens e vídeos, quatro para otimização de pesquisa, quatro para preços avançados, nove opções para inventário avançado, um campo para opções customizadas, o mesmo valor que é disponibilizado para produtos relacionados, up-sell e cross-sell além de seis campos para o design e configurações automáticas, totalizando cinquenta campos que devem ser preenchidos.

Através do facilitador proposto, a operação de inserção passa a precisar de apenas três campos por parte do utilizador, sendo um deles a categoria, um campo que não existe no Magento, mas é construído aqui para a escolha do arquivo de templates, especificado segundo apresentado no Capítulo 3. A Figura 4.1 apresenta a tela de inserção de produtos no dispositivo móvel.

Figura 4.1: Interface para inclusão de produtos

A tela da aplicação apresentada na Figura 4.1, adicionalmente a informação da categoria, contém o nome e o valor do produto a ser incluído. A categoria é escolhida a partir do arquivo de templates relacionado à aplicação. Na figura, a categoria Periférico é selecionada. Obtêm-se então uma interface bastante simples para adição de produtos, mas que garante uma grande eficácia na automatização, dado a seleção da categoria vários dados são preenchidos no sistema de acordo com a categoria proposta.

Outra funcionalidade de automação que o facilitador implementa é a recuperação de produtos no sistema. Através do evento do toque no botão "Obter Produtos" todos os produtos podem ser retornados. A interface em funcionamento é ilustrada pela Figura 4.2.

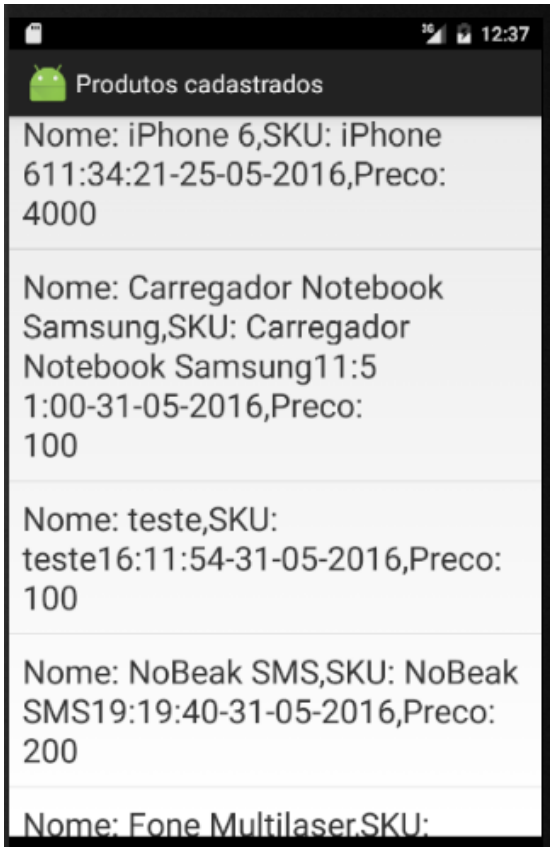

Figura 4.2: Interface para recuperação de produtos

Há então um acesso rápido aos principais dados de produtos pertencentes à loja virtual usando componentes nativos do Android para listagem.

Ao se clicar em um dos produtos listados é possível então editar o produto de modo singular, sendo aberta uma interface com os dados contidos na tela anterior em campos editáveis. Há botões então para que o usuário possa salvar sua mudanças ou

então remover por completo o produto do Magento, como visto na Figura 4.3.

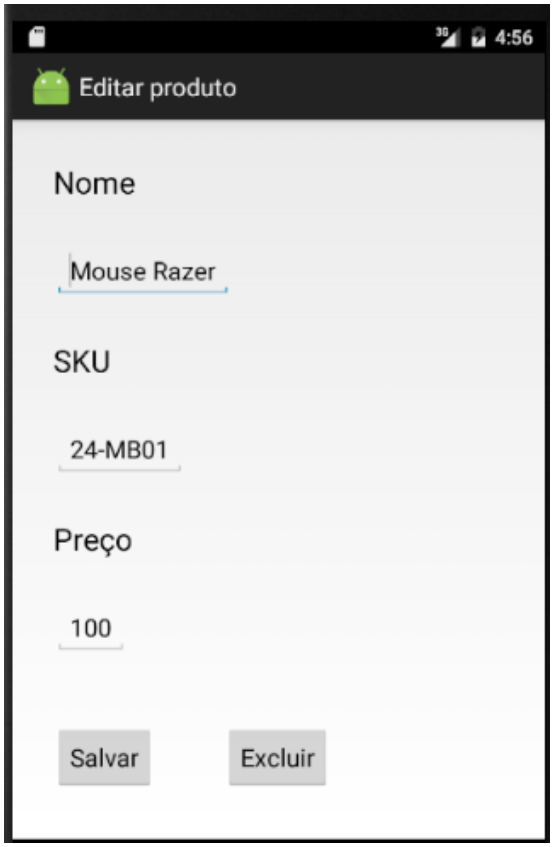

Figura 4.3: Interface para edição e exclusão de produtos

As interfaces usadas no facilitador proporcionam, em conjunto, além de uma simplificação do uso, a automatização de funções que podem ser demoradas ou apresentar complexidade alta para o usuário.

5 Adaptações

Em cenários que apresentam constante modificação e são altamente dinâmicos, como é o caso da computação móvel, a adaptabilidade é, usualmente, um requisito importante para as aplicações.

No contexto das aplicações, a adaptabilidade é usualmente mencionada para referenciar tanto sistemas adaptativos quanto adaptáveis(29), frequentemente sem distinção entre eles. Entretanto, um sistema é dito adaptativo se é capaz de mudar o seu comportamento automaticamente, durante sua execução, de acordo com mudanças no ambiente em que ele está inserido. Por outro lado, um sistema adaptável não está relacionado à auto-adaptação, mas a capacidade de adaptar uma estrutura completa em partes específicas, devido à mudanças.

Diversas operações podem ser realizadas com o objetivo de adaptar uma aplicação, seja de forma automática, de acordo com mudanças inesperadas como usualmente acontece nas aplicações hipermídia, ou ainda de forma pré-definida. É possível, por exemplo, focar as ações adaptativas na mudança do conteúdo das aplicações. Essa estratégia, usualmente mencionada como transformação, busca modificar o conteúdo sem perdas significativas de informações. Também é possível, ainda em relação ao tratamento do conteúdo, realizar ações de substituição. Nesse outro caso, um dado alternativo é aplicado para substituir o conteúdo original.

É esperado que as ações mencionadas sejam usualmente mais eficientes nas aplicações hipermídia, onde as mídias podem corresponder a uma parte significativa das aplicações. Nas aplicações móveis em geral, por outro lado, técnicas que aplicam a filtragem do conteúdo podem ser mais interessantes, ao reduzir as informações em um determinado momento àquelas que o usuário necessita.

Pode ser considerado, como exemplo, uma aplicação de correio eletrônico. Nesse tipo de aplicação, os e-mails podem ser separados pela origem. Assim, e-mails de anunciantes conhecidos podem ser separados dos demais e, em particular, daqueles que fazem parte da lista de contatos do usuário. Também podem ser analisados os anexos que

podem, por exemplo, ser trazidos apenas sob demanda. Diversas outras estratégias de filtragem podem ser empregadas.

A filtragem do conteúdo pode, portanto, prover diferentes estratégias de adaptação às aplicações. É necessário, no entanto, além de conhecer o conteúdo a ser adaptado, estabelecer estratégias para a implementação da filtragem. Neste trabalho, a estratégia inicial é focada nas vendas. O facilitador permite ao usuário filtrar os produtos com baixa quantidade de estoque, para que, por exemplo, sejam estabelecidas estratégias de aquisição de produtos entre fornecedores, promovendo menor lead time, menor possibilidade de esgotamento e previsão de gastos, além de várias outras finalidades práticas relacionadas ao negócio.

A funcionalidade pode ser proporcionada ao sistema através da API REST usando a requisição catalogInventoryStockRegistryV1, usando o caminho /V1/stockItems/lowStock/ aplicando o método HTTP GET. A execução da requisição pela interface de requisição do facilitador obtém então uma resposta do seguinte formato segundo as Figuras 5.1 e 5.2:

Figura 5.1: Formato da resposta JSON de produto em estoque - A

A Figura 5.3 ,por sua vez, apresenta a tela de filtragem de conteúdo, retornando

Figura 5.2: Formato da resposta JSON de produto em estoque - B

somente os produtos considerados com baixa quantidade no estoque para a interface. Para este fim foi considerado o valor de 15 unidades em estoque como o limiar do filtro. Valores iguais ou inferiores ao definido serão incluídos na lista e referenciados com seu id único e quantidade atual declarada no sistema. Valores associados ao campo "qty" maiores que quinze serão desconsiderados pela aplicação nesta função.

Dado a resposta do sistema em array de produtos aplica-se então o conceito de filtragem de conteúdo para executar o processamento somente com os dados relevantes para a atividade. De acordo com o contexto atual e a semântica, têm-se necessário os campos "product_id" que se refere ao produto no sistema e "qty" que retorna a quantidade do produto no estoque.

Para os produtos filtrados, o preço pode ser acrescido em um determinado percentual, seguindo a ideia de oferta e procura. Adicionalmente, podem ser estabelecidas outras estratégias, como a construção de cupons baseados em datas especiais, conforme será tratado na próxima seção.

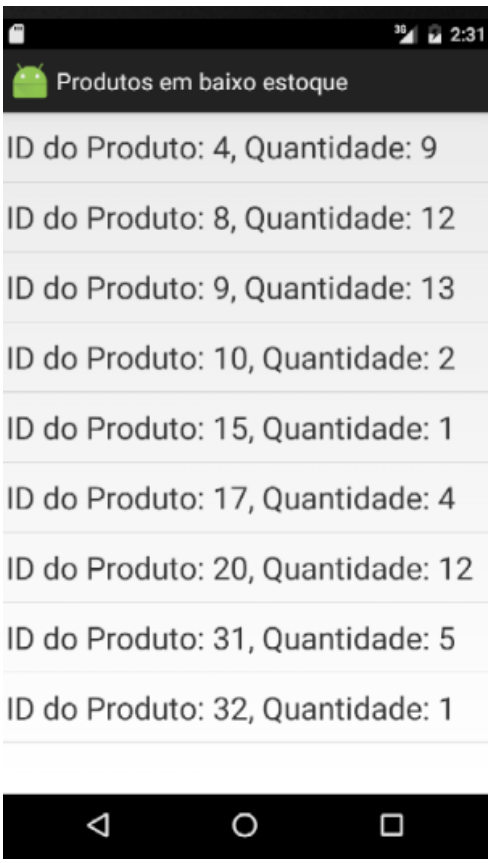

Figura 5.3: Interface para recuperação de produtos com baixo estoque

5.1 Inferência na adaptação

A inferência é a operação intelectual em que se pode afirmar algo ou criar um conhecimento do que acontece em um dado universo a partir da realização de análise sobre uma proposição ou situação. Essa situação pode ser vista como aprendizagem por parte do sistema na busca de informações mais condizentes com a realidade de uso por parte do cliente. A apresentação de uma interface humano-computador que busca atender as demandas de uso mais recorrentes por meio do uso de dados disponíveis pela aplicação também faz parte da aprendizagem por parte do sistema.

O uso da inferência permite ao sistema oferecer facilidades para o usuário na proposição de novas formas de uso. As ferramentas relacionadas a lojas virtuais, como o Magento, podem gerar uma enorme quantidade de dados, os quais nem sempre são transformados em informações de real aproveitamento. Dentre as possibilidades de inferência relacionadas às lojas virtuais talvez as mais relevantes sejam aquelas que buscam explorar o contexto temporal. Afinal, relacionar eventos ou sugestões a datas importantes pode

Figura 5.4: Sugestão de desconto para a data atual

ser de grande importância para a gerência de um e-commerce.

Neste trabalho é explorado o tempo, relacionado a datas específicas, como gatilho para sugestões de promoções. Próximo a essas datas, o usuário administrador é alertado, sendo sugerida a criação de cupons de desconto. Como exemplo, às vésperas do feriado referente ao natal uma mensagem sugerindo a criação de um cupom de desconto é exibido para alertar o usuário, conforme apresentado na Figura 5.4 para o dia em que o teste foi feito, no inuito de demonstrar a facilidade.

O desenvolvimento dessa facilidade é implementado através da REST API com o salesRuleRuleRepositoryV1 encontrado na documentação. O caminho que disponibiliza a realização é /V1/salesRules e o método HTTP utilizado para criá-lo é POST.

A comunicação é feita, como as demais, via JSON seguindo os formatos previamente definidos pela API REST Magento Swagger de acordo com a estrutura exibida na Figura 5.5.

A Tabela 5.1 explica o uso de cada atributo nesta requisição de comunicação com

Inline Model {
rule (sales-rule-data-rule-interface)
}
sales-rule-data-rule-interface {
rule_id (integer, *optional*): Rule id,
name (string, *optional*): Rule name,
store_labels (Array[sales-rule-data-rule-label-interface], *optional*): Display label,
description (string, *optional*): Description,
website_ids (Array[integer]): A list of websites the rule applies to,
customer_group_ids (Array[integer]): Ids of customer groups that the rule applies to,
from_date (string, *optional*): The start date when the coupon is active,
to_date (string, *optional*): The end date when the coupon is active,
uses_per_customer (integer): Number of uses per customer,
is_active (boolean): The coupon is active,
condition (sales-rule-data-condition-interface, *optional*),
action_condition (sales-rule-data-condition-interface, *optional*),
stop_rules_processing (boolean): To stop rule processing,
is_advanced (boolean): Is this field needed,
product_ids (Array[integer], *optional*): Product ids,
sort_order (integer): Sort order,
simple_action (string, *optional*): Simple action of the rule,
discount_amount (number): Discount amount,
discount_qty (number, *optional*): Maximum qty discount is applied,
discount_step (integer): Discount step,
apply_to_shipping (boolean): The rule applies to shipping,
times_used (integer): How many times the rule has been used,
is_rss (boolean): Whether the rule is in RSS,
coupon_type (string): Coupon type,
use_auto_generation (boolean): To auto generate coupon,
uses_per_coupon (integer): Limit of uses per coupon,
simple_free_shipping (string, *optional*): To grant free shipping,
extension_attributes (sales-rule-data-rule-extension-interface, *optional*)
}
sales-rule-data-rule-label-interface {
store_id (integer): StoreId,
store_label (string): The label for the store,
extension_attributes (sales-rule-data-rule-label-extension-interface, *optional*)
}
sales-rule-data-condition-interface {
condition_type (string): Condition type,
conditions (Array[sales-rule-data-condition-interface], *optional*): List of conditions,
aggregator_type (string, *optional*): The aggregator type,
operator (string): The operator of the condition,
attribute_name (string, *optional*): The attribute name of the condition,
value (string): The value of the condition,
extension_attributes (sales-rule-data-condition-extension-interface, *optional*)
}
sales-rule-data-rule-extension-interface {
}
sales-rule-data-rule-label-extension-interface {
}
sales-rule-data-condition-extension-interface {
}

Figura 5.5: Estrutura para criação de um cupom de desconto

Tabela 5.1: Definição dos campos na solicitação JSON para cupons

Campo	Descrição
name	Nome da regra de desconto
websiteIds	Código de websites que usaram a regra (Home/Pesquisa/Externo..)
customerGroupIds	Grupo de clientes (Logado, Deslogado, em página específica)
usesPerCustomer	Uso máximo por cliente
isActive	Campo ativo ou não
stopRulesProcessing	Processar ou não regras de menor prioridade
isAdvanced	Possui ou não detalhes pra regra
sortOrder	Será ordenado pelo sistema
discountAmount	Quantia de desconto, por default em porcentagem
discountStep	Quantas vezes o cupom pode ser usado em um carrinho
applytoShipping	O deseconto se aplica ou não ao valor de frete
timesUsed	Quantidade de vezes que o cupom pode ser usado no sistema
isRss	Publicar ou não no RSS
couponType	Tipo de cupom: "SPECIFIC_COUPON"ou "AUTO"
useAutoGeneration	Gera automaticamente o cupom, sem necessidade de inserir código
usesPerCoupon	Quantidade de vezes um usuário pode usar um cupom

o sistema, apresentando suas respectivas semânticas.

Esses campos foram escolhidos dentre os vários possíveis para uso pois são os de interesse de estudo com o fim de concretizar a criação sem informações demasiadas ou desnecessárias. Os parâmetros que não foram preenchidos ou são de baixa importância ou são automaticamente preenchidos quando há comunicação da API com o sistema.

A execução do comando não gera uma resposta em caso de sucesso, porém, podem ser gerados erros do tipo 400 (Bad request) em caso de má formação da requisição por parte do sistema, 401 (Unauthorized) em caso de falha na autenticação, 500 (Internal Server error) se houver falha por lado do servidor e também mensagem padrão, caso aconteça um erro inesperado.

Sendo possível então a criação de um cupom de desconto pelo sistema, estabelece-se um monitoramento da data ao iniciar o aplicativo de forma a verificar a data atual. Caso ela esteja em uma das datas estabelecidas como interessantes para o usuário do facilitador, a aplicação dispara uma mensagem no intuito de sugeri-lo a implementação de tal recurso no sistema.

Um exemplo prático e recorrente de aplicação seria, em véspera natalina, a recomendação do estabelecimento de um cupom de descontos para a época e caso haja interesse, o sistema possibilita então a definição do mesmo com agilidade, praticidade e

de forma intuitiva, tendo em vista que há necessidade de preenchimento de 23 campos, ao contraste do proposto pelo trabalho atual que prevê somente empenho para escolher o percentual de desconto a ser promovido pela loja virtual.

6 Conclusões

Este trabalho envolveu uma análise de uma das principais aplicações e-commerce existentes na atualidade, o Magento, com o objetivo de transformar o seu uso em uma atividade mais simples, direta, rápida e automatizada, com a filtragem de vários campos que muitas das vezes são desnecessários em seu uso cotidiano, minimizando dúvidas sobre o manuseio da ferramenta, gastos excessivos de tempo para a execução de uma atividade manual e a possível inserção de erros.

Um dos pontos principais do trabalho foi a oportunidade de uso de templates em XML para a descrição de categorias e eventos. O uso dessa tecnologia teve como objetivo favorecer a automatização de processos relacionados ao software, com destaque para a inserção de produtos. Na solução proposta, objetos pertencentes a uma mesma categoria têm propriedades semelhantes, permitindo diminuir grande parte dos campos que seriam inseridos manualmente na loja virtual.

Na verdade, o uso de templates está diretamente relacionado a propostas de automatização e adaptação. O uso de modelos para descrever eventos temporais tais como datas importantes permitiu que o sistema reconhecesse e informasse ao administrador, em tempo hábil, um acontecimento temporal e uma ação subsequente. No trabalho as ações envolveram a sugestão de implementação de descontos.

O uso da inferência e da adaptação mostrou-se necessário na busca das facilidades que constituem objetivos deste trabalho. Isto é observável no contexto da aplicação devido às facilidades que são providas por elas. Como exemplo, os templates, quando usados para especificar as datas importantes, podem gerar cupons de desconto, estimulando assim as vendas.

Uma considerável contribuição deste trabalho é a adaptação da parte administrativa de um e-commerce para o ambiente móvel. Nesse aspecto, foram explorados mecanismos para permitir que as limitações da interface móvel não interferissem na administração da plataforma. Assim, pode-se concluir que os softwares de comércio eletrônico atuais, apesar de robustos e complexos, são passíveis de melhorias em suas interfaces e na execu-

ção de atividades com menos esforços.

Uma limitação do trabalho desenvolvido é a obtenção, envio, alteração e remoção de informações totalmente dependente de API provida pelo desenvolvedor. Muitas das vezes o projeto precisou de ser remodelado e funções ganharam complexidades maiores ou até se tornaram inviáveis devido a modicidade da interface e a impossibilidade de alterar os dados do sistema diretamente em seu núcleo.

Como forma de corrigir tais obstáculos e, portanto, uma proposta de trabalho futuro, seria a ampliação da API REST disponibilizada pela empresa Magento, trazendo novas funcionalidades voltadas para o lado do administrador e possibilitando uma recuperação mais fácil de informações contidas no sistema.

Outra proposta de trabalho futuro é o preenchimento dos arquétipos por uma interface gráfica disponível pela web, tornando a tarefa mais simples e amigável ao usuário.

O trabalho também é limitado quanto a inserção de dados via facilitador, pois campos como descrição, vídeos, seções como 'Search Engine Optimization", "Advanced Pricing", "Advanced Inventory", "Related Products", "Up-sells", "Cross-sells" e "Designs" não são preenchidos tampouco é permitido fazer a inserção pelo resultado final deste trabalho.

Os códigos usados na maioria dos testes mencionados neste texto e que foram desenvolvidos durante este trabalho estão disponíveis no GitHub[1] e aberto para colaborações(30)

[1]https://github.com/

Referências Bibliográficas

[1] A. Molla and P. S. Licker, "eCommerce adoption in developing countries: a model and instrument," *Information & management*, vol. 42, no. 6, pp. 877–899, 2005.

[2] "Magento, a plataforma de e-commerce mais popular do mundo." http://allcash.com.br/magento/magento-a-plataforma-de-e-commerce-mais-popular-do-mundo/. Acessado em: 29/01/2016.

[3] A. G. M. F. D. VOLTOLINI *et al.*, "Na palma da mão: A difusão de celulares e smartphones e possibilidades para o ensino-aprendizagem no brasil," 2016.

[4] K. Figueiredo, "A logística do pós-venda," *Revista Tecnologística*, vol. 8, no. 80, 2002.

[5] "W3C HTML." https://www.w3.org/html/. Visitado em 29/07/2016.

[6] "Cascading Style Sheets." https://www.w3.org/Style/CSS/. Visitado em 29/07/2016.

[7] "JavaScript." https://www.javascript.com/. Visitado em 29/07/2016.

[8] "PHP:Hypertext Preprocessor." http://php.net. Visitado em 29/07/2016.

[9] "ASP.NET | The ASP.NET Site." http://www.asp.net/. Visitado em 29/07/2016.

[10] "Desenvolvimento Mobile: WEb App vs Native App." http://zarbsolution.com.br/desenvolvimento-mobile-web-app-vs-native-app/. Acessado em 13/01/2016.

[11] A. Al-Bar and I. Wakeman, "A survey of adaptive applications in mobile computing," in *Distributed Computing Systems Workshop, 2001 International Conference on*, pp. 246–251, IEEE, 2001.

[12] M. Satyanarayanan, "From the editor in chief: the many faces of adaptation," *IEEE Pervasive Computing*, no. 3, pp. 4–5, 2004.

[13] "Android 6.0 Marshmallow: all the key features explained." https://www.androidpit.com/android-m-release-date-news-features-name. Acessado em 29/07/2016.

[14] "Method Definitions." https://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html. Acessado em 29/07/2016.

[15] A. Parikh and S. Deshmukh, "Search engine optimization," in *International Journal of Engineering Research and Technology*, vol. 2, ESRSA Publications, 2013.

[16] A. Jalali, A. Nalawade, K. Kulkarni, and S. Mishra, "Mobile CMS Platform for Android,"

[17] M. Zur Muehlen, J. V. Nickerson, and K. D. Swenson, "Developing web services choreography standards—the case of rest vs. soap," *Decision Support Systems*, vol. 40, no. 1, pp. 9–29, 2005.

[18] M.-D. Cohen, "Exploiting response models—optimizing cross-sell and up-sell opportunities in banking," *Information Systems*, vol. 29, no. 4, pp. 327–341, 2004.

[19] R. C. Blattberg and J. Deighton, "Manage marketing by the customer equity test," *Harvard business review*, vol. 74, no. 4, p. 136, 1996.

[20] P. Brusilovsky, "Methods and Techniques of Adaptive Hypermidia," in *User Modeling and User adapted Interaction. Special issue on adaptive hypertext and hypermidia*, 1996.

[21] L. A. M. Palazzo, "Modelos proativos para hipermídia adaptativa," *Porto Alegre*, 2000.

[22] "Significado de Arquétipo." http://www.lexico.pt/arquetipo/, note=Visitado em 29/07/2016.

[23] "O que são templates?." https://screencorp.zendesk.com/hc/pt-br/articles/201227124-O-que-s%C3%A3o-templates-. Acessado: 10/05/2016.

[24] A. S. Tanenbaum, *Redes de computadoras*. Pearson Educación, 2003.

[25] "XML Tutorial." http://www.w3schools.com/xml/. Acessado em: 13/07/2016.

[26] W. M. Zintel, A. S. Gandhi, Y. Gu, S. Pather, J. C. Schlimmer, C. M. Rude, D. R. Weisman, D. R. Ryan, P. J. Leach, T. Cai, *et al.*, "XML-based template language for devices and services," June 21 2005. US Patent 6,910,068.

[27] V. Somppi, "System, method, device, and computer code product for implementing an XML template," May 5 2004. US Patent App. 10/840,392.

[28] S. Zuboff, "Automatizar/informatizar: as duas faces da tecnologia inteligente," *Revista de administração de empresas*, vol. 34, no. 6, pp. 80–91, 1994.

[29] S. Y. Chen and G. D. Magoulas, *Adaptable and adaptive hypermedia systems*. IGI Global, 2005.

[30] "Facilitador Magento." https://github.com/eduardodeasousa/appeduardo.