1
Introdução

Vivemos hoje na era da Web 2.0, onde os navegadores executam interfaces gráficas cada vez mais ricas, permitindo que virtualmente todo tipo de aplicação possa explorar a ubiqüidade dos navegadores Web sem comprometer a experiência do usuário: sites de relacionamento, sistemas online de reserva de passagens, sites de compras ou de leilão, editores de texto colaborativos, são apenas alguns exemplos de aplicações construídas sobre clientes Web ricos.

Métodos de projeto de aplicações hipermídia, tais como o SHDM (*Semantic Hypermedia Design Method*)[2], têm por objetivo introduzir modelos em diferentes camadas de abstração para capturar os requisitos e especificar o funcionamento das aplicações sob várias perspectivas. Por exemplo, a modelagem conceitual, a navegação e a apresentação são aspectos comumente tratados de forma separada pelas metodologias de desenvolvimento para a Web.

A separação entre lógica e apresentação é também preconizada pelo padrão MVC (*Model-View-Controler*), amplamente utilizado no desenvolvimento de aplicações Web. A aplicação deste padrão de projeto favorece o reuso de código através de diferentes plataformas. Seja o dispositivo de interface um microcomputador ou um telefone celular, a camada de apresentação implementa múltiplas visões sobre a mesma camada de lógica de negócio.

Os modelos de visualização para os aplicativos Web de primeira geração eram suficientes ao representar apenas a *estrutura* e o *formato* das apresentações, traduzíveis em documentos HTML e folhas de estilo CSS. No contexto da Web 2.0, entretanto, os elementos de interface das aplicações também são dotados de comportamento não definido pela linguagem HTML. Por exemplo, os controles evoluíram de botões de opção e caixas de verificação com os quais o usuário interagia de forma simples através de um clique de mouse para caixas de texto dotadas do recurso de autocompletar e tabelas reordenáveis. Surge a necessidade, portanto, de um vocabulário para definir o *comportamento* da interface, traduzível em códigos Javascript ou aplicações Flash ou Silverlight.

[2] O método SHDM é uma evolução do método OOHDM (*Object-Oriented Hypermedia Design Model*), que leva em consideração os formalismos e primitivas da Web Semântica. [1, 2]

A proposta deste trabalho consiste em definir uma linguagem de descrição, em alto nível, do funcionamento das interfaces RIA (*Rich Internet Application*), um processador para esta descrição, capaz de gerar o código executável da interface, e a respectiva máquina de *runtime* para executar as interfaces geradas.

A linguagem de descrição de interfaces visa, na verdade, estender a Ontologia de *Widgets* Abstratos e a Ontologia de *Widgets* Concretos, formalismos adotados pelo método SHDM [2], com a capacidade de expressar os objetos de interface também de um ponto de vista dinâmico, e não apenas em termos de seus atributos e composição. O presente trabalho também tem por objetivo contribuir com a evolução da ferramenta HyperDE, um ambiente de prototipação e desenvolvimento de aplicações Web baseado no método SHDM. Até o momento, o ambiente do HyperDE não apóia a modelagem de interfaces abstratas, ou seja, as interfaces das aplicações precisam ser codificadas diretamente na linguagem-alvo, Ruby. A nossa proposta inclui estender o HyperDE com a funcionalidade de interpretação de modelos abstratos de interfaces e geração automática de código.

1.1. Motivação

Avanços no projeto visual das aplicações Web e novas técnicas de programação em Javascript fizeram com que a experiência de interação na Web rivalizasse com a dos aplicativos *desktop*. Junte a isso os já conhecidos benefícios da computação baseada na Web: desenvolvimento, manutenção, atualização e escalabilidade centralizada da aplicação. Os exemplos mais populares de aplicativos que exploram recursos avançados de interface incluem: e-mails baseados em Web (Gmail, Yahoo! Mail e Hotmail), suítes de escritório e ferramentas de colaboração (Google Docs, Zoho, Office Live, Backpack e OpenRecord), serviços de mapas (Google Maps, Yahoo! Maps e MapQuest), compartilhamento de fotos (Flickr, Ipernity Oº, Picasa), entre outros.

A tecnologia dos *mashups* tornou a autoria de aplicações da Web 2.0 acessível aos usuários com pouca ou nenhuma experiência como desenvolvedores. Um *mashup* é um serviço que combina conteúdo de mais de uma fonte. Os componentes de um *mashup* são mini aplicações (um pedaço portável e autocontido de código HTML/Ajax), possivelmente desenvolvidas por terceiros, conhecidas como *widgets* ou *gadgets*, embora o termo *widget* possa se referir a qualquer objeto de interface, baseado em Ajax ou não, parte de um *mashup* ou não. O criador de um *mashup* irá explorar as interações entre seus componentes como forma de trazer uma nova experiência para o usuário. Por exemplo, em uma aplicação que utiliza as APIs

disponibilizadas pela Amazon.com e pelo YouTube, o usuário poderá visualizar não somente as faixas de um álbum selecionado, mas também assistir aos vídeos relacionados no YouTube. Este é o caso do ZonTube.

Considere também a categoria das aplicações onde é possível tirar partido do paralelismo no processamento de uma requisição do usuário. Este é o caso dos sites que oferecem acesso a diversos serviços Web de forma centralizada. Um exemplo típico seria uma aplicação Web cujo objetivo é ajudar os usuários a comparar preços de passagens aéreas, como o site Alibabuy.com. Seus requisitos funcionais incluem capturar os parâmetros de pesquisa informados pelos usuários (cidades de origem e destino, data de partida e retorno, número de passageiros, etc.) e, em seguida, submeter consultas a sites de agências de viagens, esperar pelas respostas, ordená-las e exibi-las para o usuário. Esta aplicação típica tem com característica intrínseca o paralelismo das consultas aos serviços Web das empresas parceiras. Aqui, o designer da aplicação pode optar por a interface ser atualizada de uma só vez, após o retorno de todas as chamadas aos serviços Web (“serialização” das chamadas), ou por oferecer ao usuário uma experiência de interatividade mais rica: atualizar a interface incrementalmente, no retorno de cada chamada. Este é um cenário diferente daquele presente nos *mashups*: neste último, diversos provedores de serviços oferecem componentes de interface que são integrados na camada de Visão; no primeiro, a combinação de várias fontes de dados é presumivelmente feita na camada de Modelo.

A motivação para a realização deste trabalho surgiu da ausência, no método SHDM, de uma linguagem para modelar o funcionamento das interfaces neste último tipo de aplicação, onde não somente estão presentes controles sofisticados, mas também onde se faz desejável comunicar aos usuários eventos ocorridos na camada de Modelo. Todavia, o foco aqui não é dar suporte ao desenvolvimento baseado em *mashups*.

1.2. Objetivos e Contribuições Esperadas

Este trabalho tem por objetivo suprir o método SHDM com um formalismo para expressar o funcionamento das interfaces presentes nos sites da Web 2.0. Este formalismo tem como requisitos:

a) Representar os controles de interface ricos mais comuns (barras de progresso, medidores, carrosséis de imagens, etc...);
b) Capturar as interações dos usuários com os objetos de interface (clicar, clicar-e-arrastar, selecionar);

c) Especificar as decorações sobre os objetos de interface (decorações de entrada, saída, ênfase e manutenção) como respostas às mudanças de estado da aplicação ou da interface. Este requisito pressupõe a existência de uma arquitetura que permita que as atualizações na camada de Modelo sejam comunicadas à camada de Visão da aplicação.

A proposta não se limita a estender o método SHDM com uma ontologia para modelar o comportamento das interfaces, mas também estender o ambiente HyperDE com a capacidade de auxiliar a autoria de instâncias desta ontologia e de gerar, a partir destas descrições abstratas, as interfaces concretas (executáveis). Por este motivo, a linguagem de descrição de interfaces abstratas tem como requisitos adicionais:

a) A linguagem deve oferecer um mecanismo de mapeamento entre a descrição abstrata dos elementos de interface e sua representação concreta.
b) A linguagem deve prover um mecanismo de integração com as linguagens HTML e CSS.

A solução não estaria completa, portanto, sem a especificação de uma linguagem para descrever o funcionamento das interfaces concretas, e de uma máquina para executar estas descrições. Por dar suporte ao desenvolvimento Web baseado no método SHDM, o ambiente HyperDE foi escolhido como a plataforma-alvo para a geração de interfaces concretas. O ambiente de execução de interfaces concretas a ser integrado ao HyperDE introduz um protocolo assíncrono baseado em fila de mensagens como forma de implementar a comunicação entre as camadas de Modelo e Visão.

Em uma aplicação Web convencional, quando o cliente faz uma requisição, o efeito produzido é o de uma chamada síncrona, ou seja, o usuário tem de aguardar a atualização da página antes de poder voltar a interagir com a aplicação. Por outro lado, a tecnologia Ajax permite a comunicação assíncrona entre cliente e servidor, o que previne o usuário da sensação de “congelamento” a cada requisição submetida. Na arquitetura Ajax, diferentes elementos da camada de Visão podem ser atualizados de forma independente. Contudo, a arquitetura aqui proposta tem por objetivo viabilizar um grau ainda maior de assincronismo. Nela, a camada de Visão pode ser atualizada incrementalmente com os resultados parciais do processamento de uma requisição, ou seja, o usuário não precisa esperar que o processamento de sua consulta seja concluído antes de receber qualquer retorno.

Em resumo, esperamos que a presente pesquisa contribua para:

- Atualizar a ontologia de descrição de interfaces do método SHDM para acompanhar a evolução dos recursos de interação na Web;

- Prover uma ferramenta de auxílio à criação de instâncias desta nova versão da ontologia;
- Prover uma ferramenta capaz de transformar as instâncias desta ontologia em interfaces executáveis;
- Propor uma arquitetura assíncrona como forma de simular a comunicação bidirecional entre cliente e servidor.

A **Arquitetura SWUI de Interfaces Ricas** refere-se ao inteiro conjunto de componentes de *software* e tecnologias empregadas para a modelagem, geração e execução de interfaces, conforme proposto por essa pesquisa. A **Ontologia de Descrição de Interfaces Ricas** é o principal componente desta arquitetura. O acrônimo **SWUI** (*Semantic Web User Interface*) é utilizado como abreviação para o *namespace* URI associado aos identificadores dos recursos descritos na ontologia.

1.3. Estrutura da Dissertação

O presente documento está estruturado como segue. O segundo capítulo apresenta uma revisão bibliográfica sobre a tecnologia RIA e o Desenvolvimento Dirigido por Modelos aplicado à construção de interfaces ricas.

O terceiro capítulo desenvolve o componente principal da proposta desta pesquisa, a saber, uma linguagem com expressividade para modelar os aspectos que caracterizam as interfaces RIA, mas baseada nos atuais formalismos do método SHDM para a modelagem de interfaces: a **Ontologia de *Widgets* Abstratos** e a **Ontologia de *Widgets* Concretos**.

O tema do quarto capítulo é a Geração Automatizada de Interfaces. Seu objetivo descreve os requisitos de um projeto de software para dar apoio à construção de interfaces RIA, utilizando a linguagem introduzida no capítulo 3 como instrumento de especificação. No quinto capítulo, documentamos a implementação do ambiente de modelagem, geração e execução de interfaces RIA dentro do HyperDE.

No sexto capítulo são apresentados alguns estudos de caso que exemplificam a aplicação do método de especificação de interfaces proposto neste trabalho, e avalia a sua eficácia como modelo para expressar o comportamento da interface em reação a eventos internos (de sistema) e externos (de usuário).

Finalmente, no último capítulo, avaliamos as contribuições alcançadas por meio da pesquisa e apontamos os trabalhos futuros.