Um Módulo de Fusão de Dados para Mashups

Oliver Moraes Batista[1], **William Komura**[1], **Hugo Bulegon**[1], **Carmem S. Hara**[1]

[1]Departamento de Informática – Universidade Federal do Paraná
Caixa Postal 19081 – 81531-980 – Curitiba – PR – Brasil

Resumo. *Web mashups são aplicações web que integram conteúdo de diversas fontes de dados disponibilizadas por terceiros através de uma interface de serviço. Para permitir que elas possam ser construídas por profissionais que não possuem a habilidade de programação, existem diversas ferramentas que facilitam a combinação de serviços existentes através de uma interface simples e intuitiva. Dentre estas ferramentas pode ser citado o Exhibit. Através de uma análise das funcionalidades disponibilizadas por esta ferramenta, observou-se que ela possui capacidade limitada para a integração de dados sobrepostos. Ou seja, se a mesma informação é disponibilizada por mais de uma fonte, esta ferramenta não fornece meios adequados para combinar os dados e resolver possíveis conflitos de valor. Motivado por esta deficiência, neste trabalho é proposto um novo módulo para a ferramenta Exhibit, chamado de* Mashup Importer*, que permite criar mapeamentos de dados provenientes de diferentes fontes que indicam que eles se referem ao mesmo item de dado. Este módulo de fusão de dados pode ser combinado com as demais funcionalidades já existentes na ferramenta para a criação de novos serviços web.*

1. Introdução

Com a popularização da Internet, serviços de diversas naturezas foram criados para atender seus usuários, tais como portais de notícias, redes sociais e serviços de geolocalização. Consequentemente, a quantidade de informações disponíveis tem aumentado constantemente. Grandes empresas como Facebook, Twitter, Google e Last.fm possuem um grande volume de dados e de escopo mundial, devido principalmente aos usuários de seus serviços que a cada dia fornecem novas informações. Alguns serviços web passaram a disponibilizar esses dados para que terceiros os utilizem e dessa forma aumentar a interatividade com os serviços já existentes. Hoje é comum que a partir de um portal de notícias seja possível compartilhar uma informação encontrada em uma rede social, ou que o local de um evento informado em um serviço de shows musicais seja registrado usando outro serviço, como por exemplo de geolocalização.

O termo web mashup denota um novo gênero de aplicações web capazes de integrar o conteúdo de fontes independentes. Um mashup é geralmente definido como uma aplicação web apta a agregar múltiplos componentes, que são dados ou funcionalidades de aplicativos, criados por terceiros e acessados via APIs, que servem para o desenvolvimento rápido de aplicativos de curta duração ou situacionais [Bianchini et al. 2010]. De acordo com [Tuchinda et al. 2011], a construção de um mashup envolve cinco etapas: extração de dados de diferentes fontes, transformação dos dados, limpeza para solução de inconsistências de valores e formato, integração e apresentação dos dados de forma integrada em uma interface Web. Exceto pela última etapa, que trata da apresentação da resultado, percebe-se que as etapas são idênticas a um processo de integração de dados.

O que distingue os mashups é que eles tem como objetivo permitir a construção de novos serviços web e integração de informações por profissionais que não possuem a habilidade de programação.

Diversas ferramentas e ambientes de geração de mashups foram desenvolvidos para que usuários pudessem criá-los facilmente. Dentre elas existem soluções como Yahoo Pipes![1], IBM Damia agregado à solução IBM Mashup Hub[2], Apatar[3], Karma [Tuchinda et al. 2011] e Exhibit[4]. Estas ferramentas diferem na ênfase que dão em cada uma das etapas da construção de um mashup bem como na interface do usuário oferecida para o seu desenvolvimento.

Segundo [Zang and Rosson 2009], há dois tipos principais de interfaces: baseado em *workflow* e programação baseada em exemplos. As ferramentas Yahoo Pipes, Mashup Hub e Apatar seguem a abordagem de workflows, enquanto o Karma e Exhibit utilizam a programação baseada em exemplos. Algumas das ferramentas para geração de mashups são proprietárias, como o Mashup Hub, outras geram mashups que ficam hospedadas apenas nos servidores da empresa que desenvolveu a ferramenta, como o Yahoo Pipes e outras não disponibilizam o código, como o Karma. Dentre as de código livre, encontram-se o Apatar e o Exhibit.

Embora todas as ferramentas tenham sido concebidas para que usuários com pouco ou nenhum treinamento em programação pudessem criar novos serviços a partir da integração de serviços já existentes, o estudo reportado em [Zang and Rosson 2009] mostra que mesmo com uma interface gráfica intuitiva como o Yahoo Pipes, usuários apresentam dificuldade para assimilar noções como uma iteração sobre um conjunto (`Loop`). Por outro lado, o entendimento de documentos XML e *RSS feeds* foi relativamente simples e a maioria dos usuários foi capaz de compreender o significado de algumas linhas de código, baseado no nome dos elementos (*tags*). Embora o estudo conclua que as ferramentas analisadas não estejam suficientemente maduras para permitir que usuários leigos construam mashups, a facilidade de compreensão de pequenos trechos de código mostram o potencial de ferramentas baseadas em exemplo.

Como o Exhibit é uma ferramenta de código livre e que segue a abordagem de programação por exemplo, ele foi escolhido para o desenvolvimento deste trabalho. Embora o Exhibit apresente diversas facilidades para a composição de serviços, constatou-se que o suporte dado para a fusão de dados, ou seja, dados que referem-se a uma mesma entidade no mundo real, é limitado. Para exemplificar, considere dois serviços que tenham como saída arquivos JSON possuindo informações sobre eventos musicais, como nas Figura 1(a) e Figura 1(b). Deseja-se criar um mashup que agrega informações de ambos, para fornecer uma informação mais completa sobre eventos que estão por acontecer.

Suponha que ambas as fontes possuem campos identificadores como por exemplo, *called* na Fonte 1 e *name* na Fonte 2. Quando ambas as fontes apresentam o mesmo valor para um determinado item de dado, pode-se eliminar um desses campos sem perda de informação. Já para outros campos, esse tratamento pode não ser interessante. No

[1]http://pipes.yahoo.com/

[2]http://www-142.ibm.com/software/products/br/pt/mashuphub/

[3]http://www.apatar.com/

[4]http://www.simile-widgets.org/exhibit/

```
1.  { items : [
2.      {  type: “evento”,
3.         called: “Festival de Rock”,
4.         price: “$ 150”,
5.         artist: “Jimi Hendrix”,
6.         “geo:lat”: “25.430401”,
7.         “geo:long”: “49.280946”,
8.         category: “60’s”,
9.         location: “somewhere”,
10.        date: “20/09/2012”,
11.        desc: “Melhor show de todos os tempos”
12.    } ] }
```

(a) Fonte 1 - data1.js

```
1.  { items : [
2.      {  type: “show”,
3.         name: “Festival de Rock”,
4.         headliner: “Jimi Hendrix”,
5.         price: “$300”,
6.         genre: “rock”,
7.         date: “18/09/2012”,
8.         desc: “Tributo ao verdadeiro rei”,
9.         location: “anywhere”,
10.    } ] }
```

(b) Fonte 2 - data2.js

```
1.  { items : [
2.      {  type: [“evento”, “show”],
3.         label: “Festival de Rock”,
4.         headliner: “Jimi Hendrix”,
5.         price: “$300”,
6.         “geo:lat”: “25.430401”,
7.         “geo:long”: “49.280946”,
8.         genre: [“60’s”, “rock”],
9.         date: “18/09/2012”,
10.        desc: “Melhor show de todos os tempos”,
11.        location: “somewhere”
12.    } ] }
```

(c) Resultado da fusão

Figura 1. Fontes de dados e resultado da Fusão

exemplo da Figura 1 isso ocorre nos campos *category* e *genre*. Neste caso, fazer a união dos valores de ambos e registrar em apenas um campo torna a informação mais relevante. Já em outros casos, como nos campos *price*, *date*, *desc* e *location* a melhor opção pode ser escolher o valor que se deseja manter como resultado da fusão.

Através do Exhibit um usuário comum não conseguiria fazer a integração de dados do exemplo acima. Para isso, seria necessário que ele tivesse conhecimento da linguagem de programação na qual a ferramenta é implementada. Com o Exhibit é possível carregar os dados de ambas as fontes e informar qual campo deve ser tratado como campo identificador. Mas não é possível realizar, de uma forma simples, esse mapeamento para os outros campos e obter uma visão integrada dos dados, como ilustrado na Figura 1(c).

O objetivo deste trabalho é propor um novo importador para a ferramenta Exhibit. O objetivo do importador é permitir que um usuário comum, sem conhecimentos de linguagens de programação, possa integrar dados provenientes de fontes distintas. Isso pode ser feito, bastando que o usuário conheça as estruturas dos dados e faça o mapeamento entre os campos correspondentes, além de definir de que forma possíveis inconsistências entre os dados devem ser resolvidos.

O restante deste artigo está organizado da seguinte forma. A seção 2 apresenta algumas estratégias de fusão de dados propostas na literatura. A seção 3 apresenta alguns

detalhes da ferramenta Exhibit. O módulo importador que estende a ferramenta com estratégias de fusão de dados é apresentado na seção 4. A seção 5 apresenta trabalhos relacionados. Por fim, o trabalho é concluído na seção 6 com algumas considerações finais e trabalhos futuros.

2. Fusão de Dados

Fusão de dados é o processo de combinar múltiplas representações de um mesmo objeto, extraídas de diversas fontes de dados externas, em uma representação única e limpa; ou seja, uma representação sem inconsistências. Ela em geral é a última etapa do processo de integração de dados, realizada após as etapas de casamento de esquemas e identificação de entidades. Em outras palavras, a fusão de dados é realizada após os objetos que se referem a uma mesma entidade no mundo real já term sido identificados e seus atributos correspondentes terem sido mapeados.

A ferramenta Exhibit dá suporte ao processo de identificação de entidades através da definição de um atributo de cada fonte de dados que é considerado como chave primária. No exemplo da figura 1, é possível definir que o atributo *called* é chave para a Fonte 1, bem como o atributo *name* é chave para a Fonte 2. Assim, sempre que as fontes possuirem itens que tenham valores coincidentes para estes atributos, a ferramenta considera que eles se referem à mesma entidade no mundo real. Contudo, o Exhibit não dá suporte ao casamento de esquemas, considerando como atributos correspondentes somente aqueles que possuem o mesmo nome. Além disso, as inconsistências entre os valores de atributos também não são tratadas pela ferramenta. Ou seja, ela não dá suporte à fusão de dados.

A fusão de dados em geral é baseada em estratégias que determinam como possíveis inconsistências entre os dados são resolvidas. Um tutorial das estratégias existentes pode ser encontrada em [Bleiholder and Naumann 2008]. Neste trabalho, são consideradas duas estratégias:

- **concatenação**: esta estratégia concatena todos os valores distintos fornecidos pelas fontes de dados;
- **escolha de fonte:** esta estratégia mantém apenas o valor fornecido pela fonte definida como prioritária.

Um exemplo da estratégia de concatenação é apresentada na figura 1 para obter o valor do atributo *type* no documento apresentado na Figura 1(c), que contém os valores "evento" e "show", fornecidos pelas Fontes 1 e 2. Já para o atributo *price* a Fonte 2 foi escolhida como prioritária e para o atributo *location* a Fonte 1 foi escolhida como prioritária. Como resultado, os valores após a fusão dos dados são "$300" e "somewhere", fornecidos pelas Fontes 2 e 1, respectivamente.

3. A Ferramenta Exhibit

O Exhibit é uma ferramenta livre para a geração de mashups na qual o desenvolvedor da aplicação tem à sua disposição arquivos HTML com exemplos de utilização dos componentes de visualização da ferramenta. Como base nestes exemplos, que podem ser considerados "esqueletos" de arquivos HTML, novas aplicações podem ser criadas através da composição e integração de dados provenientes de diversas fontes. A Figura 2 ilustra a arquitetura do Exhibit do ponto de vista de conteúdo e apresentação. Objetos importadores fornecem dados à ferramenta obtidos a partir de fontes de dados em vários formatos. Este

conteúdo é utilizado por objetos de apresentação que inserem código nos locais marcados no arquivo HTML.

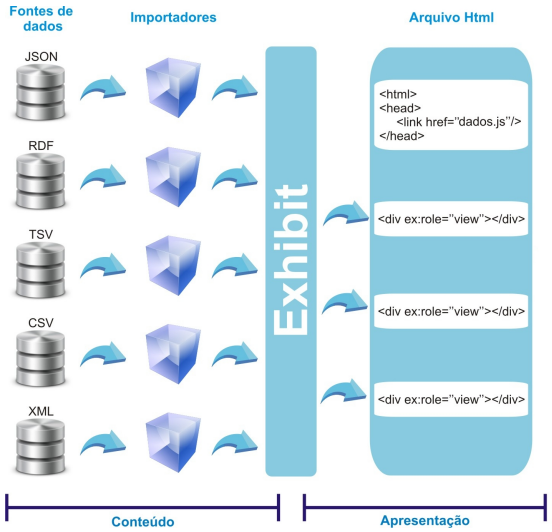

Figura 2. Arquitetura do Exhibit

Uma fonte de dados é um arquivo contém os dados a serem inseridos no website. Os dados podem estar nos formatos JSON, RDF, TSV, CSV e XML. O formato padrão do Exhibit é o JSON, com uma sintaxe um pouco mais relaxada que o padrão JSON[5] proposto. Ele consiste em um array de itens, como ilustrado na Figura 1(a), 1(b) e 1(c). Cada item pode ser pensado como um registro de um banco de dados e consiste basicamente de um conjunto de pares "propriedade: valor".

Para cada um dos formatos de descrição de dados, existe um importador correspondente. Importadores são objetos instanciados pelo Exhibit, responsáveis por carregar os dados e repassá-los ao banco de dados da ferramenta. Com exceção do formato padrão, os importadores também realizam um pré-processamento, no qual analisam o conteúdo em seu formato e traduzem para o formato JSON.

O arquivo HTML é um componente de apresentação. Nele é descrita a fonte de dados dentro da tag *head* e no corpo do HTML são descritos os marcadores. Marcadores são tags HTML, a príncipio tags *div*, que possuem atributos reconhecidos pela ferramenta, como o atributo *ex:role*. É dentro destas tags marcadas, que o Exhibit insere código que produzirá uma vizualização dos dados. Objetos de apresentação são elementos da ferramenta responsáveis pela geração de código. É através do valor do atributo de marcação que o Exhibit escolhe qual objeto de apresentação será responsável em gerar código no trecho marcado.

Para desenvolver uma nova aplicação, é necessário apenas um editor de texto. Um conhecimento básico em HTML é útil, mas não essencial, pois em muitos momentos o usuário precisa apenas copiar e colar trechos e modificá-los de acordo com a necessidade.

A Figura 3 apresenta um arquivo HTML (*fonte1.html*) para apresentar um conjunto de itens de dados como ilustrado na Figura 1(a). No arquivo HTML existe uma chamada para um script externo (Linha 5), que carrega a ferramenta e a inicializa. Existem

[5]http://www.json.org/

```
1.  <html>
2.  <head>
3.    <title>Eventos Musicais</title>
4.    <link href="fonte1.js" type="application/json" rel="exhibit/data"/>
5.    <script src="http://api.simile-widgets.org/exhibit/2.2.0/exhibit-api.js"type="text/javascript">
6.    </script>
7.    <style> </style>
8.  </head>
9.  <body>
10.   <h1>Eventos Musicais</h1>
11.   <table width="100%">
12.     <tr valign="top">
13.       <td ex:role="viewPanel"> <div ex:role="view"></div>
14.       </td>
15.       <td width="25%">
16.         controles de navegação são colocados aqui
17.       </td>
18.     </tr>
19.   </table>
20. </body>
21. </html>
```

Figura 3. Utilização do Exhibit para visualização da Fonte 1.

também dois atributos de elementos que não fazem parte do padrão HTML. Os atributos *ex:role="viewPanel"* e *ex:role="view"* (Linha 13) funcionam como marcadores para o Exhibit. Para ter acesso à nova aplicação, basta abrir o arquivo *fonte1.html* em um navegador. Com apenas estes passos é possível construir um serviço que mostra uma base de dados de uma forma simples. Nota-se aqui a simplicidade para criação de um serviço usando Exhibit, o que vai de encontro com a sua proposta.

O Exhibit, após ser iniciado, instancia um objeto que é responsável por buscar as referências a arquivos JSON descritas no HTML e carregar seus conteúdos através do respectivo importador. As fontes de dados são declaradas através de elementos *link*(Linha 4 da Figura 3) inseridos nos escopo de elementos *head* do HTML. Estes elementos possuem basicamente três atributos. O atributo *rel* informa que o link é do contexto do Exhibit. O nome do arquivo que contém os dados é informado no atributo *href*. Para saber qual importador usar a ferramenta obtém a informação contida no atributo *type*. Como citado anteriormente, o Exhibit já contém importadores para os formatos JSON, RDF, TSV, CSV e XML. Porém, ele não provê suporte para a fusão de dados. Um novo módulo para estender o Exhibit com esta funcionalidade é descrito na próxima seção.

4. Um Módulo de Fusão de Dados para o Exhibit

Quando uma aplicação/serviço web disponibiliza seus dados para uso por terceiros ela define um formato para exportação destes dados. Contudo, na maioria dos serviços que disponibilizam dados, não existe um padrão de nomeação das propriedades ou atributos. As figuras 1(a) e 1(b) mostram exemplos que, embora possuam propriedades diferentes, representam o mesmo evento no mundo real. Estas saídas também apresentam valores diferentes para estes atributos. Um mashup que possua como entrada estes dados deve agrupá-los em uma representação única e consistente. A Figura 1(c) ilustra um resultado

dessa fusão de dados. Para prover esta facilidade pela ferramenta exhibit, nesta seção é apresentado o *Mashup Importer*. Ele é um importador que possibilita que duas fontes de dados no formato JSON sejam carregados, mapeando os atributos correspondentes e a definição de estratégias para a resolução de conflitos. A Figura 4 mostra a arquitetura do importador no escopo da ferramenta Exhibit.

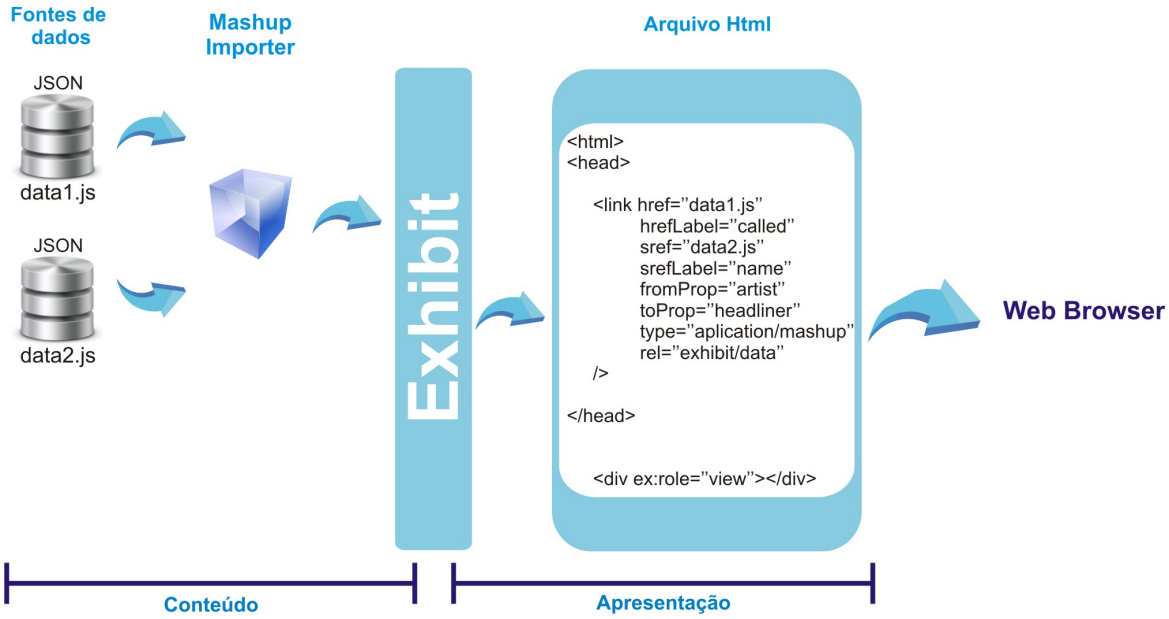

Figura 4. Arquitetura do *Mashup Importer*

Como em qualquer outro importador da ferramenta, para que a fusão de dados seja realizada, no código HTML deve ser criada uma tag *link* dentro do escopo da tag *head*. Entretanto o importador criado necessita de mais informações para poder prover as novas funcionalidades. A figura 5 apresenta um exemplo deste trecho de código.

Para usar o *Mashup Importer* o desenvolvedor deve fornecer algumas informações através de atributos, que são descritos na sequência.

1. *href* - Nome do arquivo da primeira fonte de dados.
2. *hrefLabel* - Nome do campo que atua como identificador na primeira fonte.
3. *sref* - Nome do arquivo da segunda fonte de dados.
4. *srefLabel* - Nome do campo que atua como identificador na segunda fonte.
5. *fromProp* - Nomes de campos da primeira fonte que serão mapeados.
6. *toProp* - Nomes de campos da segunda fonte que serão mapeados.

```
1. <link href="data1.js"hrefLabel="called"
2.       sref="data2.js"srefLabel="name"
3.       fromProp="artist,category"toProp="headliner,genre"
4.       fusionDefault="concat"
5.       fusionAttrib= "(price, sref) (date, sref) (desc, href) (location, href)"
6.       type="application/mashup"rel="exhibit/data"/>
```

Figura 5. Declaração de fusão de dados no Exhibit

7. *fusionDefault* - define a estratégia padrão para realizar a fusão de dados, que pode ser:
    - **concat**: para a estratégia de concatenação de dados;
    - **href**: para escolher prioritariamente o valor fornecido pela fonte relacionada ao atributo *href*;
    - **sref:** para escolher prioritariamente o valor fornecido pela fonte relacionada ao atributo *sref*.
8. *fusionAttrib* - sequência de pares (atributo, estratégia) para declarar estratégias diferentes para atributos específicos. Ou seja, caso haja uma inconsistência em um atributo, ela é primeiramente resolvida com a estratégia declarada especificamente para o atributo. Caso não haja uma estratégia específica, a estratégia default é utilizada.
9. *type* - Indica ao Exhibit qual importador usar.
10. *rel* - Indica ao Exhibit que esta é uma tag link que ele deve considerar.

A ferramenta Exhibit requer que cada item de uma base de dados possua um campo chamado *label*. Este campo é obrigatório pois através dele o Exhibit realiza a fusão de itens. Todo item que possua o mesmo valor para o campo *label* é tratado como um só. Ou seja, é considerado que eles referem-se ao mesmo objeto do mundo real e portanto devem ser apresentados como um único objeto no novo serviço. Para utilizar o *Mashup Importer* o desenvolvedor do mashup deve indicar quais campos serão tratados como label através dos atributos *hrefLabel*, para a primeira fonte, e *srefLabel*, para a segunda.

O mapeamento de campos é realizado pelo desenvolvedor informando os campos a serem mapeados nos atributos *fromProp* e *toProp*, separados por vírgula. O primeiro campo descrito em *fromProp* será mapeado com o primeiro campo descrito em *toProp*, e assim por diante. A Figura 5 mostra o mapeamento do campo *artist* da primeira fonte com o campo *headliner* da segunda fonte.

O atributo *type* deve sempre possuir o valor *application/mashup*. É através desse atributo que a ferramenta sabe qual importador deve usar. O atributo *rel* deve sempre possuir o valor *exhibit/data*, para indicar ao Exhibit que a tag possui dados que ele deve carregar.

Com o uso do *Mashup Importer* o problema relatado na introdução deste artigo é solucionado. O importador realiza sua tarefa através da renomeação de nomes de campos. Os campos *called* (Figura 1(a)) e *name* (Figura 1(b)) recebem o nome *label*. Dessa forma os itens são tratados como um único dado. Para o mapeamento de campos o importador renomeia os campos descritos no atributo *fromProp* utilizando os nomes de campos descritos no atributo *toProp*. Sendo assim o campo *artist* (Figura 1(a)) é renomeado para *headliner*. Quando os campos mapeados possuem o mesmo valor, um dos valores é desprezado para que não haja duplicatas no resultado final. Já para os campos *category* (Figura 1(a)) e *genre* (Figura 1(b)) o resultado final é uma concatenação dos valores de ambos, como o resultado esperado, ilustrado pela Figura 1(c). Similarmente, conflitos nos valores dos atributos *price* e *date* escolhem o valor fornecido pela Fonte 2, enquando os valores mantidos para os atributos *desc* e *location* são provenientes da Fonte 1.

5. Trabalhos Relacionados

Um exemplo de serviço criado apenas com o uso de dados ou serviços providos por terceiros é o StereoMap[Duszczak et al. 2010]. Ele é um mashup que permite a busca de eventos musicais por localidade, fazendo uso dos serviços Last.fm[6], Google Maps[7], Twitter[8] e Wikipedia[9]. Nele o usuário, ao informar uma localidade, tem como retorno os eventos musicais da cidade de uma forma visual com o uso de marcadores em um mapa. O usuário pode acessar informações do artista via Wikipedia e comunicar ou convidar seus amigos através do Twitter. O StereoMap foi criado usando a linguagem de programação Javascript.

O conceito de mashup auxilia a composição de serviços e seu entendimento. Em [Abiteboul et al. 2008] é apresentado um modelo formal para mashups baseado em mashlets, componente básico do modelo. Um mashlet pode importar dados de determinada fonte, importar outro mashlet, usar serviços web externos e impor padrões de interação entre seus componentes. O modelo é hierárquico no sentido que um mashlet pode incorporar outro mashlet, que por sua vez incorpora outros mashlets, e assim por diante, recursivamente. Há sistemas que proveem soluções avançadas para a integração de dados, como Potter's Wheel [Raman and Hellerstein 2001] , que dá suporte ao processo de limpeza de dados e integração de esquemas, mas que não foi proposto para a construção de mashups especificamente. O trabalho que mais se aproxima aos propósitos do Mashup Importer é o sistema Karma [Tuchinda et al. 2011], que tem como objetivo não apenas a construção de mashups através de exemplos, mas também a integração e resolução de conflitos.

6. Conclusão

Este artigo apresenta uma extensão da ferramenta de construção de mashups Exhibit com a funcionalidade de fusão de dados. Esta funcionalidade foi obtida com o desenvolvimento de um novo importador na ferramenta. O importador possibilita a declaração de atributos correspondentes em duas fontes de dados e estratégias para a resolução de valores de atributos, caso elas existam. Web mashups são uma tendência já estabelecida na internet. Entretanto, pode ser concluído através deste estudo, que a tecnologia ainda é um campo de investigação não completamente explorado. Isso vai de encontro com a proposta do Mashup Importer, que facilita a fusão de dados na ferramenta Exhibit, tornado-a mais robusta para geração de web mashups. O novo módulo soluciona o problema introduzido na introdução do artigo. Este trabalho pode ser estendido de diversas formas em trabalhos futuros:

- Inclusão de suporte a outros formatos de entrada como o XML, outro padrão de disponibilização de conteúdo na internet.
- Possibilitar a fusão de dados de mais que duas fontes de dados e utilização de novas políticas para fusão, tais como o dado mais recente ou o valor fornecido pela maioria das fontes.

[6]http://www.last.fm/
[7]http://maps.google.com/
[8]http://twitter.com/
[9]http://www.wikipedia.org/

Referências

Abiteboul, S., Greenshpan, O., and Milo, T. (2008). Modeling the mashup space. In *Proceeding of the 10th ACM workshop on Web information and data management*, WIDM '08, pages 87–94, New York, NY, USA. ACM.

Bianchini, D., De Antonellis, V., and Melchiori, M. (2010). Semantic-driven mashup design. In *Proceedings of the 12th International Conference on Information Integration and Web-based Applications & Services*, iiWAS '10, pages 247–254, New York, NY, USA. ACM.

Bleiholder, J. and Naumann, F. (2008). Data fusion. *ACM Computing Survey*, 41(1):1–41.

Duszczak, J., Zambom, L., Batista, O., Ferreira, L., Ibrahim, I., and Hara, C. (2010). Stereomap: Um mashup para busca de eventos musicais. In *VI Escola Regional de Banco de Dados*, ERBD '2010.

Raman, V. and Hellerstein, J. M. (2001). Potter's wheel: An interactive data cleaning system. In *Proc. of the 27th VLDB Conference*.

Tuchinda, R., Knoblock, C. A., and Szekely, P. (2011). Building mashups by demonstration. *ACM Transactions on the Web*, 5(3).

Zang, N. and Rosson, M. B. (2009). Playing with information: How end users think about and integrate dynamic data. In *IEEE Symposium on Visual Languages and Human-Centric Computing*, pages 85–92.