Universidade Federal de Pernambuco

Graduação em Ciência da Computação

Centro de Informática

**Uma Ferramenta para Coleta de *Feedback* do Usuário em Sistemas de Integração de Dados *Pay-as-You-Go***

**Aluno:** Rafael Barbosa Gonçalves {rbg2@cin.ufpe.br}

**Orientadora:** Bernadette Farias Lóscio {bfl@cin.ufpe.br}

Dezembro, 2011

Universidade Federal de Pernambuco

Centro de Informática

**Rafael Barbosa Gonçalves**

**Uma Ferramenta para Coleta de *Feedback* do Usuário em Sistemas de Integração de Dados *Pay-as-You-Go***

*Trabalho apresentado ao Programa de Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.*

***Orientadora:*** *Profa. Bernadette Farias Lóscio*

Recife, Dezembro de 2011

*“A persistência é o menor caminho do êxito.”*

***(Charlie Chaplin)***

***Aos meus pais, Rufino e Zélia e aos meus irmãos Rodrigo e Renata.***

Agradecimentos

Quando penso em agradecer a alguém por esta conquista, as primeiras pessoas que me vêm em mente são meus pais, Rufino e Zélia. Meu pai sempre muito companheiro, sempre do meu lado quando precisei, de alguma maneira dava um jeito pra eu vir de carro quando era necessário passar a madrugada na faculdade. Minha mãe sempre muito atenciosa e preocupada, sempre ligando quando eu ficava até tarde fazendo projetos. Obrigado por tudo, queridos pais!

Meus irmãos Rodrigo e Renata também estiveram sempre comigo, procurando me acalmar quando me viam apreensivo por conta da faculdade.

Agradeço também a todos os meus familiares que, mesmo que não estivessem comigo, estavam na torcida pra que tudo desse certo.

Gostaria de agradecer também à professora Bernadette Lóscio. Durante o convívio neste semestre vi que era uma pessoa muito boa, além de muito competente. Sempre disponível e disposta pra tirar minhas dúvidas. Foi uma honra ter sido seu primeiro aluno da disciplina de Trabalho de Graduação na UFPE. Obrigado, Berna!

Também gostaria de agradecer a Helio Rodrigues, aluno de mestrado da professora Bernadette e meu co-orientador. Também me ajudou bastante neste semestre. Valeu, Helio!

Pra finalizar, eu não poderia deixar de agradecer a meu grande amigo Flavinho! Que também estava na batalha pra se formar este semestre. Sempre trocando ideias pelo gtalk. Valeu, parceiro!

A todos, meu sincero e profundo muito obrigado!

Resumo

Com a popularização da Web, é crescente a quantidade de informações distribuídas pela rede, as quais podem estar dispostas nas mais diferentes fontes de dados. Da mesma maneira, é crescente a necessidade de ter acesso a esta vasta quantidade de dados. Diante dessa nova realidade, uma parcela da comunidade científica estuda o campo de Integração de Dados, o qual busca desenvolver meios para prover acesso integrado a informações armazenadas em múltiplas bases de dados distribuídas e heterogêneas [1].

Porém, construir sistemas de integração de dados na Web possui um alto custo na etapa de inicialização, quando é preciso identificar as fontes de dados, gerar esquema de mediação e mapear esquemas. Uma alternativa para esse alto custo inicial dos sistemas de integração de dados é utilizar a abordagem de integração de dados *pay-as you-go*, que se caracteriza por inicializar o sistema de integração com o mínimo de mapeamentos estabelecidos e com fontes de dados ainda não muito precisas, mas que serão refinadas de forma incremental à medida que forem sendo coletados o *feedback* dos usuários.

A utilização do *feedback* dos usuários tem se tornado evidente em sistemas que utilizam a abordagem *pay-as-you-go*. Para isso, é interessante que se tenha alguma maneira eficiente de coletar esse *feedback* para que o sistema possa retornar resultados cada vez mais precisos. Neste contexto, o objetivo principal deste trabalho consiste na implementação de uma ferramenta para a coleta do *feedback* do usuário a partir do resultado de consultas de sistemas de integração de dados *pay-as-you-go*. Por meio da ferramenta proposta, o usuário poderá avaliar os resultados das consultas, permitindo, desta forma, identificar se o resultado apresentado para uma dada consulta foi satisfatório ou não. Como fase anterior à implementação da ferramenta foi realizado um estudo sobre o comportamento dos sistemas que utilizam a colaboração de usuários (*crowdsourcing*) para melhorar seus serviços, assim como os pré-requisitos para se criar esses tipos de sistemas.

**Palavras-chave**: *Feedback*, Integração de dados, *pay-as-you-go*, *Linked Data.*

Sumário

[Capítulo 1 - Introdução 1](#_Toc312144798)

[1.1 Motivação 1](#_Toc312144799)

[1.2 Contribuição 2](#_Toc312144800)

[1.3 Estrutura do Documento 2](#_Toc312144801)

[1.4 Considerações 3](#_Toc312144802)

[Capítulo 2 - Contextualização 4](#_Toc312144803)

[2.1 Integração de dados 4](#_Toc312144804)

[2.2 Sistemas de integração de dados *pay-as-you-go* 5](#_Toc312144805)

[2.3 *Feedback* do usuário 5](#_Toc312144806)

[2.4 Considerações 7](#_Toc312144807)

[Capítulo 3 - Sistemas de colaboração em massa 8](#_Toc312144808)

[3.1 Definição de sistemas de colaboração em massa 8](#_Toc312144809)

[3.2 Classificando os sistemas de colaboração em massa 9](#_Toc312144810)

[3.3 Exemplos de sistemas de colaboração em massa na Web 11](#_Toc312144811)

[3.3.1 Sistemas explícitos 13](#_Toc312144812)

[3.3.2 Sistemas implícitos 14](#_Toc312144813)

[3.3.3 Desafios 15](#_Toc312144814)

[3.4 Considerações 21](#_Toc312144815)

[Capítulo 4 - RDF Fan - Ferramenta de coleta de feedback para triplas RDF 22](#_Toc312144816)

[4.1 Classificação da ferramenta 22](#_Toc312144817)

[4.2 Visão geral 24](#_Toc312144818)

[4.3 Especificação e Implementação 26](#_Toc312144819)

[4.4 Considerações 31](#_Toc312144820)

[Capítulo 5 - Conclusão 32](#_Toc312144821)

[Referências Bibliográficas 33](#_Toc312144822)

# Introdução

Neste capítulo será feito uma breve introdução do trabalho, mostrando qual foi a motivação para o seu desenvolvimento, quais as contribuições do trabalho e como o documento está estruturado.

## Motivação

A Web é um dos meios mais utilizados para a disseminação de informações e abrange diversas aplicações. O grande volume de informações disponível na Web tem motivado o crescimento de aplicações que, frequentemente, necessitam integrar dados, ou seja, devem permitir o acesso, por meio de uma visão unificada, chamada esquema de mediação, a um conjunto de fontes de dados autônomas e heterogêneas. As soluções de integração de dados buscam oferecer aos usuários uma interface uniforme de acesso à diferentes fontes de dados, de forma que os usuários definam consultas especificando “o que” se deseja saber e o sistema determine “onde” a informação pode ser encontrada e, em seguida, apresente as respostas para as consultas do usuário [16].

Duas décadas de extensa pesquisa com experiências no mundo real mostraram que a construção de sistemas de integração de dados não é uma tarefa fácil [2]. De maneira geral, a dificuldade está em compreender as necessidades dos usuários, assim como o quão relevantes as fontes de dados são, a forma como o conteúdo das fontes deve ser combinado e estruturado de forma que seja compatível com os requisitos dos usuários [3].

Outra dificuldade em construir sistemas de integração de dados na Web está no grande custo na fase de inicialização, quando é preciso identificar quais serão as fontes de dados, gerar os esquemas de mediação e os mapeamentos entre esses esquemas e os esquemas das fontes a serem integradas. Uma alternativa para esse alto custo de inicialização é utilizar a abordagem *pay-as-you-go*, que faz uso de uma abordagem de integração de dados sob demanda, onde o sistema de integração vai sendo refinado de forma automática e incremental. Á medida que o sistema vai sendo utilizado, os mapeamentos entre os esquemas vão sendo refinados, proporcionando ao usuário resultados cada vez melhores.

Como forma de contribuir nesse processo de refinamento, alguns pesquisadores vêm explorando a ideia de utilizar o *feedback* dos usuários para auxiliar o processo de integração de informações [3]. A partir do *feedback* do usuário é possível identificar o que o usuário de fato espera receber como resposta às consultas submetidas ao sistema, por exemplo. De maneira geral, temos que o *feedback* do usuário tem o objetivo de orientar a construção de sistemas de integração de dados e/ou no aperfeiçoamento da qualidade dos serviços que prestam [4, 5, 6, 7, 8, 9, 10, 11].

## Contribuição

Este estudo foi proposto por conta da necessidade de uma ferramenta de coleta do *feedback* do usuário sobre resultados de consultas na Web da dados [12]. Essa ferramenta faz parte de uma solução que está sendo implementada em uma dissertação de mestrado [13], a qual tem como objetivo auxiliar a análise de relevância de fontes de dados distribuídas na Web da dados. Especificamente, as anotações de *feedback* geradas pela ferramenta serão utilizadas para avaliar o benefício que a inserção de uma nova fonte de dados pode trazer ou não para um sistema de integração de dados *pay-as-you-go*, contribuindo para que melhores resultados sejam retornados para o usuário.

Utilizando a ferramenta proposta, denominada de RFD Fan, o usuário poderá avaliar os resultados de consultas SPARQL, permitindo, desta forma, identificar se o conjunto de triplas retornado como resultado para uma dada consulta é satisfatório ou não. A partir desse *feedback* será possível, de forma incremental, avaliar a qualidade das fontes que fornecem os dados para as consultas e, desta maneira melhorar o resultado de tais consultas.

Como fase anterior à implementação da ferramenta foi realizado um estudo sobre sistemas de colaboração em massa, que são sistemas que utilizam o *feedback* do usuário como forma de melhorar os serviços prestados. Esse estudo contou com a classificação dos sistemas de colaboração em massa, para que pudéssemos melhor classificar e caracterizar a ferramenta proposta.

## Estrutura do Documento

O trabalho está estruturado da seguinte forma: No Capítulo 2, apresentamos uma breve contextualização a respeito de integração de dados, sistemas de integração de dados *pay-as-you-go* e *feedback* do usuário. No Capítulo 3, definimos sistemas de colaboração em massa, destacando alguns dos sistemas de colaboração em massa que são utilizados atualmente e apresentando os desafios encontrados na implementação de tais sistemas. No Capítulo 4 apresentamos a ferramenta implementada neste projeto. Primeiramente classificamos a ferramenta de acordo com as dimensões vistas no Capítulo 3. Em seguida, mostramos uma visão geral do sistema que a ferramenta RDF Fan está inserida, analisando o papel de cada um dos módulos deste sistema. Para finalizar o capítulo, mostramos com será o funcionamento da ferramenta, detalhando sua especificação e implementação. No capítulo 5, apresentamos uma breve conclusão, onde descrevemos os conhecimentos adquiridos no desenvolvimento do trabalho. Também mostramos algumas sugestões de trabalhos futuros que podem ser realizados com relação à ferramenta implementada.

## Considerações

Neste capítulo, mostramos a importância do *feedback* do usuário em sistemas de integração de dados *pay-as-you-go* e a necessidade de uma ferramenta que realize a coleta de *feedback* para este tipo de sistema. Como contribuição, foi construída uma ferramenta que irá gerenciar esse processo de coleta de *feedback*. Por último, falamos a respeito da estrutura do documento.

No Capítulo 2, veremos uma breve fundamentação teórica para que fiquem claras as definições que serão usadas nos capítulos seguintes.

# Contextualização

Neste capítulo serão abordados alguns conceitos básicos e de fundamental importância para o entendimento do propósito deste trabalho. Primeiramente, falaremos a respeito de integração de dados e suas características principais. Em seguida, veremos como funcionam os sistemas de integração de dados *pay-as-you-go*, caracterizados por adotar uma abordagem incremental de integração e que fazem uso do *feedback* do usuário para o refinamento de seus dados. Por último, iremos entender melhor a definição e a importância do *feedback* dos usuários.

## Integração de dados

Integração de dados é uma importante área de pesquisa que evoluiu bastante a partir do surgimento da Web e da crescente facilidade de acesso a esse grande volume de dados. De uma maneira geral, o objetivo da integração de dados é permitir o acesso, por meio de uma visão unificada, a um conjunto de fontes de dados autônomas e heterogêneas [14].

Um sistema de integração de dados é composto essencialmente por quatro elementos: esquemas das fontes, os conjuntos de dados a serem integrados, um esquema de integração sobre os quais os usuários submetem consultas e um conjunto de mapeamentos que especifica como os dados estruturados sob os esquemas das fontes podem ser transformados e combinados em dados estruturados de acordo com o esquema de integração [5].

Uma das grandes dificuldades enfrentadas no desenvolvimento e manutenção de um sistema de integração de dados é que as fontes de dados podem ser autônomas e dinâmicas, obrigando o sistema de integração a ser extensível, ou seja, capaz de se adaptar de acordo com a evolução das fontes de dados a serem integradas [15].

Os sistemas de integração de dados também precisam lidar com a heterogeneidade das fontes de dados, as quais, na maioria das vezes, foram projetadas independentemente utilizando diferentes modelos de dados e diferentes representações para os mesmos conceitos do mundo real [15].

Como passo de inicialização no processo de integração, temos a identificação das fontes de dados, a geração do esquema de mediação e dos mapeamentos entre esquemas [13]. Esses sistemas têm uma visão “primeiro o esquema, depois os dados”, ou seja, investem, inicialmente, uma grande quantidade de recursos para que o esquema de mediação e os mapeamentos sejam bem definidos, e somente depois oferecem o acesso integrado aos dados distribuídos. Dessa forma, o resultado das consultas sobre os dados integrados são mais precisos, exigindo, porém um alto custo de inicialização [13]. Esta inicialização tem sido apontada como um dos principais gargalos dos tradicionais sistemas de integração de dados [15], principalmente, por se tratar de um processo manual ou semiautomático.

## Sistemas de integração de dados *pay-as-you-go*

Em sistemas de integração de dados *pay-as-you-go*, a integração é um processo contínuo. O processo inicia-se com o mínimo de mapeamentos entre as fontes de dados, e, de forma incremental, os mapeamentos vão melhorando na medida em que vai se adicionando semântica entre as fontes. Esta abordagem é diferente da abordagem tradicional de sistemas de integração de dados, onde é necessário ter um conhecimento inicial mais completo a respeito dos mapeamentos entre as fontes [17].

Os sistemas de integração de dados *pay-as-you-go* surgiram, portanto, como uma solução para o alto do custo de inicialização dos sistemas de integração de dados convencionais [18]. Porém, uma consequência desta abordagem é que os resultados recuperados pelas consultas principalmente na fase de inicialização do sistema podem ser menos precisos. Como forma de incrementar a precisão do retorno dessas consultas, pode ser utilizado o *feedback* do usuário [13].

Em particular, os sistemas de integração de dados *pay-as-you-go* utilizam o *feedback* que o usuário fornece sobre os resultados das consultas realizadas no sistema [8]. As anotações obtidas são, em geral, utilizadas para permitir o refinamento incremental dos mapeamentos entre esquemas. O processo de refinamento pode ser realizado interativamente até que os mapeamentos sejam capazes de atingir os requisitos do usuário [13].

## *Feedback* do usuário

Em sistemas de integração de dados, o *feedback* pode ser definido como sendo um conjunto de anotações que o usuário provê sobre artefatos do sistema, sejam consultas, mapeamentos, esquema de mediação, entre outros, com o objetivo de melhorar a qualidade dos serviços disponibilizados. Estas anotações exprimem de forma mais abstrata quais são os requisitos do usuário sobre o artefato anotado. Uma anotação de *feedback* pode ser definida pela tupla [3]:

<obj; t; u; k>,

onde o usuário *u* faz uma anotação sobre o artefato *obj* usando o termo *t*. Considerando que o usuário pode fornecer diferentes tipos de *feedback*, *k* indica o tipo de *feedback* que está sendo fornecido. O conjunto de termos que pode ser usado como valor de *k* depende do tipo de *feedback* que o usuário deseja fornecer.

O *feedback* pode ser utilizado, por exemplo, para anotar mapeamentos de esquemas com estimativas que especifiquem o grau em que eles atendam às expectativas do usuário. Pala ilustrar considere uma relação *r* no esquema de mediação, um mapeamento *m* usado para gerar as instâncias de *r* a partir dos dados das fontes, e uma tupla *t* de *r* que é recuperada usando *m*. O seguinte exemplo de *feedback*, <<r, m, t>, tp, Rafael, tuple\_membership>, especifica que *t* é um *true positive*, isto é, que *t* é membro de *r* de acordo com as expectativas do usuário Rafael. A partir desse *feedback*, mapeamentos candidatos para preencher os elementos de um esquema de integração de dados podem ser anotados, selecionados e refinados [3].

O modelo de *feedback* acima é geral e pode ser usado para descrever os diferentes tipos de *feedback* dos usuários definidos na literatura de integração de informações. A Tabela 1 apresenta propostas de destaque que utilizam a colaboração dos usuários e mostra como o *feedback* utilizado em cada proposta pode ser descrito usando o modelo apresentado acima [3].

No próximo capítulo faremos uma análise de alguns sistemas que utilizam o *feedback* do usuário na sua essência. Esses sistemas são mais conhecidos como sistemas de colaboração em massa.

|  |  |  |
| --- | --- | --- |
| **Proposta** | **Objetos em que o *feedback* é dado** | **Conjunto de termos usados para anotar os objetos** |
| [4] | Uma instância de um determinado esquema e a instância obtida pela sua transformação em outro esquema | {‘yes’, ‘no’}  // usados ​​para comentar sobre a transformação do esquema |
| [5] | Uma tupla resultante | {‘true positive’, ‘false positive’, ‘false negative’} |
| Um atributo e seu valor | {‘true positive’, ‘false positive’, ‘false negative’} |
| [6] | Uma consulta candidata | {‘true positive’, ‘false positive’} |
| Um par de consultas candidatas | {‘bafore’, ‘after’} // usado na ordenação de queries |
| [7] | Uma visualização de resultados de tuplas | {‘insert’, ‘delete’, ‘update’} |
| [8] | Um mapeamento | {‘true positive’, ‘false positive’} |
| [9] | Um atributo da relação | Conjunto de atribitos de tipos de dados |
| Dois atributos de uma relação dada | Conjunto de restrições |
| Um match | {‘true positive’, ‘false positive’} |
| [11] | Uma tupla resultado | {‘true positive’, ‘false positive’} |
| Um par de tuplas resultado | {‘bafore’, ‘after’} // usado na ordenação de queries |

Tabela 1: Tipos de *feedback* considerados na literatura de integração de informação

## Considerações

Neste capítulo, vimos a fundamentação teórica do nosso trabalho, onde falamos a respeito de Integração de dados, mostrando as principais diferenças entre a abordagem clássica e a *pay-as-you-go*. Destacamos o motivo que torna a abordagem *pay-as-you-go* mais viável para alguns tipos de sistemas que necessitam de um processo de inicialização mais rápido. Definimos também o *Feedback* do usuário e como ele será utilizado no nosso trabalho.

No Capítulo 3, falaremos a respeito dos sistemas de colaboração em massa, no qual o RDF Fan está incluído. Iremos mostrar porque é importante fazer um estudo a respeito deste tipo de sistemas antes de implementá-los.

# Sistemas de colaboração em massa

Os sistemas de colaboração em massa funcionam por meio da mobilização de uma multidão de humanos para ajudar a resolver uma grande variedade de problemas. Durante a última década, vários desses sistemas foram propostos. Os principais exemplos incluem Wikipedia[[1]](#footnote-1), Linux[[2]](#footnote-2), Yahoo! Respostas[[3]](#footnote-3), sistemas mecânicos baseados em opiniões.

Este tipo de sistema tem recebido vários nomes, tais como *crowdsourcing* [19], sistemas orientados ao usuário, conteúdo gerado pelo usuário, sistemas colaborativos, sistemas comunitários, sistemas sociais, mídia social, inteligência coletiva, sabedoria da multidão, *smart mobs*, colaboração em massa e computação humana. O tópico tem sido amplamente discutido em livros, imprensa popular e academia [20].

Este capítulo busca fornecer uma visão global de sistemas de colaboração em massa na Web. Serão apresentados os principais conceitos relacionados a estes sistemas, bem como alguns exemplos reais. Também serão discutidos os principais desafios como, por exemplo, a forma de recrutar e avaliar usuários e capturar suas contribuições.

## Definição de sistemas de colaboração em massa

Definir os sistemas de colaboração em massa ainda é uma tarefa difícil. A princípio, podemos dizer que um sistema de *cowdsourcing* é um sistema que necessita da colaboração de uma multidão de usuários de forma explícita para construir um artefato de longa duração que trará benefícios para todos os usuários do sistema.

Esta definição, no entanto, parece bastante restrita, uma vez que exclui, por exemplo, o jogo ESP [20], onde os usuários colaboram implicitamente rotulando as imagens para poderem continuar jogando. O ESP claramente se beneficia de uma multidão de usuários. Ele enfrenta o mesmo desafio da importância humana da Wikipedia e do Linux, que é a forma de recrutar e avaliar usuários e como combinar suas contribuições. Diante disso, parece insatisfatório considerar apenas as colaborações explícitas; devemos permitir também as implícitas.

A definição também exclui, por exemplo, o sistema *Amazon Mechanical Turk*[[4]](#footnote-4), onde tarefas que precisam ser feitas por pessoas estão disponíveis para serem feitas por quem tiver tempo disponível. Os usuários pedem, por exemplo, para encontrar um barco desaparecido a partir de milhares de imagens de satélite [20]. Neste caso, os usuários não constroem qualquer artefato, sem dúvida nada é duradouro, e não existe qualquer comunidade (apenas usuários se unindo para esta tarefa particular). Então, assim como o ESP, este sistema claramente se beneficia dos usuários. Diante disso, ele deve ser considerado um sistema de colaboração em massa.

Seguindo a definição apresentada em [20], nós consideramos *crowdsourcing* como um método de resolver problemas num propósito geral. Dizemos que um sistema pode ser classificado como sendo de colaboração em massa se recruta uma multidão de seres humanos para ajudar a resolver um problema definido pelos proprietários do sistema, e ao fazê-lo, aborda quatro desafios fundamentais: i) como recrutar e reter os usuários? ii) que contribuições os usuários podem proporcionar? iii) como combinar as contribuições do usuário para resolver o problema-alvo? iv) como avaliar os usuários e suas contribuições?

## Classificando os sistemas de colaboração em massa

Sistemas de crowdsourcing podem ser classificados de acordo com várias dimensões. Neste trabalho, discutimos as nove dimensões consideradas mais importantes de acordo com [20]. As duas primeiras são referentes à natureza da colaboração, que pode ser implícita ou explícita. Outras três dimensões são referentes ao tipo de problema-alvo. São elas: a construção de artefatos temporários, a construção de artefatos permanentes e a execução de tarefas. A sexta dimensão refere-se a como um sistema de colaboração em massa resolve os quatro desafios fundamentais (como recrutar e reter os usuários, o que os usuários podem fazer, como combinar suas entradas, e como avaliá-las). As três últimas dimensões são relativas ao grau de esforço manual dos usuários, o papel de usuários humanos e arquiteturas *stand-alone* contra arquitetura *piggyback* [20]. A seguir, estas dimensões são descritas:

**Sistemas explícitos**. Sistemas onde os usuários colaboram de forma explícita.

**Sistemas implícitos**. Sistemas que capturam o *feedback* dos usuários de forma implícita, por meio de ações realizadas pelos usuários ao manusear o sistema.

**Construção de artefatos temporários**. Sistemas que têm como objetivo a construção de artefatos que serão usados para resolver um problema específico e logo serão descartados.

**Construção de artefatos permanentes**. Sistemas que têm como objetivo a construção de artefatos permanentes, que serão aprimorados a cada nova colaboração dos usuários. Podemos usar como exemplo o sistema Linux, um sistema operacional *open source*.

**Execução de tarefas**. Sistemas explícitos que contam com a colaboração do usuário simplesmente para executar uma determinada tarefa, como por exemplo, na busca de pessoas desaparecidas.

Os sistemas explícitos, implícitos, de construção de artefatos e de execução de tarefas serão vistos com mais detalhes na próxima sessão.

**Como um sistema de colaboração em massa resolve os quatro desafios fundamentais?** Estas quatro questões são pontos críticos para se construir sistemas de colaboração em massa. Por essa razão, analisaremos essa questão de forma mais detalhada na sessão 3.4.

**Grau de esforço manual.** Ao construir um sistema de colaboração em massa, devemos decidir o quanto de esforço manual será exigido para resolver cada um dos quatro desafios citados. É preciso decidir como dividir o esforço manual entre os usuários e os proprietários do sistema. Alguns sistemas, por exemplo, pedem para os usuários colaborarem de maneira simples e deixam mais trabalho para os proprietários. Por exemplo, para detectar usuários mal-intencionados, os usuários podem simplesmente clicar em um botão para relatar comportamentos suspeitos, enquanto que os proprietários devem examinar cuidadosamente todas as provas pertinentes para determinar se um usuário está realmente mal-intencionado. Alguns sistemas podem funcionar da maneira inversa. Por exemplo, a maior parte da carga manual do Wikipedia recai sobre os usuários (que são responsáveis por atualizar a editar as informações), e não os proprietários [20].

**Papel dos usuários humanos.** Podemos considerar quatro tipos básicos de colaboração de humanos em um sistema de colaboração em massa. i) Escravos: os humanos ajudam a resolver o problema de maneira dividir-para-conquistar, para subdividir os recursos (tempo e esforço, por exemplo) que seriam destinados aos proprietários do sistema; ii) Provedores de perspectiva: os humanos contribuem com perspectivas diferentes, que quando combinadas produzem frequentemente uma solução melhor do que com um único humano (única perspectiva); iii) Provedores de conteúdo: os humanos contribuem como geradores de conteúdo (vídeos no YouTube e imagens no Flickr, por exemplo) e iv) Fornecedores de componentes: os humanos podem ser os componentes de um artefato alvo, tal como uma rede social, ou simplesmente uma comunidade de usuários [20].

Os humanos muitas vezes desempenham múltiplos papéis dentro de um único sistema colaboração em massa (por exemplo os escravos, provedores de perspectiva ou provedores de conteúdo no Wikipedia). É importante conhecer esses papéis, porque isso pode determinar a forma de recrutar usuários. Para usar seres humanos como provedores de perspectiva, por exemplo, é importante recrutar um público diverso, onde cada ser indivíduo pode tomar decisões independentes com o objetivo de evitar "pensamento de grupo" [20].

***Stand-alone* vs *piggyback***. Ao construir um sistema de *crowdsourcing*, podemos optar por criar um sistema *piggyback*, que terá a característica de explorar os traços que os usuários deixam no sistema para resolver um determinado problema-alvo. Por exemplo, o “Você quis dizer...” do Google e o Search Assist do Yahoo utilizam os logs de busca e os clicks do usuário para um motor de busca corrigir erros de ortografia. Outro sistema pode explorar compras do usuário em uma livraria on-line (ex: Amazon) para recomendar livros. Por outro lado, os sistemas *standalone* não precisam resolver os desafios de recrutamento de usuários para decidir o que eles podem fazer (os usuários se encarregam disso). Mas eles têm que decidir como avaliar os usuários e seus *inputs*, além de combinar essas entradas para resolver o problema-alvo [20].

## Exemplos de sistemas de colaboração em massa na Web

Nesta seção, descreveremos alguns exemplos de sistemas de colaboração em massa.

A Tabela 2 mostra um conjunto de tipos de sistemas básicos de *crowdsourcing* e coloca como exemplos aqueles que atualmente têm recebido mais atenção. Da esquerda para a direita, a tabela é organizada pela natureza da colaboração, arquitetura, necessidade de recrutar os usuários e ações que os usuários podem tomar. Vamos agora discutir a respeito desse conjunto de sistemas, que foram divididos em dois grandes grupos: sistemas explícitos e implícitos.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Natureza da colaboração** | **Arquitetura** | **Deve recrutar usuários?** | **O que os usuários fazem?** | **Exemplos** | **Problemas alvo** | **Comentários** |
| Explicitas | Standalone | Sim | Avaliação   * Revisão, voto, rotulação | * Revisar e votar na Amazon | Avaliação de uma coleção de itens (por exemplo, produtos, usuários) | Seres humanos como os prestadores de perspectiva. Ausência ou combinação flexível das contribuições. |
| Compartilhamento   * Itens * Conhecimento textual * Conhecimento estruturado | * Napster, YouTube, FlickrMailing lists, Yahoo! Answers, QUIQ, ehow.com * Swivel, Many Eyes, Google Fusion Tables, Google Base, bmrb.wisc.edu | Construição de uma coleção de itens (distribuídos ou centrais) que podem ser compartilhados entre os usuários. | Seres humanos como os provedores de conteúdo. Ausência ou combinação frouxa das contribuições. |
| Networking | * Linkedin, MySpace, Facebook | Construção de redes sociais | Seres humanos como os provedores de componentes. Ausência ou combinação flexível das contribuições. |
| Construindo artefatos   * Software * Bases de conhecimento textual * Bases de conhecimento estruturadas * Sistemas * Outros | * Linux, Apache, Hadoop * Wikipedia, openmind, Intellipedia, ecolicommunity * Wikipedia infoboxes/DBpedia, IWP, Google Fusion Tables * Wikia Search, mahalo, Freebase, Eurekster * Newspaper at Digg.com, Second Life | Construção de artefatos físicos | Os seres humanos podem desempenhar todas as funções. Normalmente uma combinação firme das contribuições. Alguns sistemas pedem para seres humanos e máquinas para contribuir. |
| Execução de tarefas | * Encontrar extraterrestres, eleições, encontrar pessoas, criação de conteúdo (por exemplo a Demand Media, Associated Content) | Possivelmente qualquer problema |  |
| Implícitas | Standalone | Sim | * Jogar jogos com um objetivo * Apostar em mercados de previção * Uso de contas privadas * Resolver captchas * Comprar/vende/leiloar, pogar jogos com multiplayers maciços | * ESP * intrade.com, Iowa Electronic Markets * IMDB private accounts * recaptcha.net * eBay, World of Warcraft | * Rotulação * Previsão de eventos * Classificando filmes * Digitalizando texto escrito * Construção de uma comunidade de usuários (para fins de cobrança de taxas, publicidade) | Os seres humanos podem desempenhar todas as funções. Combinação flexível ou firme das contribuições. |
| Piggyback em outro sistema | Não | * Busca por palavra-chave * Comprar produtos * Procurar websites | * Google, Microsoft, Yahoo * O recurso de recomendação da Amazon * Web sites adaptativos (por exemplo, a primeira página do Yahoo!) | * Correção ortográfica, previsão de epidemias * Recomendação de produtos * Reorganizar um site para um melhor acesso | Os seres humanos podem desempenhar todas as funções. Combinação flexível ou firme das contribuições. |

Tabela 2: Uma amostra dos tipos básicos de sistemas de colaboração em massa na Web

### Sistemas explícitos

Sistemas explícitos são sempre do tipo *standalone* e coletam contribuições dos usuários de forma explícita. Os usuários podem contribuir de cinco maneiras: através de uma avaliação, compartilhando algo, trabalhando em rede de forma colaborativa, construindo artefatos ou executando determinadas tarefas. Discutiremos esses tipos de colaboração a seguir:

**Avaliação**. Os usuários podem fornecer *feedback* avaliando "itens" (por exemplo livros, filmes, páginas da Web, resultados de consultas, outros usuários), usando comentários textual, pontuação numérica ou rótulos [20].

**Compartilhamento**. Certos sistemas permitem que os usuários compartilhem "itens", tais como produtos, serviços, conhecimento textual e conhecimento estruturado. Alguns deles compartilham serviços e produtos, tais como o Napster, YouTube, CPAN [20]. Outros compartilham conhecimento textual, como em listas de discussão, Twitter ou sistemas de suporte online para clientes (como o QUIQ, [21]).

Sistemas que compartilham conhecimentos estruturados (por exemplo XML, dados RDF) incluem Swivel, Many Eyes, Google Fusion Tables, Google Base, diversos Web sites de eScience (como galaxyzoo.org). No geral, os sistemas de compartilhamento podem ser centrais (como o YouTube, ehow, Google Fusion Tables, Swivel) ou distribuídos, de forma peer-to-peer (como o Napster, Orquestra) [20].

***Networking***. Estes sistemas permitem que os usuários construam de forma colaborativa esquemas de rede social. Em seguida, eles exploram as redes para a prestação de serviços (por exemplo, sugestão de amigos, anúncios, e assim por diante). Em menor grau, sistemas de blogs também são sistemas em rede, pois em muitos casos blogueiros apontam para outros blogueiros [20].

**Construção de artefatos**. Os sistemas que permitem aos usuários construir artefatos (como o Wikipedia) muitas vezes fazem uma forte junção das entradas dos usuários e permitem que os usuários editem e mesclem seus materiais com materiais de outros usuários. Um exemplo de artefato popular são aqueles que geram bases de conhecimentos. Para construir bases de conhecimentos (ex: Wikipedia), os usuários contribuem com dados, tais como frases, parágrafos, páginas Web e, em seguida, editam e mesclam suas contribuições. A “*knowledge capture*” (k-cap.org) e comunidades de Inteligência Artificial têm estudado a construção de bases de conhecimentos já há algum tempo. O sucesso recente da Wikipedia tem inspirado muitas "comunidades wikipedias", tais como a Intellipedia (sistema que inclui informações nas áreas, pessoas e assuntos de interesse daquela comunidade de inteligência dos EUA) [20].

**Execução de tarefas**. O último tipo de sistema de *feedback* explícito que nós consideramos é o tipo que o usuário executa ação (tarefa) como meio de fornecer o *feedback*. Este sistema é usado, por exemplo, na busca de pessoas desaparecidas [20]. Um sistema de coleta de *feedback* explícito foi usado também nas eleições norte americanas de 2008, onde a equipe de Obama fez uma grande operação de colaboração em massa on-line pedindo uma grande quantidade de voluntários para ajudar a mobilizar os eleitores. Para aplicar colaboração em massa a uma tarefa, temos de encontrar partes da tarefa que podem ser "crowdsourced" (colocadas em sistemas de colaboração em massa) de forma que cada usuário possa fazer uma contribuição e as contribuições, por sua vez possam ser combinados para resolver as partes. O Amazon Mechanical Turk pode ajudar a distribuir pedaços de uma tarefa para uma multidão de usuários [20].

### Sistemas implícitos

Como discutido anteriormente, esses sistemas permitem que os usuários colaborem de forma implícita para resolver um problema. Estes sistemas podem ser divididos em dois grupos: autônomos (*standalone*) e *piggyback* [20].

Um sistema autônomo oferece um serviço de tal forma que é usada a colaboração dos usuários implicitamente (como um efeito colateral) para resolver um problema. Muitos desses sistemas estão na Tabela 2, que lista exemplos representativos de alguns. O jogo ESP [22], como dito antes, permite que os usuários joguem um jogo de adivinhar palavras comuns que descrevem as imagens (mostrada de forma independente para cada usuário) e, em seguida, usa essas palavras para rotular as imagens (Google Image Labeler foi baseado nesse jogo). Existem ainda muitos outros jogos com esse propósito [23]: *Prediction markets* [20] permite que os usuários apostem em eventos (como as eleições, eventos esportivos). A intuição é que a "sabedoria coletiva" é muitas vezes precisa (sob certas condições) [24] e que isso ajuda a incorporar informações privilegiadas fornecidas por usuários. reCAPTCHA pede aos usuários para resolver captchas, que são meios automatizados de gerar novos desafios que os computadores atuais são incapazes de resolver, mas a maioria de seres humanos podem resolver, para provar que eles são seres humanos por meio da digitalização de textos escritos [20].

O problema-alvo de muitos desses sistemas (que fornecem serviços para o usuário) é simplesmente produzir uma grande comunidade de usuários, para poder, por exemplo, cobrar taxas de assinatura ou vender anúncios, por exemplo, sites de leilões (compra/venda), como o eBay, e jogos multiplayer, como World of Warcraft, por exemplo, se encaixam nessa descrição [20].

O segundo tipo de sistema implícito que nós consideramos é o *piggyback*, que explora os traços dos usuários (portanto, tornando os usuários deste último sistema colaboradores implícitos) para resolver um problema, isso tudo com o auxílio de motores de busca (como logs de busca ou clicks do usuário). Os traços coletados dos usuários podem ser utilizados em uma ampla gama de tarefas, tais como correção ortográfica, busca por sinônimos ou previsão de epidemias, por exemplo. Outros exemplos incluem a exploração de compras de usuários para recomendar produtos e a exploração de logs de clicks para melhorar a apresentação de um Web site [20].

Os sistemas colaboração em massa muitas vezes combinam várias dessas características básicas. O Wikipedia, por exemplo, constrói uma base de conhecimento textual. Mas também constrói uma base de conhecimento estruturada (via infoboxes) e hospeda fóruns de compartilhamento de conhecimentos (grupos de discussão). O YouTube permite que usuários compartilhem e avaliem vídeos. Portais de comunidades muitas vezes combinam todas as características *crowdsourcing* discutidas até agora [20].

### Desafios

Nesta seção discutiremos os principais desafios dos sistemas de colaboração em massa. Para isso precisamos responder a quatro questões: “Como recrutar e reter usuários?”, “Que contribuições esses usuários podem nos proporcionar?”, “Como combinar essas contribuições?” e “Como avaliar os usuários e as contribuições?” [20].

**Como recrutar e reter os usuários?** Recrutamento de usuários é um dos desafios mais importantes de sistemas de colaboração em massa [20]. A seguir, destacamos algumas soluções:

* Exigir que os usuários contribuam: se tivermos autoridade para fazê-lo (por exemplo, um gerente pode recrutar 100 funcionários para ajudar a construir um sistema em toda a empresa) [20].
* *Feedback* forçado: Toda vez que o usuário fizer uma consulta, faremos ele "pagar pela informação". O “pagamento” é uma caixa de diálogo com uma simples pergunta que os usuários respondem clicando em um botão "sim", "não", ou "não sei", por exemplo. Para utilizar os serviços do sistema o usuário deve, portanto, "pagar" por ele, e o "pagamento" aqui é um pouco de conhecimento para ajudar a construir e/ou manter o sistema. O *feedback* forçado em troca de vantagens não deve ser usado com frequência para não desencorajar o usuário a usá-lo. Devemos também nos certificar de que o sistema é tão atraente para o uso que o usuário esteja disposto a "pagar". O ideal é começar com um sistema que as pessoas já queiram realmente usar, para, gradualmente, construir um mecanismo de *feedback* para expandir as capacidades do sistema [20].
* *Feedback* voluntário: Esta solução é gratuita e fácil de executar e, portanto, é mais popular. A maioria dos sistemas atuais de colaboração em massa na Web, como Wikipedia e YouTube, usam esta solução. A desvantagem do voluntariado é que é difícil prever quantos usuários uma aplicação pode recrutar [20].
* *Feedback* voluntário com gratificação instantânea: O sistema produz os resultados da consulta mas também indica ao usuário que ainda existem mais detalhes sobre os resultados para ser fornecidos, se o usuário está interessado. Para chegar a esses detalhes o sistema necessita que o usuário forneça algum *feedback*. Assim, o usuário tem um forte incentivo para fornecer um *feedback* simples. Por exemplo, suponha que um sistema de integração de dados em um domínio de livros tem uma interface de consulta que permite aos usuários consultar sobre o título e preço do livro. O sistema executa uma consulta e exibe a lista de livros desejados em uma tabela com várias colunas. As duas primeiras colunas são título e preço, que o sistema sabe como preencher (porque já tem os mapeamentos semânticos do título e preço no esquema mediado para aqueles esquemas fonte). A terceira coluna é referente a editores, mas o sistema não sabe como preencher porque não tem nenhum mapeamento semântico para publicar ainda. Assim, a aplicação preenche essa coluna com pontos de interrogação. Suponha agora que o usuário deseja encontrar a editora de um determinado livro na tabela de resultados. O sistema irá fazer ao usuário algumas perguntas, a fim de descobrir qual atributo da fonte (que contém o livro) poderia mapear para editora. Caso descubra, os campos “editora” de todos os livros dessa fonte podem ser preenchidos com os valores corretos, e o sistema também aprendeu um mapeamento semântica correto durante o processo [20].
* *Feedback* voluntário com gratificação atrasada: Em certos domínios, os usuários podem estar dispostos a fornecer *feedback* se eles sabem que tal *feedback* trará benefícios a longo prazo. Por exemplo, uma equipe de desenvolvimento pode trabalhar junta para construir um sistema de integração de dados sem ter qualquer gratificação imediata. Dentro de uma intranet da organização, os funcionários poderão entender que os benefícios de fornecer o *feedback* virão a longo prazo e estão se dispuserem a fazê-lo para ajudar a construir sistemas sobre os dados organizacionais. Pode ser também instituído um mecanismo de distribuição de créditos para os contribuintes, semelhante aos utilizados por muitos sites colaborativos, como epinions.com e amazon.com [20].
* *Piggyback* sobre os traços do usuário: Como exemplo podemos citar sistema de correção ortográfica ou que explora traços de usuários em um motor de busca. Isso nos dá um fluxo constante de usuários, mas ainda temos que resolver o desafio de determinar como esses traços podem ser explorados dependendo do propósito [20].
* Criação de competições: sistemas que utilizam artifícios como atribuir pontuação por colaborações satisfatórias, fazendo com que haja uma competitividade entre os usuários que fornecem o *feedback*, estimulando assim a colaboração por parte de quem utiliza o sistema.
* Oferecer situações de propriedade: um usuário pode ter a sensação de que "possui" uma parte do sistema e, portanto, se sente na obrigação de "cultivar" essa parte. Por exemplo, zillow.com exibe casas e estima seu preço de mercado. Ele fornece uma maneira que permita que o proprietário de uma casa conteste a respeito do preço da sua casa e forneça dados corretos (tais como número de quartos) que, por sua vez, ajuda a melhorar a estimativa de preço da casa [20].

**Que contribuições os usuários podem proporcionar?** Em muitos sistemas simples de *crowdsourcing,* os tipos de contribuição que os usuários podem dar são normalmente um pouco limitados. Podem, por exemplo, avaliar, revisar, classificar ou colocar rótulos; no caso de compartilhamento, os usuários adicionam itens a um site central; em rede, usuários ficam ligados a outros usuários [20].

Em sistemas mais complexos os usuários muitas vezes podem fazer uma gama maior de contribuições. Por exemplo, quando está sendo construída uma base de conhecimentos estruturada, os usuários podem adicionar uma URL, adicionar rótulos, fornecer pares atributo-valor. Mas eles podem também fornecer regras de inferência, resolver questões controversas ou e mesclar entradas conflitantes [25]. O desafio é definir esta gama de possíveis contribuições para projetar o sistema de tal forma que ele possa reunir uma multidão crítica para tais contribuições.

Direcionados a esse objetivo, devemos considerar quatro fatores importantes. Primeiro, o grau de complexidade exigido nas contribuições. Sistemas de colaboração em massa organizam os usuários em grupos, tais como hóspedes, frequentadores, editores, administradores, e "ditadores". Devemos tomar o cuidado de projetar tipos de contribuições apropriadas para esses diferentes grupos de usuários. Usuários de baixa patente devem fazer apenas contribuições simples. Se a dificuldade pra contribuir é alta, eles podem ficar relutantes em participar. Já usuários de alto escalão (editores ou administradores) estão mais dispostos a fazer grandes contribuições (como a resolução de questões controversas, por exemplo).

Qual deve ser o impacto de uma contribuição? Podemos medir o impacto potencial considerando a forma como a contribuição potencialmente afeta o sistema. A edição de uma frase em uma página da Wikipedia, por exemplo, em grande parte afeta apenas a página, enquanto que a revisão de uma política de edição pode potencialmente afetar milhões de páginas. Da mesma maneira, sinalizar um pedaço de dados incorreto normalmente tem menos impacto potencial do que fornecer uma regra de inferência. Quantificar o impacto potencial de um tipo de contribuição em um sistema complexo pode ser uma tarefa não trivial. [26, 27] Mas é importante fazê-lo, porque normalmente têm muito menos usuários de alta patente (editores e administradores) do que regulares, por exemplo. Para maximizar a contribuição total desses poucos usuários, devemos pedir-lhes contribuições de alto impacto potencial sempre que possível [20].

A quantidade de contribuições automáticas. Se um sistema de *crowdsourcing* emprega um algoritmo para realizar uma tarefa é porque seria inviável que os humanos a executassem. Queremos que usuários humanos façam contribuições que sejam fáceis para os seres humanos, porém difícil para as máquinas. Por exemplo, examinar descrições textuais e de imagem é relativamente fácil para os seres humanos, mas muito difícil para as máquinas. Em suma, o trabalho deve ser distribuído entre os usuários humanos e máquinas de acordo com o que cada um deles faz melhor [20].

Por ultimo, é importante que a interface do sistema seja amigável para o usuário fornecer sua contribuição. Isso não é uma tarefa fácil. Interfaces em que o usuário possa se expressar em linguagem natural (como em openMind.org) são bastante cômodas para os usuários, mas se torna uma tarefa de difícil entendimento para as máquinas, e um formato de linguagem formal ocorre o problema inverso. Portanto, é preciso encontrar um meio termo para que o sistema não exija um grande esforço de nenhuma das partes [20].

**Como combinar as contribuições dos usuários?** Muitos sistemas de colaboração em massa não combinam contribuições, ou o fazem de uma forma solta. Grande parte dos sistemas de avaliação, por exemplo, não combinam as análises, ou combinam com a utilização de fórmulas relativamente simples. Sistemas em rede simplesmente fazem a ligação entre as contribuições (homepages e relacionamentos) para formar um grafo da rede social. Sistemas mais complexos de colaboração em massa, como aqueles que constroem softwares, bases de conhecimentos, sistemas de jogos, combinam contribuições de forma mais elaborada. Portanto o tipo da combinação depende da aplicação. A Wikipedia, por exemplo, permite aos usuários mesclar manualmente as edições, enquanto ESP faz isso automaticamente, esperando apenas que os dois usuários concordem no caso de uma palavra comum [20].

Não importa como as contribuições são combinadas, um problema chave é decidir o que fazer se os usuários são diferentes. Imagine um caso em que três usuários afirmam "A" e dois usuários afirmam "não A." Soluções automáticas e manuais foram desenvolvidas para este problema. Soluções automáticas normalmente combinam contribuições ponderadas por algumas pontuações do usuário [20].

Soluções de gerenciamento manual de debates tipicamente permitem que usuários discutam e resolvam entre si. Questões não resolvidas serão resolvidas pela hierarquia do usuário. Sistemas como o Wikipedia e Linux empregam tais métodos. Soluções automáticas são mais eficientes, mas eles só funcionam para as formas relativamente simples de contribuições. Soluções manuais ainda são a melhor forma para combinar contribuições conflitantes em "massa" [20].

**Como avaliar os usuários e as contribuições?** Sistemas de colaboração em massa muitas vezes devem gerenciar os usuários mal-intencionados. Para isso, podemos usar uma combinação de técnicas para bloquear, detectar e desencorajar esses usuários. Primeiro, podemos bloquear usuários mal-intencionados, limitando quem pode fazer que tipo de contribuição. Muitos sistemas e-Science, por exemplo, permitem que qualquer pessoa possa editar dados, mas apenas determinados cientistas têm domínio para limpar e mesclar esses dados no banco de dados central [20].

Em segundo lugar, podemos detectar usuários mal-intencionados através de técnicas manuais, que incluem o monitoramento do sistema por parte dos proprietários, distribuindo a carga de trabalho de monitoramento entre um conjunto de usuários de confiança, e alistando usuários comuns (como quadros de mensagens com sinalização de contribuições ruins, por exemplo). Métodos automáticos normalmente envolvem alguns testes, por exemplo um sistema pode fazer perguntas aos usuários que ele já sabem as respostas e, em seguida, usar as respostas dos usuários para calcular o seu *score* de confiabilidade. [20] Muitos outros esquemas para calcular a segurança, confiança, fama e reputação dos usuários têm sido propostos. [28, 29]

Finalmente, podemos inibir usuários mal-intencionados com ameaças de "punição". Uma punição comum é a proibição. A forma mais recente, porém controversa de punição é a "humilhação pública", onde um usuário malicioso é julgado publicamente e marcado como mal-intencionados para o resto da comunidade (possivelmente sem o conhecimento do usuário) [20].

Não importa quão bem conseguimos inibir os usuários mal-intencionados, as contribuições maliciosas muitas vezes ainda se infiltram no sistema. O sistema de colaboração em massa deve, portanto, encontrar uma maneira de desfazê-las. Se o sistema não combina contribuições ou faz isso apenas de uma forma muito trivial, desconsiderar esses comentários mal-intencionados é relativamente fácil. Porém, se o sistema combina fortemente as contribuições, mas as mantém localizadas, então ainda podemos desfazer esses registros de maneira relativamente simples. Por exemplo, usuários que editam na Wikipédia podem ter suas colaborações combinadas diversas vezes em uma única página, porém a contribuição se manteve localizada nessa página (não propagadas para outras páginas). Entretanto, se as contribuições são colocadas profundamente no sistema, desfazê-las pode ser muito difícil. Suponha que uma regra de inferência R contribuiu para uma base de conhecimento no dia 1. Em seguida, usamos R para inferir novos fatos, aplicar outras regras para eles e para outros já existentes na base de conhecimento para inferir mais fatos, permitir que os usuários editem esses fatos extensivamente, e assim por diante. No dia 3, se fosse encontrado algo incorreto em R, seria muito difícil de remover, sem reverter a base de conhecimento R para seu estado no dia 1, perdendo assim todas as contribuições feitas entre os dias 1 e 3 [20].

No outro extremo, muitos sistemas que utilizam a coleta de *feedback* também identificam e alavancam os usuários influentes, usando tanto técnicas manuais quanto automáticas. Alguns usuários da redes sociais de compra/venda, por exemplo, utilizam bastante a influência de outros usuários para tomar decisões. Alguns trabalhos têm analisado como identificar automaticamente estes usuários e aproveitá-los em marketing viral dentro de uma comunidade [30].

## Considerações

Neste capítulo, falamos a respeito dos sistemas de colaboração em massa, definindo estes tipos de sistemas, mostrando como eles podem ser classificados e discutindo algumas das principais questões que devem ser levantadas antes de implementar um sistema de colaboração em massa.

Com esse estudo, constatamos a importância de enquadrar um sistema de colaboração em massa em algumas dessas dimensões antes da sua implementação. Este processo ajuda a definir como será o funcionamento do sistema e que tipo de usuários ele vai atingir.

No capítulo 4, falaremos da nossa contribuição neste trabalho: a ferramenta RDF Fan. Primeiramente iremos mostrar em que contexto a ferramenta será utilizada e em seguida detalharemos a respeito da sua implementação.

# RDF Fan - Ferramenta de coleta de feedback para triplas RDF

Neste capítulo, será descrita a ferramenta de coleta de *feedback*, denominada RDF Fan, implementada neste trabalho. Primeiramente, iremos classificar a ferramenta de acordo com o que foi discutido nas seções 3.3 e 3.4. Em seguida, iremos apresentar o contexto em que esta ferramenta foi desenvolvida, mostrando a arquitetura global onde o RDF Fan está inserido. Em seguida, serão mostradas as características e os principais objetivos da ferramenta proposta. Por último, serão mostrados detalhes técnicos da implementação.

## Classificação da ferramenta

É fundamental classificar um sistema de colaboração em massa para que se saiba, por exemplo, o que deve ser exigido dos usuários em forma de *feedback*, que grupo de usuários irá utilizar o sistema, qual o grau de complexidade da colaboração exigida. Por essa causa, é importante que, antes de mostrar a ferramenta RDF Fan em detalhes, nós analisemos como ela deve se comportar em relação às questões envolvendo sistemas de colaboração em massa vistas no capítulo anterior. Iremos agora utilizar a discussão dos capítulos anteriores para classificar a ferramenta proposta:

* **Sistema explícito**: o usuário sabe que está fazendo uma anotação de *feedback*. Ele tem o papel de avaliar itens retornados na consulta para refinar cada vez mais o sistema de integração de dados. No caso da nossa ferramenta, o usuário tem a liberdade de fornecer o *feedback* de todos os itens retornados, de parte dos resultados ou simplesmente não colaborar com o *feedback*.
* ***Standalone***: todo sistema de coleta de *feedback* explícito é *standalone*. No nosso caso é um sistema *standalone* que recruta usuários para avaliar itens retornados pelo sistema.
* **Construção de artefato permanente**: ao fornecer *feedback*, o usuário estará colaborando para a construção de um sistema de integração de dados cada vez mais completo, com fontes de dados mais precisas. Como as fontes de dados são parte fundamental de um sistema de integração de dados, ao colaborar para refinar essas fontes de dados, o usuário estará ajudando a aprimorar o sistema, que é o artefato permanente.
* **Execução de tarefas**: é necessário que os usuários executem tarefas, tais como verificar as triplas retornadas, avaliá-las (todas ou algumas) e atribuir um grau de confiança dessa anotação. Essas contribuições, em conjunto, servirão para que o sistema de integração de dados retorne valores cada vez mais relevantes.

Agora faremos uma análise de como a ferramenta se comporta em relação aos quatro desafios dos sistemas de colaboração em massa.

**Como recrutar e reter usuários?**

O usuário que utiliza a nossa ferramenta para anotar *feedback* é um especialista do domínio em que o sistema de integração de dados está imerso. Por conta disso, na maioria das vezes, ele possui o conhecimento de alguns resultados, sejam corretos ou incorretos. Com isso, ele pode avaliar os resultados e sugerir novas fontes caso alguns valores conhecidos não tenham sido retornados. Por esse motivo, nossa ferramenta não tem a preocupação em recrutar usuários.

Assim como na maior parte dos sistemas de colaboração em massa, o *feedback* é voluntário. O usuário precisa sentir a necessidade de fornecer sua colaboração. Ele anota *feedback* para todos os itens, para parte deles ou para nenhum deles.

**Que contribuições os usuários podem proporcionar?**

Como vimos anteriormente, os usuários podem contribuir com o *feedback* de diferentes maneiras dependendo do problema-alvo do sistema de colaboração em massa. Para isso, é importante saber que tipo de usuários pode fornecer determinado *feedback*, quais as modalidades de *feedbacks* que cada usuário pode fornecer, qual deve ser o grau de complexidade exigido para fazer essas anotações, o impacto que cada *feedback* pode causar no sistema e a facilidade para o usuário contribuir.

Como os usuários da nossa ferramenta são especialistas e o objetivo do *feedback* é bem definido, existe apenas um tipo de usuário e de contribuição. As anotações coletadas serão utilizadas para escolher que fontes de dados são mais relevantes para o sistema de integração.

Por conta do tipo de usuário que irá utilizar a ferramenta, o grau de complexidade das contribuições pode ser um pouco mais alto. Além de avaliar os itens e adicionar URI para novos resultados (false negatives), o usuário precisa estimar um grau de confiança para a sua avaliação.

O impacto das contribuições interfere no resultado do cálculo de relevância, de forma que o *feedback* do usuário reflete quais são os seus requisitos para que o retorno das consultas do sistema seja satisfatório. Estes requisitos, baseados no *feedback* do usuário, influenciam na classificação das fontes com relação à relevância destas para as consultas realizadas pela aplicação. Quanto mais anotações forem realizadas, será mais precisa a escolha das fontes, de acordo com a sua relevância.

Outro fator de grande importância para que o sistema tem uma maior participação do usuário é se o sistema tiver uma interface amigável. Com base nisso, a ferramenta foi implementada com a preocupação de ter uma boa usabilidade. Essa característica é fundamental para não desencorajar o usuário de anotar o *feedback*.

**Como combinar as contribuições do usuário?**

O artefato gerado pela ferramenta será composto por anotações de *feedback* sobre os resultados das consultas. Diversos usuários podem realizar consultas, consequentemente fornecer *feedback*. Se dois ou mais usuários realizam feedback sobre um mesmo resultado, ambas as anotações estarão no artefato gerado pela ferramenta. Em nossa proposta, a aplicação que deve se preocupar em tratar essas anotações que fazem referência aos mesmos valores.

**Como avaliar os usuários e as contribuições?**

Neste protótipo não iremos tratar o problema de avaliação de qualidade do *feedback*. Neste caso, supomos que os usuários realizam o *feedback* correto sobre o resultado das consultas. Caso o usuário não tenha conhecimento suficiente para realizar anotações, este não é obrigado a fornecer *feedback* sobre os resultados das consultas.

## Visão geral

A ferramenta de coleta de *feedback* proposta neste trabalho faz parte de um sistema maior proposto por [13], denominado RDFilter, o qual tem como objetivo principal a identificação de novas fontes de dados relevantes em soluções de integração de dados que utilizam uma abordagem incremental. Este sistema faz uso de anotações fornecidas pelos usuários para medir a relevância das fontes com relação aos requisitos dos usuários do sistema. Estas anotações podem ser classificadas em um dos seguintes casos: i) resultados esperados pelo usuário (*true positives* ou TP); ii) resultados que não estão de acordo com os requisitos do usuário (*false positives* ou FP); e iii) resultados omissos (*false negative*s ou FN).

Estas anotações são utilizadas em [13] para o cálculo dos valores de *Precision* e *Recall* obtidos a partir do *feedback* dado para cada consulta. De maneira mais específica, temos que *Precision* é a razão entre o número de *true positives* e a soma de *true positives* e *false positives*. Similarmente, *Recall* é a razão do número de *true positives* e a soma de *true positives* e *false negatives*.

O sistema RDFilter é composto por módulos, sendo a ferramenta RDF Fan um deles. A seguir será explicada a arquitetura global do RDFilter, apresentada na Figura 1, de forma mais detalhada:

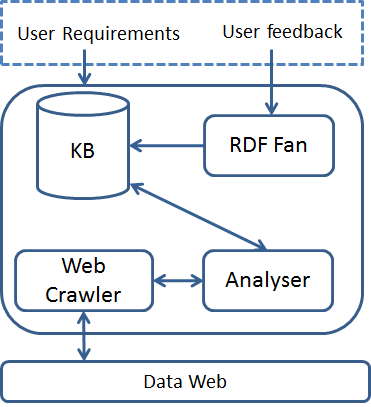


Figura 1 Arquitetura do RDFilter

Assim como todo sistema de integração de dados, o usuário utiliza o sistema fazendo consultas sobre uma base de conhecimentos. No nosso caso, as consultas são feitas em SPARQL, linguagem padrão para consultar dados em RDF, um modelo para descrição de informações na Web com sintaxe e semântica interpretável por máquina.

O usuário tem a possibilidade de avaliar a qualidade dos resultados retornados pelo sistema através do módulo RDF Fan, que armazena anotações de *feedback* em uma base de conhecimento.

O Web Crawler é o módulo responsável por navegar pela Web coletando triplas RDF e povoar a base de conhecimento. Os dados coletados pelo Web Crawler passam pelo módulo Analyser. Este módulo tem a tarefa de avaliar a relevância da fonte candidata a entrar no sistema, fazendo uso do *feedback* obtido sobre o resultado das consultas.

Agora que vimos como funciona o sistema RDFilter de maneira geral, podemos explicar detalhadamente a ferramenta RDF Fan proposta no trabalho.

## Especificação e Implementação

A ferramenta RDF Fan foi implementada na linguagem Java por meio da IDE Eclipse. Como ferramenta de auxílio foi utilizada a IDE NetBeans, que ajudou na implementação da interface gráfica.

A linguagem Java foi escolhida por diversos motivos. Um deles foi por causa do Jena[[5]](#footnote-5), API Java para criação de aplicações para Web Semântica (manipulação de ontologias). Outra razão foi que Java é uma linguagem portável, ou seja, é independente de plataforma, podendo ser facilmente integrada com outros sistemas operacionais.

Iremos agora descrever o funcionamento da ferramenta em cinco passos:

Passo 1: Selecionar uma consulta e seu resultado.

A primeira ação que deve ser realizada pelo usuário da ferramenta é selecionar um arquivo com uma consulta SPARQL e outro com o seu resultado (triplas RDF). É sobre esse resultado selecionado que o usuário vai gerar anotações. Esta ação está ilustrada na Figura 2.

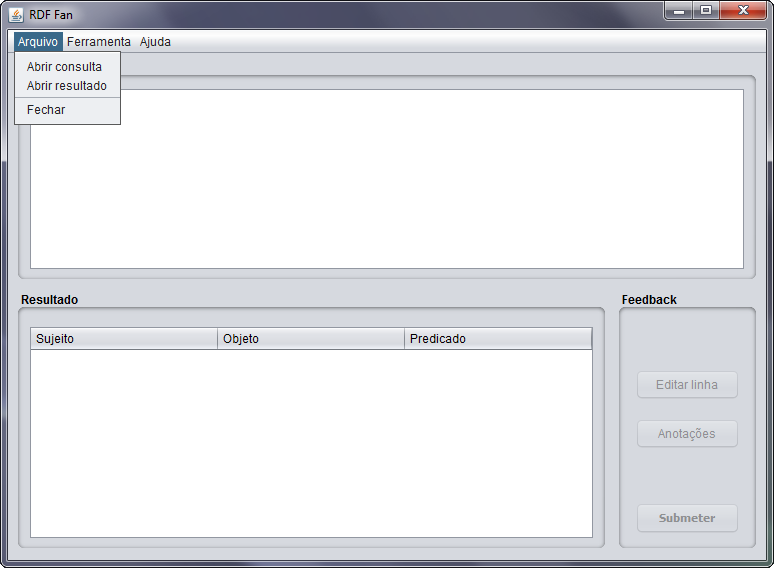


Figura 2 Escolha da consulta e do resultado

Passo 2: Apresentar o resultado para o usuário

A consulta e seu resultado serão mostrados na interface da ferramenta como ilustrado na Figura 3. Os botões de “Editar linha”, “Anotações” e “Submeter” só serão habilitados se a consulta e seu resultado estiverem sido carregados na ferramenta.

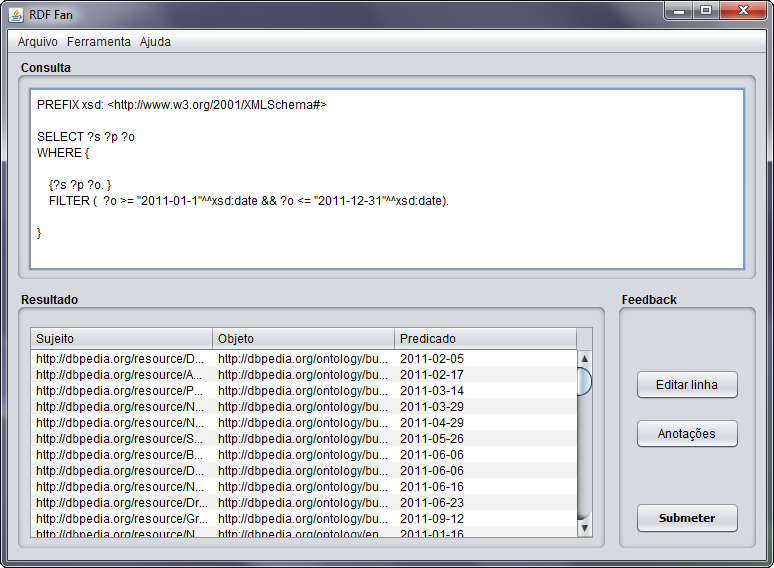


Figura 3 Consulta e resultado carregados

Passo 3: Adquirir *feedback* do usuário

O usuário tem a opção de inserir anotações de *feedback* sobre uma tupla selecionada na tabela de resultados. Essa ação consiste em selecionar “*true positive*”, no caso da linha estar correta, ou “*false positive*”, em caso da linha estar errada. Abaixo, o usuário deve colocar o grau de confiança da sua avaliação. Essa ação pode ser repetida para todas as tuplas retornadas. A tela de edição da linha é mostrada na Figura 4.

O usuário pode também adicionar uma nova linha de anotação acrescentando uma tripla RDF às anotações. Para isso ele precisa estar na tela de Anotações (Figura 5) e selecionar a opção “Adicionar”. Irá aparecer uma tela semelhante à de Editar resultado (Figura 4). Além de adicionar, o usuário também pode remover as anotações que ele não quer mais submeter como *feedback*.

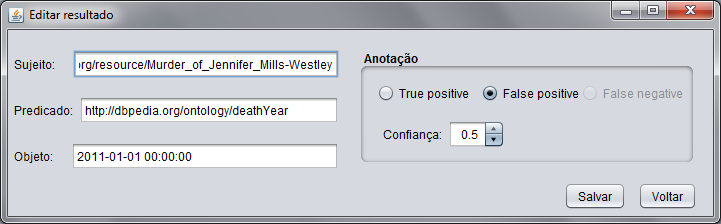


Figura 4 Tela para edição dos resultados

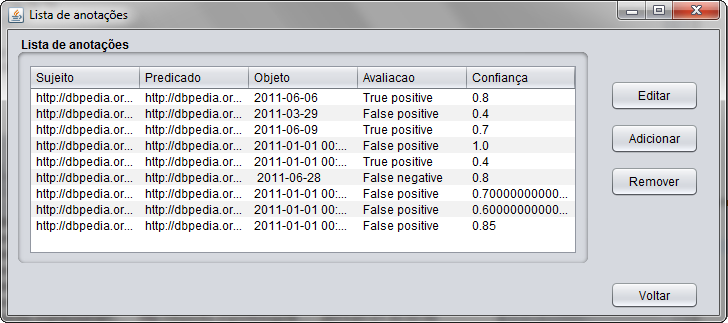


Figura 5 Tela de lista de avaliações

Passo 4: Submeter *feedback*.

Após avaliar as tuplas, o usuário aperta o botão “Submeter” mostrado na Figura 2. Será gerado um arquivo XML com as consultas e as anotações relativas a elas. O arquivo gerado será utilizado como entrada para o módulo de avaliação das fontes de dados. Este artefato gerado guarda uma lista de consultas, que possuem anotações que, por sua vez, são compostas por uma tripla, um identificador da consulta, a avaliação da linha (TP, FP ou FN) e o grau de confiança na avaliação, como pode ser visto na Figura 8.

****

Figura 6 Output gerado pela ferramenta RDF Fan

Passo 5: Visualizar estatísticas

A partir das anotações de *feedback*, são geradas estatísticas que podem ser utilizadas para avaliar a qualidade do retorno da consulta. Estas estatísticas serão recuperadas pelo módulo de análise das fontes, onde serão realizados todos os cálculos que forem apresentados nesta tela. A tela de Estatísticas pode ser vista na Figura 8. Na Figura 7 é mostrada a opção de Gerar estatísticas.

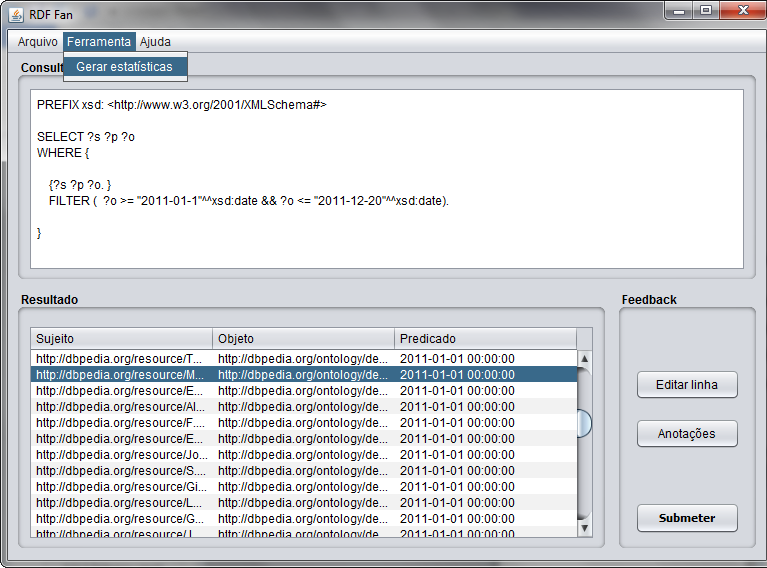


Figura 7 Menu de Geração estatísticas

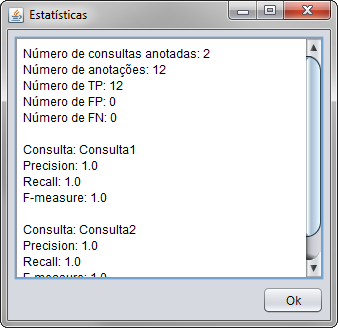
****

Figura 8 Tela de Estatísticas

Passo 6: Armazenar estatísticas na base de conhecimento

As estatísticas geradas a partir das anotações são armazenadas em um repositório e podem ser visualizadas pelo usuário (Figura 8).

## Considerações

Neste capítulo, classificamos a ferramenta proposta no trabalho de acordo com as dimensões apresentadas no Capítulo 3 e mostramos como o RDF Fan responde aos desafios dos sistemas de colaboração em massa. Em seguida, apresentamos, com um pouco mais de detalhes, como foi implementada a ferramenta e apresentamos um exemplo de fluxo básico da sua execução.

A ferramenta RDF Fan foi validada pelo aluno de mestrado que está desenvolvendo o RDFitter, e terá grande valia para auxiliar o seu projeto.

No próximo capítulo, veremos a conclusão do trabalho e alguns possíveis trabalhos futuros.

# Conclusão

Neste trabalho, foi implementada a RDF Fan, uma ferramenta de coleta de *feedback* de usuários para sistemas de integração de dados *pay-as-you-go*. A criação da ferramenta veio da necessidade de coletar o *feedback* de usuários por parte de um sistema maior que está sendo implementado como trabalho de mestrado na UFPE. A ferramenta irá possibilitar que usuários colaborem para o refinamento de fontes de dados em um sistema de integração de dados *pay-as-you-go*.

Para conduzir a implementação da ferramenta, foi necessário fazer um estudo sobre *feedback* do usuário e sistemas de colaboração em massa. Este estudo foi necessário para saber de que maneira o sistema deveria funcionar, mais especificamente, quais requisitos seriam importantes para que o sistema de colaboração em massa funcionasse de forma satisfatória.

A partir dessa pesquisa, constatamos a importância do *feedback* do usuário como forma de auxiliar na construção de sistemas de integração de dados *pay-as-you-go.*

Como parte dos trabalhos futuros, podemos destacar o desenvolvimento de uma abordagem para tratar as anotações de *feedback* conflitantes, que são anotações diferentes sobre o mesmo resultado. Outra questão que pode ser abordada diz respeito à detecção e tratamento de usuários que venham a utilizar a ferramenta de maneira mal intencionada.

Como atividade futura para refinar a ferramenta, é necessário que seja analisada, de maneira mais minuciosa, quais dados estatísticos são relevantes para o usuário.

# Referências Bibliográficas

[1] Siedler, M. S. (2004). “Sistema de Integração de Dados usando Técnicas de Web Semântica”. Dissertação de Mestrado, Centro de Informática, Universidade Federal de Pernambuco, Recife, Brasil, 2004.

[2] Halevy, A. Y., Rajaraman, A., Ordille, J. J. (2006). “Data integration: The teenage years”. In: Proceedings of the 32nd International Conference on Very Large Data Bases, VLDB 2006, 9-16.

[3] Belhajjame, K., Paton, N. W., Fernandes, A. A. A. Hedeler, C., Embury, S. M. (2011). “User Feedback as a First Class Citizen in Information Integration Systems”. In: Proceedings of the 5th Conference on Innovative Data Systems Research, CIDR 2011, 175-183.

[4] Alexe, B., Chiticariu, L., Miller, R. J., Tan., W. C. (2008). “Muse: Mapping understanding and design by example”. In: Proceedings of the 24th International Conference on Data Engineering, ICDE 2008, 10-19.

[5] Belhajjame, K., Paton, N. W., Embury, S. M., Fernandes, A. A. A., Hedeler, C. (2010). “Feedback-Based Annotation, Selection and Refinement of Schema Mappings for Dataspaces”. In: Proceedings of the Extending Database Technology, EDBT 2010, 573–584.

[6] Cao, H., Qi, Y., Candan, K. S., Sapino, M. L. (2010). “Feedback-driven result ranking and query refinement for exploring semi-structured data collections”. In: Proceedings of the 13th International Conference on Extending Database Technology, EDBT 2010, 3-14.

[7] Chai, X., Vuong, B. Q., Doan, A., Naughton, J. F. (2009). “Efficiently incorporating user feedback into information extraction and integration programs”. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2009, 87-100.

[8] Jeffery, S. R., Franklin, M. J., Halevy, A. Y. (2008). “Pay-as-you-go User Feedback for Dataspace Systems”. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, 847-860.

[9] McCann, R., Shen, W., Doan, A. (2008). “Matching schemas in online communities: A web 2.0 approach”. In: Proceedings of the 24th International Conference on Data Engineering, ICDE 2008, 110-119.

[10] Talukdar, P. P., Ives, Z. G., Pereira, F. (2010). “Automatically incorporating new sources in keyword search-based data integration”. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2010, 387-398.

[11] Talukdar, P. P., Jacob, M., Mehmood, M. S., Crammer, K., Ives, Z. G., Pereira, F., Guha, S. (2008). “Learning to create data-integrating queries”. In: Proceedings of the Very Large Data Bases, PVLDB 2008, 785-796.

[12] Heath T., Bizer C. (2011). “*Linked Data: Evolving the Web into a Global Data Space”* (1st edition). Synthesis Lectures on the Semantic Web: Theory and Technology, 1-136.

[13] Oliveira, H. R., Lóscio, B. F. (2011). “Avaliando a Relevância de Novas Fontes de Dados em Sistemas de Integração de Dados Pay-as-you-go”. In: Proceedings of the X Workshop de Teses e Dissertações em Banco de Dados, WTDBD 2011.

[14] Lóscio, B. F. (2003). “Managing the Evolution of XML-based Mediation Queries”. PhD thesis, Informatics Center, Federal University of Pernambuco, Recife, Brasil, 2003.

[15] Lóscio, B. F. e Salgado, A. C.(2001), “Integração de Dados na Web”. In:Proceedings of the Jornada de Atualização em Informática, CSBC, JAI 2001

[16] McCann, R., AlShebli, B. K., Le, Q., Nguyen, H., Vu, L. Doan, A. (2005). “Mapping Maintenance for Data Integration Systems”. In: Proceedings of Very large Data Bases Conference,VLDB 2005, 1018–1030.

[17] [Madhavan](http://research.google.com/pubs/author6192.html) J., Jeffery S. R., Cohen S., Xin (Luna) Dong, Ko, D., [Cong Yu](http://research.google.com/pubs/CongYu.html), [Halevy](http://research.google.com/pubs/author1112.html) A. Y. (2007). “Web-scale Data Integration: You can only afford to Pay As You Go”. In: Proceedings of the Conference on Innovative Data Systems Research, CIDR 2007, 342-350.

[18] Vaz Salles, M. A., Dittrich, J. P., Karakashian, S. K., Girard, O. R., Blunschi, L. (2007). “iTrails: Pay-as-you-go Information Integration in Dataspaces”. In: Proceedings of Very Large Data Bases Conference, VLDB 2007, 663-674.

[19] Koutrika, G., Bercovitz, B., Kaliszan, F., Liou, H. and Garcia-Molina, H. (2009). “Courserank: A closed-community social system through the magnifying glass”. In: Proceedings of the 3rd Int'l AAAI Conference on Weblogs and Social Media, ICWSM 2009, 17-20.

[20] Doan, A., Ramakrishnan, R., Haleyy, A. Y. (2011). “Crowdsourcing Systems on the World-Wide Web”, In: Communications of the ACM, CACM, 54 (4), 86-96.

[21]. Ramakrishnan R., Baptist A., Ercegovac V., Hanselman M., Kabra N., Marathe A., Shaft U. (2004). “Mass collaboration: A case study”. In: Proceedings of the Database Engineering and Applications Symposium, IDEAS 2004, 133-146.

[22] von Ahn, L, Dabbish, L. (2004). “Labeling images with a computer game.” In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2004, 319-326.

[23] von Ahn, L. Dabbish, (2008). “Designing games with a purpose”. In: Communications of the ACM - Designing games with a purpose. CACM, 51 (8), 58–67.

[24] You – Yes, You – Are TIME’s Person of the Year 2006; <http://www.time.com/time/magazine/article/0,9171,1569514,00.html>. Acessado em 13/12/2011

[25]. Richardson, M., Domingos, P. (2003). “Building large knowledge bases by mass collaboration”. In: Proceedings of the 2nd International Conference on Knowledge Capture, K-CAP 2003, 129-137.

[26]. McCann, R., Doan, A., Varadarajan, V., and Kramnik, A. (2003). “Building data integration systems: A mass collaboration approach”. In: Web and Databases, WebDB 2003, 183-188.

[27]. McCann, R., Shen, W. and Doan, A. (2008) “Matching schemas in online communities: A Web 2.0 approach”. In Proceedings of International Conference on Data Engineering, ICDE 2008, 110-119.

[28] Kasneci, G., Ramanath, M., Suchanek, M. and Weiku, G. (2008). “The yago-naga approach to knowledge discovery”. In: Proceedings of the SIGMOD Record. SIGMOD 2008, 37(4), 41-47.

[29] Sarwar, B.M., Karypis, G., Konstan, J.A. and Riedl, J. (2001). “Item-based collaborative filtering recommendation algorithms”. In: Proceedings of International World Wide Web Conference, WWW 2001, 285-295.

[30] Richardson, M., Domingos, P. (2002). “Mining knowledge sharing sites for viral marketing”. In: Proceedings of International Conference on Knowledge Discovery and Data Mining, KDD 2002, 61-70.

1. <http://www.wikipedia.org/> [↑](#footnote-ref-1)
2. <http://www.linux.org/> [↑](#footnote-ref-2)
3. <http://br.answers.yahoo.com/> [↑](#footnote-ref-3)
4. <https://www.mturk.com/mturk/welcome> [↑](#footnote-ref-4)
5. <http://jena.sourceforge.net/> [↑](#footnote-ref-5)