UNIVERSIDADE ESTADUAL DE MARINGÁ
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ARIEL GUSTAVO ZUQUELLO

OERecommender: um sistema de recomendação de REA para MOOC

Maringá
2015

ARIEL GUSTAVO ZUQUELLO

OERecommender: um sistema de recomendação de REA para MOOC

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Departamento de Informática, Centro de Tecnologia da Universidade Estadual de Maringá, como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dra. Itana Maria de Souza Gimenes

Maringá
2015

Z95o Zuquello, Ariel Gustavo
OERecommender: um sistema de recomendação de REA para MOOC / Ariel Gustavo Zuquello. - 2015.
116 f.: il.; 30 cm

Orientadora: Itana Maria de Souza Gimenes
Bibliografia: p.109-116
Dissertação (mestrado) - Universidade Estadual de Maringá. Centro de Tecnologia, Mestrado em Ciência da Computação. Maringá, 2015.

1. OERecommender. 2. Sistemas de recomendação. 3. MOOC. I. Itana Maria de Souza Gimenes.II. Universidade Estadual de Maringá. Mestrado em Ciência da Computação. III.Título.

CDD: 004

Ficha catalográfica elaborada por: Marilene dos Santos Franceschi – CRB-14/812

FOLHA DE APROVAÇÃO

ARIEL GUSTAVO ZUQUELLO

OERecommender: um sistema de recomendação de REA para MOOC

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Departamento de Informática, Centro de Tecnologia da Universidade Estadual de Maringá, como requisito parcial para obtenção do título de Mestre em Ciência da Computação pela Banca Examinadora composta pelos membros:

BANCA EXAMINADORA

Profa. Dra. Itana Maria de Souza Gimenes
Universidade Estadual de Maringá – DIN/UEM

Prof. Dr. Edson Alves de Oliveira Junior
Universidade Estadual de Maringá – DIN/UEM

Profa. Dra. Mariangela de Oliveira Gomes Setti
Universidade Tecnológica Federal do Paraná – DAINF/UTFPR

Aprovada em: 24 de agosto de 2015.
Local da defesa: Sala 101, Bloco C56, *campus* da Universidade Estadual de Maringá.

AGRADECIMENTO(S)

Primeiramente a Deus pela força e luz nos momentos mais difíceis, sempre esteve comigo. Aos meus pais, Valcir e Leonélia, minha irmã Dayana e minha filha Ana Carolina que sempre me apoiaram. Um agradecimento especial à minha namorada Miriele Favero que sempre teve palavras de carinho e amor em todos os momentos, mesmo quando pensei em desistir, peço desculpas pelas ausências.

Um agradecimento aos meus amigos da turma 2013 do mestrado, Romulo, Nanni, Yoji, Emanuel, Frank, Douglas, Leandro, Rodolfo, Presto, Luciano, Pietro, enfim, todos. Outro agradecimento especial à Inês, secretaria do mestrado, que sempre esteve conosco, lembrar-me-ei sempre com um carinho mais do que especial.

Agradeço também aos professores do DIN da UEM que contribuíram para meu crescimento intelectual, em especial à minha orientadora Itana Maria de Souza Gimenes, sempre com muita paciência e presteza.

EPÍGRAFE

“ O segredo é:
cair sete vezes
e levantar-se
oito”.

(PAULO COELHO)

OERecommender: um sistema de recomendação de REA para MOOC

RESUMO

Massive Open Online Course (MOOC) é a mais nova tendência em recurso educacional a distância; seu objetivo é levar conhecimento para uma grande massa de pessoas em qualquer lugar por meio da Web, na maioria dos casos gratuitamente. Um ponto chave dos MOOCs é oferecer mecanismos de apoio ao processo de aprendizagem aos seus participantes. Sua população de participantes é culturalmente diversa e a taxa de abandono é considerada alta, em torno de 90%. Uma das deficiências reconhecidas é a falta de indicação de materiais abertos que possam enriquecer a base de apoio aos participantes. Para contribuir com a melhoria deste cenário, este trabalho propõe o *OERecommender*, um Sistema de Recomendação de Recursos Educacionais Abertos (REAs) para MOOC. O *OERecommender* visa apoiar os participantes na busca e obtenção de REAs que possam ajudar em seu processo de aprendizagem. A concepção do modelo conceitual foi embasada em arquiteturas similares já existentes do mundo real. Para encontrar a similaridade entre usuários e os REAs optou-se pelo método de grafos. Algoritmos de ordenação e comparação foram utilizados respectivamente para ordenar os REAs por relevância e comparar instâncias para encontrar o contexto mais similar entre os usuários. Ao final, um algoritmo de recomendação foi adotado para prever quais REAs eram mais relevantes ao usuário apresentando-os por meio de um *widget*. Como forma de avaliação do *OERecommender*, foram realizadas simulações por meio de uma prototipação de cenários. Após a execução das simulações, os resultados indicaram que é viável a introdução de mecanismos de recomendação de REA em MOOC, e que contribuem para melhorar o apoio a seus participantes.

Palavras-chave: MOOC, *OERecommender*, REA, Sistemas de Recomendação.

OERecommender, a OER recommendation system for MOOC

ABSTRACT

Massive Open Online Course (MOOC) is one of the newest trend in education at a distance. Its goal is to bring knowledge to a large body of people anywhere through the Web, in most cases free of charge. A key point of MOOCs is to provide mechanisms to support the learning process to its participants. Its population of participants is culturally diverse and the dropout rate is considered high, around 90%. One of its recognized shortcomings is the lack of support from open materials that can enrich participants' learning process. This work proposes the OERecommender, a Recommendation System of Open Educational Resources (OER) that aims to contribute to the improvement of this scenario. The OERecommender aims to support participants in searching and recovering OER that can help in their learning process. The design of the OERecommender conceptual model was based on existing and similar architectures. The similarity between users and OER was based on graph methods. Algorithms for sorting and comparison were used, respectively, to sort OER by relevance and compare instances to find the most similar context among users. Finally, a recommendation algorithm was adapted to predict which REA were most relevant to the user presenting them through a widget. In order to assess the OERecommender, simulations were performed through prototyping scenarios. The simulations indicate that the introduction of OER recommendation mechanisms in MOOC is feasible and can contribute to improve the support to its participants.

***Keywords*:** MOOC, OER, OERecommender, Recommendation System.

LISTA DE QUADROS

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AVA	Ambiente Virtual de Aprendizagem
EAD	Educação a distância
TIC	Tecnologias de Informação e Comunicação
REA	Recursos Educacionais Abertos
CC	*Creative Commons*
MOOC	*Massive Open Online Courses*
CCK	*Connectivism & Connective Knowledge*
SR	Sistema de Recomendação
OA	Objetos de Aprendizagem
CMOOC	*Connectivist Massive Open Online Courses*
XMOOC	*Behavioral Massive Open Online Courses*
WYSIWYG	*What You See Is What You Get*
SAML2	*Security Assertion Markup Language*
XML	*eXtensible Markup Language*
MIT	*Massachusetts Institute of Technology*
LMS	*Learning Management System*
GCB	*Google Course Builder*
HTML	*HyperText Markup Language*
LA	*Learning Analytics*
IEEE	*Institute of Electrical and Electronic Engineers*
GUI	*Graphical User Interfaces*
DOC	*Document*
PDF	*Portable Document Format*
LOM	*Learning Object Metadata*
MIME	*Multipurpose Internet Mail Extensions*
URL	*Uniform Resource Locator*
PHP	*Hipertext preprocessor*

GPL	*General Public License*
API	*Application Programming Interface*
SDK	*Software Development Kit*
UNESCO	Organização das Nações Unidas
COL	Comunidade de Aprendizagem
OECD	Organização para a Cooperação e Desenvolvimento Econômico
ICDE	Conselho Internacional de Educação a Distância
GPS	*Global Positioning System*
JPEG	*Joint Photographic Experts Group*
CAM	*Contextualized Attention Metadata*
UNED	*Universidad Nacional de Educación a Distancia*
CSEV	*Centro Superior para la Enseñanza Virtual*
MTC	*Model-View-Controller*
MERLOT	*Multimedia Educational Resource for Learning and Online Teaching*
MPC	Métrica de Popularidade Composta
MR	*Rank* Manual
PR	*Rank* de Popularidade
SCORM	*Sharable Content Object Reference Model*

SUMÁRIO

1. INTRODUÇÃO

1.1 CONTEXTUALIZAÇÃO

A educação a distância (EAD) teve grande crescimento nos últimos anos em virtude do uso das Tecnologias de Informação e Comunicação (TICs). A Web 2.0 introduziu recursos que permitem uma aprendizagem social em que o aluno não apenas consome informação, mas também produz, colabora e coopera com seus pares (BROWN E ADLER, 2008). Como resultado desse crescimento desencadearam-se vários movimentos, dentre eles o de Educação Aberta.

A Educação Aberta, um movimento que vem se fortalecendo como forma complementar a educação tradicional em qualquer modalidade (MEISZNER, 2011), oferece a possibilidade de aprendizagem gratuita e autônoma, fomentada por instituições de ensino e educadores que valorizam a educação social ampla e para todos.

No contexto da Educação Aberta, têm-se dois importantes elementos: os Recursos Educacionais Abertos (REAs) e os *Massive Open Online Courses* (MOOCs). Os REAs são materiais de ensino, aprendizagem e pesquisa, em qualquer mídia, que estão sob domínio público ou possuem uma licença aberta para que sejam utilizados ou adaptados por terceiros (BUTCHER, 2011; MOTA e INAMORATO, 2012). Podem incluir livros didáticos, artigos acadêmicos, aulas ou cursos completos, software, vídeos, ferramentas, materiais ou técnicas. Esses materiais possuem licenças de autoria como, por exemplo, a *Creative Commons* (CC). O foco das iniciativas REA é disponibilizar e compartilhar várias partes ou unidades do saber, usando a premissa dos 4 R's: Reusar, Revisar, Remixar e Redistribuir (WILEY, 2010); (BUTCHER, 2011) e (SANTANA et al., 2012). Os REAs são atualmente ofertados em diversas plataformas como MIT *Open Courseware* (ocw.mit.edu), *OpenLearn* (open.edu/openlearn), *Connexions* (cnx.org) e *U-Now* (unow.nottingham.ac.uk).

Os MOOCs são cursos *online* abertos, ofertados para uma grande quantidade de pessoas que podem estar em qualquer lugar do mundo em que se tenha acesso à Internet. Este termo foi cunhado em 2008, durante um curso sobre "Conectivismo e Conhecimento Conectivo" promovido pelos canadenses, George Siemens, Stephen Downes e Dave Cornier (CCK08, 2008; DOWNES, 2008). Tal curso foi projetado e executado para 25 alunos pagantes, porém mais outros 2.300 puderam participar do curso gratuitamente pela Internet. O curso utilizou uma variedade de ferramentas de mídia social, incluindo RSS *feeds*, *blogs*,

mundos virtuais e reuniões *online* síncronas. Os MOOCs, diferentemente da EAD tradicional, promovem o surgimento de redes de aprendizagem sem limite de participantes, em que todos estão ao mesmo tempo ensinando e aprendendo de forma ativa (DOWNES, 2008). No contexto da Educação Aberta, um MOOC só pode ser considerado um REA se os seus recursos forem disponibilizados de acordo com os mesmos princípios de compartilhamento e licenças dos REAs.

1.2 MOTIVAÇÃO

O progresso dos MOOCs tem causado muitas discussões em todos os níveis, desde políticos até operacionais. Fala-se em uma possível "revolução" da Educação Aberta digital contemporânea (MOTA e INAMORATO, 2012). Seja uma revolução, ou somente uma nova onda de oportunidades, os maiores beneficiados são as pessoas que com os mais diversos objetivos individuais e profissionais, estão tirando proveito para adquirir conhecimento e melhorar seus currículos, a partir de grandes universidades.

Existe uma série de questionamentos sobre o impacto dos MOOCs na educação tradicional e de como podem ser utilizados pela população. Welsh e Dragusin (2013) acreditam que os MOOCs provavelmente tenderão a ter um impacto maior sobre o ensino superior, devido aos seus principais pontos fortes: conteúdo de alta qualidade fornecidos pelas grandes universidades; a existência de prazos e graus; e o fato da maioria ser ofertada de forma gratuita.

Recentemente, várias *startups* têm surgido de forma independente, como: Coursera (www.coursera.org), Udacity (www.udacity.com), edX (www.edx.org), Veduca (www.veduca.com.br), FutureLearn (www.futurelearn.com) e OpenMOOC (unedcoma.es).

Pesquisas são necessárias para explorar vários aspectos dos MOOCs, tanto do ponto de vista educacional e seus impactos políticos e de negócios, quanto do ponto de vista de recursos computacionais de apoio, que é o foco deste trabalho de mestrado.

Apesar do crescimento dos MOOCs, pouco ainda se sabe sobre a estrutura interna das plataformas de MOOCs, sobre a articulação entre as TICs utilizadas e os recursos de aprendizagem, do ponto de vista pedagógico. Além do questionamento sobre a sustentabilidade dos MOOCs, principalmente sobre os ofertados de forma gratuita, vários trabalhos como Chung (2013) e Siemens (2012) relatam outras dificuldades encontradas em MOOCs. Chung (2013) indica que a evasão está em torno de 90% devido a diversos fatores

como: objetivos dos alunos ao realizarem esses cursos, idioma, cultura, bem como ao fato dos alunos não estarem preparados para receber e assimilar os conteúdos.

Siemens (2012) explica sobre a importância da criação de aplicativos para aumentar a diversidade cognitiva e propõe possíveis soluções, como o incentivo à criação de um universo de aplicativos, próximo ao que foi desenvolvido ao redor do Facebook e outras redes sociais similares, ao redor dos MOOCs, constituindo MOOC *Apps* e envolvendo inúmeras *startups*. Esses fatores somados trariam maior satisfação aos alunos, com isso diminuindo os índices de evasão.

Uma potencial melhoria já aplicada em domínios semelhantes como, por exemplo, em Ambientes Virtuais de Aprendizagem (AVA), são os Sistemas de Recomendação (SR) (BARCELLOS et al., 2007). Um SR tem por objetivo desenvolver modelos para reduzir a sobrecarga de informações que um usuário recebe ao procurar algum produto, por intermédio da recomendação personalizada, com base no perfil do usuário (ADOMAVICIUS e TUZHILIN, 2005). As recomendações geradas sugerem novos recursos de aprendizagem que podem estar internos ou externos à plataforma utilizada pelo usuário, auxiliando assim no aprendizado e motivando o aluno a continuar o curso (HILEN, 2006).

Os REAs são importantes recursos adicionais que podem ser utilizados pelos participantes de MOOCs para ampliar conhecimento ou fornecer conceitos que constituem pré-requisitos ao curso, de modo a apoiar uma população de aprendizes com passados diversos. Alguns MOOCs possuem conteúdo aberto e podem ser considerados REAs, mas muitos possuem conteúdo fechado de propriedade dos fornecedores. Assim, mecanismos que apoiem a utilização de REAs em MOOCs como os SRs, são importantes para proporcionar aos aprendizes uma diversidade de materiais de apoio no processo de ensino aprendizagem. O que sugere formas de certificação do conhecimento adquirido nos cursos, melhorando os currículos dos participantes e aumentando suas chances de sucesso no mercado de trabalho.

1.3 OBJETIVOS

Este trabalho tem por objetivo propor um SR, a ser integrado em uma arquitetura de MOOC para apoiar a utilização de REAs. Para tal utiliza a interpretação dos dados provenientes das interações efetuadas pelos usuários no decorrer do curso. Ferramentas de Análise de Dados de Aprendizagem, do termo inglês *Learning Analytics* (LA), são mecanismos que interpretam esses dados (EDUCASE, 2010), juntamente com os metadados de atenção contextualizada (CAM – *Contextualized Attention Metadata*) coletadas por meio de eventos realizados pelos

usuários no ambiente MOOC. Combinando essas duas fontes de dados, pode-se compor um perfil comportamental do usuário e, consequentemente, fazer recomendações de REAs para ele. Isto pode melhorar o aprendizado e contribuir para reduzir os 90% de evasão que ocorre atualmente.

Como forma de avaliar o SR em uma arquitetura de MOOC, utilizou-se avaliação qualitativa com prototipação baseada em cenários fictícios no contexto educacional, no qual foram utilizados algoritmos de ordenação, comparação e também de predição para prever a utilidade dos REAs que o sistema recomenda ao usuário.

1.4 METODOLOGIA

A metodologia de pesquisa adotada neste trabalho consistiu de: pesquisa bibliográfica utilizando-se da prática de revisão sistemática; projeto conceitual de uma arquitetura para MOOC incluindo um sistema de recomendação e, validação de viabilidade com base em cenários de execução. A condução da revisão caracterizou-se pela identificação, seleção e avaliação dos estudos primários de acordo com critérios de inclusão e exclusão estabelecidos durante a definição do protocolo de revisão (KITCHENHAM, 2004), que em nosso trabalho são conceitos relacionados aos MOOCs.

Os procedimentos metodológicos inicialmente objetivaram mapear um modelo conceitual das arquiteturas atuais das plataformas MOOC para encontrar o atual estado da arte para posteriormente propor o incremento dos componentes necessários nessa arquitetura, concebendo assim um SR de REAs em ambientes MOOC.

O mapeamento da arquitetura se deu por meio de um levantamento de características em diversos ambientes, criando assim uma arquitetura de referência.

A avaliação da proposta foi planejada e executada utilizando simulações com procedimentos reais dos MOOCs, a fim de tornar a simulação o mais próximo da realidade possível.

1.5 ORGANIZAÇÃO

Este trabalho de mestrado apresenta mais cinco capítulos. O Capítulo 2 apresenta uma revisão bibliográfica sobre MOOCs, contendo um panorama geral das arquiteturas das plataformas dos MOOCs, breve explicação sobre Recursos Educacionais Abertos e Análise de Dados de Aprendizagem; CAM – Metadados de Atenção Contextualizada e seus detalhamentos;

Usando CAM para ranquear e recomendar objetos de aprendizagem; Sistema de Pesquisa Federada utilizando *widget; SeeOER* (*Search Engine for Open Education Resources*)– Mecanismo de busca na Web por REAs e Sistemas de Recomendação. No Capítulo 3 é descrita a proposta do *OERecommender* – um SR de REA para MOOCs, objeto principal desta pesquisa. No Capítulo 4 tem-se a avaliação qualitativa por meio de cenários fictícios e, no Capítulo 5 encontram-se as conclusões.

2. REVISÃO BIBLIOGRÁFICA

2.1 CONSIDERAÇÕES INICIAIS

Esta seção contém conceitos importantes para o entendimento do trabalho e constituem a base para o desenvolvimento da presente proposta de mestrado. Sendo eles: apresentação do termo MOOC, sua classificação, evolução nos últimos anos e recursos de apoio aos MOOCs, enfatizando repositórios externos às plataformas; arquiteturas de plataformas existentes; conceitos sobre REAs; componentes e ferramentas de análise de dados de aprendizagem; metadados de atenção contextualizada – CAM seu uso para ranquear e recomendar REAs; sistemas de pesquisa federada utilizando *widget; SeeOER* - mecanismo de busca na Web por REAs e sistemas de recomendação, detalhando os sistemas com filtragem baseada em conteúdo, colaborativa e híbrida. Por fim, têm-se as considerações finais desta seção.

2.2 MOOC

Os MOOCs, em geral, não possuem requisitos formais de entrada, não possuem limite de participantes, são geralmente gratuitos e não há recompensa formal de crédito como nos cursos tradicionais (GAEBEL, 2013).

Desde o primeiro MOOC em 2008, tem-se observado um aumento significativo na quantidade de instituições que os ofertam, como observado na Figura 1.

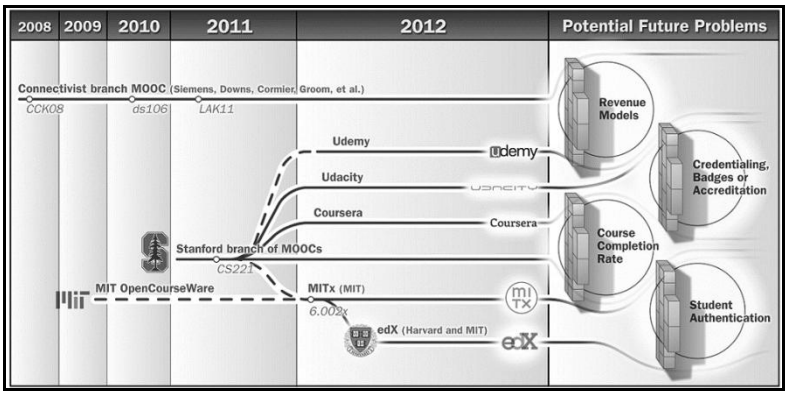


Figura 1 - Evolução dos MOOCs (GAEBEL, 2013)

Inicialmente os MOOCs foram classificados em dois grandes grupos no que tange à estilos e teorias de aprendizagem (Siemens, 2012):

- **cMOOCs:** constituem a primeira geração que começou em 2008. Têm foco na criação e geração de conhecimento; criatividade, autonomia e, as atividades compartilhadas e colaborativas são incentivadas, enriquecendo o conteúdo do curso. Não existe uma relação bem definida entre professores e alunos;
- **xMOOCs:** constituem a segunda geração que começou em 2012. Baseiam-se em um formato mais tradicional, com conteúdo estruturado fixo, apoiado por fóruns de discussão centralizados e automatizados por métodos de avaliação, por exemplo, Coursera, EDX e Udacity. Os alunos aprendem por meio de apresentações de pequenos vídeos e testes. Existe uma relação de um professor para muitos alunos.

Na Figura 2, é possível visualizar um mapa com alguns recursos que apóiam os MOOCs, os quais podem estar disponíveis na própria plataforma ou na Web. Repositórios na Web já são comumente utilizados hoje em dia para oferecer serviços de apoio como, Youtube, *SlideShare* e *Flickr*.

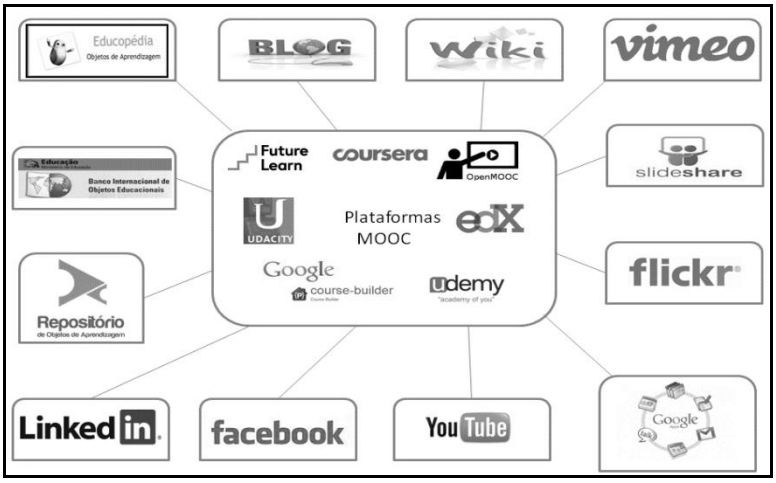


Figura 2 - Mapa com alguns recursos de apoio aos MOOCs (adaptado de (REICH, 2012))

No contexto deste trabalho será adotado o termo genérico MOOC. A princípio o SR a ser desenvolvido independe de uma abordagem de aprendizagem específica.

2.3 ARQUITETURAS DAS PLATAFORMAS DE APOIO A MOOCS

Os MOOCs têm sido ofertados tanto em plataformas proprietárias como em plataformas *Open Source*. No contexto desta dissertação, é importante conhecer as plataformas *Open Source*, as quais permitem a realização de experimentos e a integração de novos componentes. Apesar de existirem diversas plataformas de apoio a MOOCs, muitas dessas não serão abordadas neste

trabalho por serem proprietárias, não possuindo documentações e código-fonte disponíveis. As seções seguintes destacam três plataformas: OpenMOOC, EDX e Google Course Builder, que dentre as várias existentes foram as que mais apresentam documentação de apoio.

2.3.1 OpenMOOC

A plataforma OpenMOOC é uma solução de código aberto e licenciado com *Apache 2.0*, oferecida pela Universidade Nacional de Educação a Distância (UNED) e o Centro Superior para la Enseñanza Virtual (CSEV), uma fundação criada para promover a educação virtual, utilizando as TIC no ensino superior (OPENMOOC, 2013). É baseada em três principais elementos: (i) Vídeo + Internet - o uso combinado de vídeo e fóruns de discussão, o conteúdo está disponível *online* a partir de um *desktop*, *tablet* ou *smartphone*; (ii) Progressivo - os cursos consistem em conjuntos de recursos de aprendizagem e; (iii) Participativa - cada curso está associado a um fórum de discussão em que os alunos e professores podem discutir e colaborar em uma unidade de conhecimento.

Além de exigir uma infraestrutura local própria (servidor, *link* de internet e domínio), as principais características do OpenMOOC são: (i) Solução 100% de código aberto; (ii) Integração de vídeo com documentos e observações do professor; (iii) Interface de criação de curso; (iv) Auto-avaliação do progresso do curso; (v) Fórum de discussão; (vi) Não necessita transmitir vídeos a partir da plataforma local, pois usa repositório fora da plataforma; (vii) Interface *WYSIWYG* para criação de perguntas para avaliação do usuário; (viii) Medalhas (emblemas) para distinguir o comportamento dos participantes nos cursos e; (ix) Autenticação é baseadas no padrão *SAML2*, que é um formato padrão aberto baseado em XML para troca de dados de autenticação e autorização entre as partes.

A Figura 3 ilustra a plataforma do OpenMOOC que é composta de três componentes principais: (i) Provedor de identidade**,** responsável pela identificação dos usuários. Inclui o registro de usuários, dados de gerenciamento de usuários e *Single Sign-On* que permite acesso a múltiplos serviços após autenticar somente uma vez, baseado em *simpleSAMLphp* (provedor de identidade); (ii) Moocng, escrito em *Python/Django*, é o motor da plataforma, que permite ao professor criar, gerenciar e liberar os cursos e aos alunos a segui-los e; (iii) Askbot é um componente de perguntas e respostas similar ao Yahoo respostas, escrito em *Python/Django*. A plataforma OpenMOOC faz uso do *framework Django* que utiliza um modelo próprio dividido em três camadas, MTV *(Model, Template, View).* O *Template* em um módulo de

Administração e um Cliente LMS (*Learning Management System*) utilizado por alunos, tutores e professores. Esta camada é responsável pela apresentação do conteúdo usando modelos *HTML*. Os módulos Admin+LMS localizados na segunda camada contêm as *Views*, que executam as ações do sistema. A terceira camada *Model*, representa as entidades do banco de dados que são subdivididas em três módulos: Provedor de identidade, MOOCng e Askbot que terão acesso ao banco de dados, neste caso o *BD Mongo*.

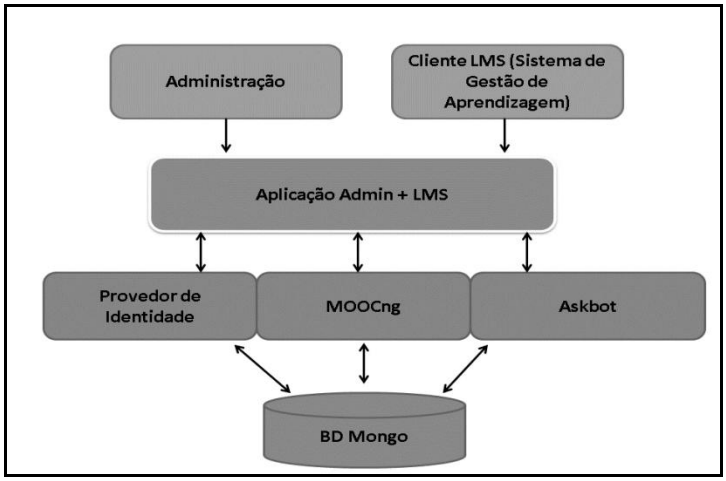


Figura 3 - Arquitetura da Plataforma OpenMOOC (adaptada de (OPENMOOC, 2013))

2.3.2 Edx

Edx é uma empresa sem fins lucrativos constituída por várias instituições que formam o *xConsortium*. Desde a sua fundação, em Abril 2012, a EDX tem se comprometido com uma visão de código aberto (EDX, 213). A sua plataforma Open EDX tem recebido contribuições de código de todo o mundo, entre elas da Universidade de Stanford, Google, MIT, da Universidade de Queensland, Tsinghua University, UC Berkeley, e da Universidade de Harvard.

A plataforma do Open EDX exige uma infraestrutura própria ilustrada na Figura 4:

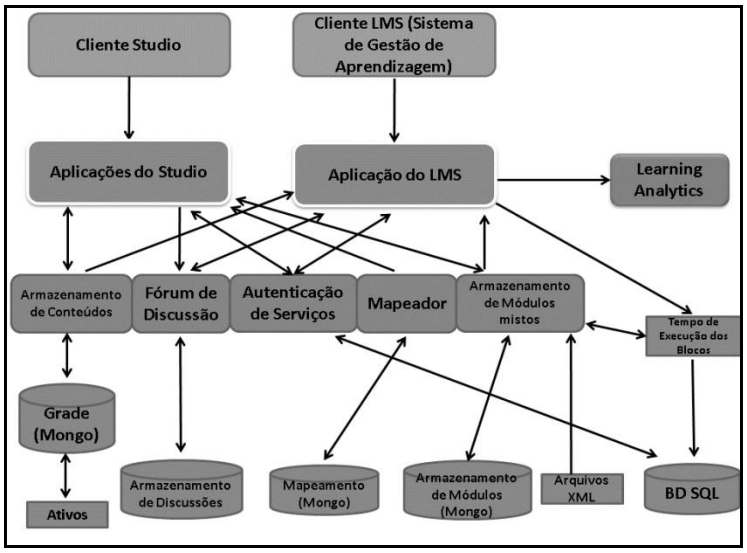


Figura 4 - Arquitetura de Plataforma Edx (adaptada de (EDX, 2013))

Open Edx é implementado principalmente em *Python*, com algumas partes em *Ruby* e *NodeJS*. Seu código está sendo disponibilizado sob uma licença *GPL*. O repositório principal é o *edx-plattform*, que inclui tanto os LMS e a ferramenta de autoria, Studio.

Como se observa, a arquitetura da plataforma *Open Edx* segue o modelo do *Framework Django* (*MTV*). A camada *Template* possui os módulos Cliente Studio (Administrador) e Cliente LMS (professor/tutor, aluno). Esta camada é responsável pela apresentação do conteúdo usando modelos *HTML*. Na camada das *Views* têm-se as ações tomadas pelo sistema. Ela contém dois módulos separados: Aplicação do Studio que são recursos para criação de curso e; Aplicações do LMS que os alunos usam para realizar as atividades do curso e as equipes do curso usam para administrá-lo. O LMS inclui o material didático, o painel do aluno e o painel do professor/tutor. Na camada *Views* tem-se também o módulo *Learning Analytics* que realiza a análise de dados de aprendizagem, decorrente do monitoramento das atividades realizadas pelo aluno. A terceira camada, *Model* representa as entidades no banco de dados. Tem-se o núcleo da aplicação que é dividido em cinco módulos: Armazenamento dos conteúdos, Fórum de discussões, Autenticação de serviços, Mapeamento e Armazenamento dos módulos dos cursos, componentes que terão acesso ao banco de dados.

2.3.3 Google Course Builder

O Google Course Builder (GCB) oferece uma plataforma de código aberto com ferramentas para criar cursos *online* (GOOGLE, 2013). É um projeto aberto do Google, ainda não é um

produto do Google. O GCB oferece uma infraestrutura para publicação de materiais para cursos *online*, acompanhamento e avaliação do desempenho dos alunos. Pode ser uma solução adequada para as instituições que procuram uma plataforma em nuvem. Como o código-fonte é aberto, pode ser modificado, as instituições podem desenvolver e personalizar suas próprias aplicações. Para que se possa manipular a plataforma, um administrador precisa ter habilidades em *HTML*, *JavaScript*, *Google App Engine* e *Python*.

Google App Engine (GAE) utiliza uma tecnologia no modelo plataforma como serviço, virtualizando aplicações em vários servidores, fornecendo *hardware*, conectividade, sistema operacional e serviços de *software* como ilustrado na Figura 5 (GOOGLE, 2013).

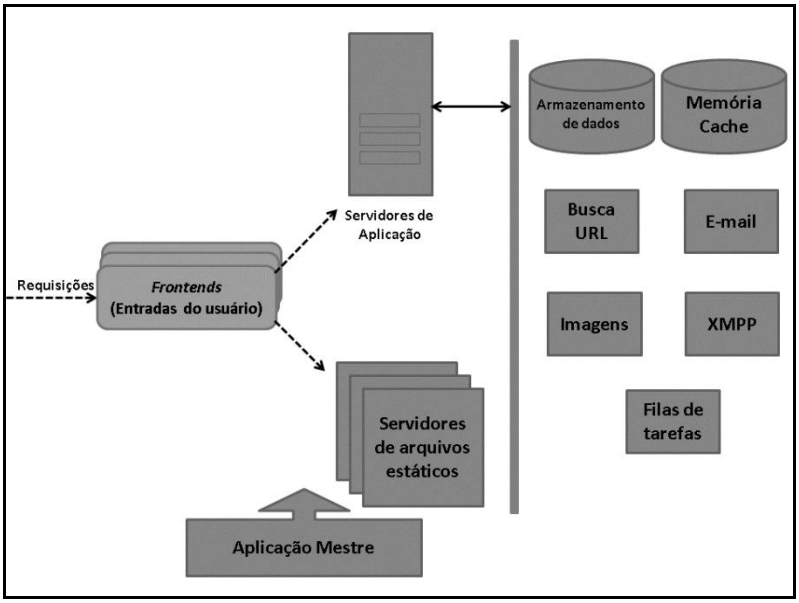


Figura 5 - Estrutura do GAE (GOOGLE, 2013)

A aplicação do GAE é baseada em uma arquitetura cliente/servidor de três camadas: apresentação, lógica de negócios e dados. A camada de apresentação é representada pelo navegador Web e utiliza *JavaScript*, onde o código é automaticamente interpretado rodando no navegador Web do cliente. A camada lógica de negócios inclui vários serviços que interagem com as API's dos prestadores de serviço externos, como Facebook, Youtube e outros. Essa camada interage com a camada de apresentação via solicitações *HTTP* e chamadas *RPC*, além disso, usa um parser XML para processar os dados recebidos das API's. Utiliza uma API chamada Memcache para armazenar os dados temporariamente e utiliza a API de armazenamento de dados para manter os dados. Já a camada de dados oferece a persistência dos dados.

O GCB é executado no GAE, ao contrário das demais plataformas OpenMOOC e EDX que necessitam de infraestrutura própria. Ela dispõe de uma plataforma de computação em nuvem conforme ilustrado na Figura 6. Uma característica interessante na versão mais recente do GCB é a possibilidade de adicionar componentes e módulos personalizados.

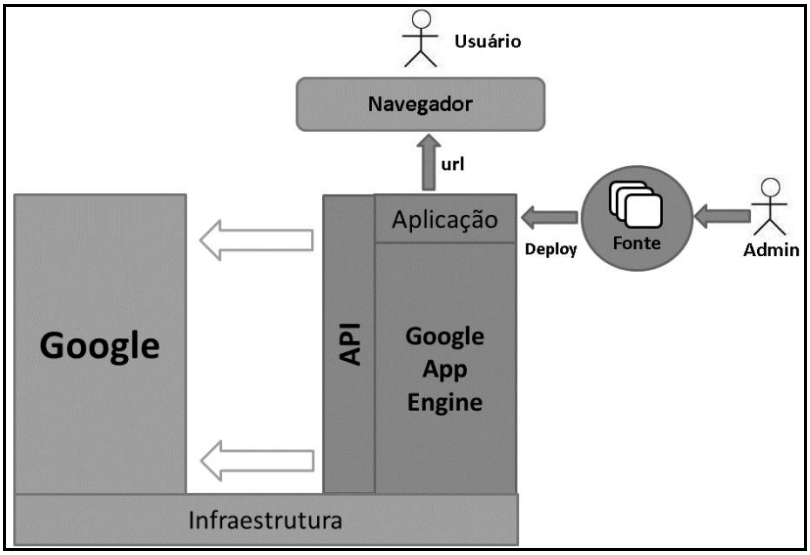


Figura 6 - Arquitetura da Plataforma GCB (adaptado de (GOOGLE, 2013))

Para iniciar um projeto utilizando o GAE utiliza-se o *Software Development Kit* (SDK) selecionando-se uma linguagem de programação dentre *Python, Java, PHP* e *Go,* para criação e utilização da aplicação do curso. Nesta aplicação encontram-se todos os módulos necessários para criação do MOOC e também usa-se a extensão *.yaml* que é aceita e utilizada pelos servidores do Google. Após a construção da aplicação na linguagem selecionada localmente, faz-se o *deploy,* tornando a aplicação disponível *online* utilizando a infraestrutura do Google. Peco e Lujan-Mora (2013) utilizaram a plataforma GCB, realizando alterações no código-fonte para seu domínio específico e obtiveram bons resultados nos quesitos estabilidade, flexibilidade e escalabilidade.

2.4 RECURSOS EDUCACIONAIS ABERTOS

Como já descrito, REAs podem ser definidos como materiais de ensino, aprendizagem e pesquisa em qualquer meio de armazenamento, que estão amplamente disponíveis por meio de uma licença aberta que permite reuso, readequação e redistribuição sem restrições ou com restrições limitadas (ATKINS et al., 2007; GIMENES et al., 2012). Em 2002 a UNESCO

cunhou esse termo em um fórum internacional (WILEY, 2002), do qual vários países participaram (GAZZOLA et al., 2014).

Apesar dos primeiros REAs terem sido publicados em formatos textos ou documentos baseados em um formato texto, isso não significa que os REAs possuem limitações sobre os tipos de mídias ou os tipos de arquivos a serem usados. Muitos REAs modernos são liberados em diferentes formatos, como em imagens, clipes de filmes animações, conjunto de dados e arquivos de áudios, dentre outros. Eles fornecem, portanto, material multimídia rico para uso e reutilização, o qual é disponibilizado por meio de grandes repositórios como Youtube (vídeos), Flickr (imagens) e iTunes (*podcasts*), sobre o regime de licenciamento *Creative Commons* (CC) (ABEYWARDENA e CHAN, 2013).

O conteúdo dos REAs também podem incluir cursos completos, partes de cursos, módulos, guias para estudantes, anotações, livros didáticos, artigos de pesquisa, instrumentos de avaliação, recursos interativos como simulações e jogos de interpretação, aplicativos, dentre outros recursos.

Os REAs têm um potencial para se tornar uma fonte importante de material didático e de pesquisa, especialmente para a educação superior, pois grandes organizações mundiais estão a favor de sua expansão em escala global, como a UNESCO, Comunidade de Aprendizagem (COL), Organização para a Cooperação e Desenvolvimento Econômico (OECD), e o Conselho Internacional de Educação a Distância (ICDE) (ABEYWARDENA e CHAN, 2013).

2.5 ANÁLISE DE DADOS DE APRENDIZAGEM

Análise de Dados de Aprendizagem, do inglês *Learning Analytics* (LA), é o processo de medição, coleta, análise e divulgação de dados sobre alunos e seus contextos para fins de compreensão e evolução da aprendizagem e dos ambientes em que ocorre (SIEMENS et al., 2011). A LA fornece aos envolvidos: alunos, professores e administradores, um ambiente de aprendizagem, informações que podem auxiliar na compreensão dos fatores que influenciam o processo de aprendizagem para o sucesso do aluno. A LA pode auxiliar na tomada de decisão sobre intervenções necessárias pelo professor ou pelos administradores, para que o aluno corrija falhas de aprendizagem. Pode-se apontar como objetivos da LA: melhorar as taxas de conclusão de curso, verificar grupos que estejam em risco de desistência, avaliar os alunos dentre outros. (SIEMENS et al., 2011).

Tipicamente, um processo de LA, ilustrado na Figura 7, começa com a fase de coleta de informação. Neste passo, as informações são geradas a partir das diversas atividades dos alunos quando interagem com os elementos de um Sistema de Gestão de Aprendizagem (LMS). Exemplos dessas atividades incluem escrever num fórum, ler um documento ou participar exercício colaborativo. Nesse passo é necessária atenção às questões de privacidade.

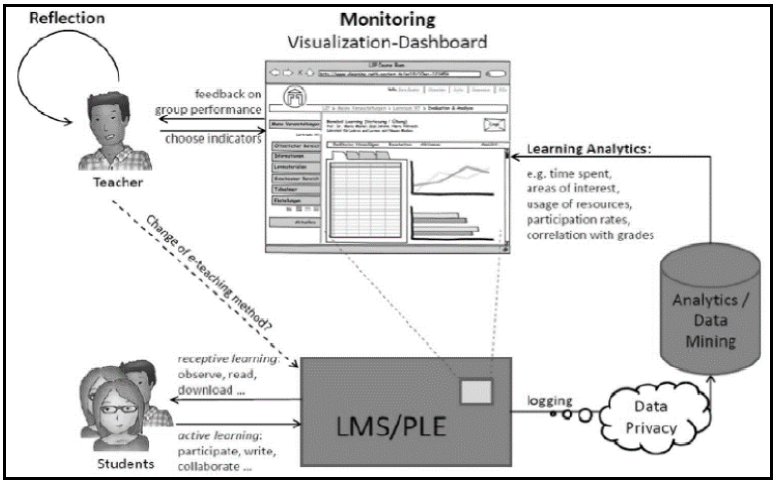


Figura 7 - O Processo da LA (DYCKHOFF et al. 2012)

Nesse sentido, é comum a saída da extração dos dados e a fase do pré-processamento serem transferidas para uma base de dados separada do ambiente. O segundo passo do processo da LA é a mineração da informação pré-processada por meio de várias técnicas (DYCKHOFF et al., 2012). Os resultados desse passo podem ser apresentados como um *widget*, que é um pequeno aplicativo com funcionalidades limitadas que pode ser instalado e executado numa página Web pelo utilizador final. *Widget* é um componente utilizado em computadores, celulares, *tablets* e outros aparelhos para simplificar o acesso a um outro programa ou sistema (EL HELOU et al., 2010). Geralmente contêm janelas, botões, barras de rolagem e também outras funcionalidades. Esse componente com interface gráfica (GUI - *Graphical User Interface*) permite adicionar tarefas específicas ao sistema operacional, como *widgets* de previsão de tempo, relógio, atualização de redes sociais, e-mails, etc. (EL HELOU et al. 2010).

Os *widgets* são classificados geralmente em dois tipos (EL HELOU et al. 2010):

- `Baixo nível`: são *widgets* utilizados para a criação do sistema operacional e fazem parte do núcleo do sistema;

- `Alto nível:` referem-se aos objetos finais. São facilmente encontrados em *toolkit* ou em *frameworks.*

O *widget* pode ser integrado em um LMS ou um *Dashboard*. Depois desses passos, os professores, por meio da visualização gráfica apropriada das informações, podem interpretar as informações, refletir sobre os seus métodos de ensino, sobre o comportamento e desempenho dos alunos e concluir sobre a eficácia de suas práticas (DYCKHOFF et al., 2012).

A maioria das atuais ferramentas de LA usa como base para a sua análise, ações concretas realizadas pelos alunos nos ambientes educacionais ou ferramentas de aprendizagem. Algumas ferramentas integram características dos alunos em suas análises, tais como, a informação acadêmica e dados pessoais. É importante notar que plataformas LMS e MOOCs dependem de dados *Web-analytics* adicionais como páginas visualizadas e tempo para sua análise.

De acordo com Del Blanco et al. (2013) os dados utilizados pelas ferramentas podem ser classificados em duas categorias:

- `Ações do Estudante, realizadas com um determinado resultado:` Por exemplo, um aluno ter acessado um recurso durante certo tempo; terminar uma atividade com um determinado resultado; ou concluir um *quiz* com uma percentagem de respostas corretas;
- *`Perfil do Aluno:`* idade, interesses, sexo, residência, formação, etc.

Com estas duas categorias de base, e meios adequados de agregação e sintetização, podem ser gerados dados mais complexos. Por exemplo, a partir de uma forma "aluno escreveu um post", é possível derivar o número total de mensagens escritas em um fórum por tal aluno, por um grupo de alunos, ou em um curso.

De uma perspectiva de LA, também poderia ser interessante saber sobre a história do aluno, resultados acadêmicos anteriores. Tal informação pode ser acessada por meio da coleta de dados da primeira categoria.

2.5.1 Análise de Dados de Aprendizagem para Sistema de Recomendação

As informações que são geradas por meio das interações realizadas pelos usuários de ambientes de aprendizagem são interpretadas e armazenadas na análise de dados de aprendizagem. O *framework* proposto por Verbert et al., (2012) utiliza informações

contextuais do usuário para recomendar itens pertinentes à situação de momento, conforme mostra a Figura 8. Esses itens são descritos a seguir.

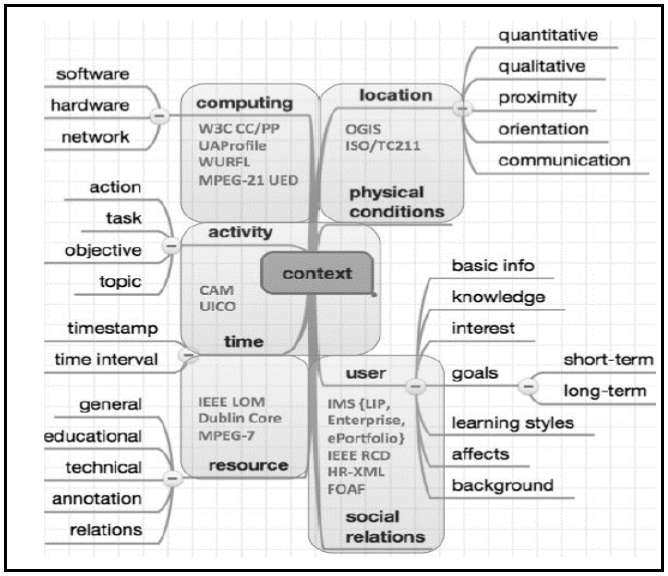


Figura 8 - *Framework* das categorias de contexto para LA (VERBERT et al., 2012)

Contexto Computacional (*Computing*) – A busca e uso dos contextos relacionados à computação são necessários para apoiar interfaces inteligentes que podem, por exemplo, selecionar recursos de aprendizagem adequados para o dispositivo que é usado.

Características do contexto computacional podem ser classificadas em três áreas:

- Rede: inclui propriedades estáticas e dinâmicas da rede, como a largura de banda máxima e disponível;
- Hardware: é composto por recursos de entrada e saída de equipamentos, recursos de armazenamento ou de CPU;
- Software: contexto descreve suporte a certas APIs, formatos de documentos, sistemas operacionais, protocolos de camada de aplicação.

Contexto de Localização (*Location*) – este contexto tem sido utilizado na pesquisa sobre computação móvel. Modelos de localização têm sido propostos para capturar informações geométricas de objetos, incluindo pessoas e dispositivos, e as relações entre os objetos. Exemplos são: (1) a proximidade de objetos dentro de um espaço, (2) capacidade de comunicação e, (3) a orientação, que pode, por exemplo, indicar em qual dispositivo de exibição do usuário está sendo feita a busca (BURNETT et al., 2001). Muitas vezes são referenciados por aplicações de aprendizagem, em sala de aula, em casa ao ar livre e em diversas variações sobre esses elementos. Algumas aplicações de aprendizagem usam a resolução mais precisa dentro dessas categorias por meio de um sensor de localização, como o GPS ou Wi-Fi.

Contexto de Atividades (*Activity*) – este contexto reflete as tarefas, objetivos ou ações do usuário. Um exemplo é o formato *Contextualized Attention Metadata* (CAM). O formato CAM foi desenvolvido para capturar focos de atenção dos usuários em diferentes aplicações. O esquema CAM é baseado em AttentionXML que é uma especificação aberta para capturar dados sobre como as pessoas usam a informação em navegadores, páginas Web, *feeds* de notícias e *blogs*. A AttentionXML descreve quais objetos de dados atraem a atenção dos usuários e que ações os usuários executam com esses objetos e contextos. Ao capturar essas informações, pode-se analisar os dados para fornecer mecanismos de auto-reflexão e de recomendação.

Wolpers et al. (2007) introduz o esquema CAM, ilustrado na Figura 9 para capturar informações de comportamento de usuários dentro de diferentes aplicações. Os principais elementos do esquema são: o elemento grupo (*group*) que integra a atenção do usuário e identifica-o em todos os sistemas; o elemento de alimentação (*feed*) integra a atenção do usuário em um sistema específico; o elemento item (*item*) que coleta a atenção dada a um documento digital específico e; o elemento evento (*event*) que representa os eventos que ocorrem em documentos (por exemplo, ler, atualizar, fazer *download*).

A arquitetura CAM permite a coleta de metadados a partir de qualquer *desktop* ou aplicativo do lado do cliente, juntando-os em um único fluxo por usuário, e armazenando dados de CAM.

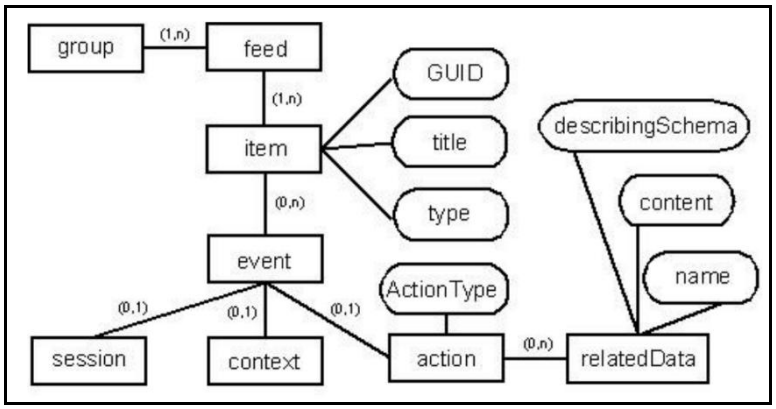


Figura 9 - Os principais elementos do esquema CAM (WOLPERS et al., 2007)

Contexto de Tempo (*Time*) – Contexto de tempo inclui informações de data e hora, ou menos específicas como, informações sobre a semana, mês ou semestre do ano. Tempo é muitas vezes utilizado em conjunto com outras peças de contexto, como um *timestamp* (momento em que o evento ocorreu) ou como intervalo de tempo, o que indica um instante ou período durante o qual outras informações contextuais são conhecidas.

Contexto de Recursos (*Resource*) – Captura características relevantes de recursos físicos ou virtuais. O IEEE Metadados de Objetos de Aprendizagem (LOM) (IEEE 1484.12.2, 2002) é um padrão para a descrição de recursos de aprendizagem. Os elementos são organizados em várias categorias, incluindo uma descrição geral do recurso, os requisitos técnicos, as características educacionais, as relações entre os recursos e os comentários de anotação sobre o uso educacional dos recursos. Outras normas que são frequentemente utilizadas para a descrição de recursos de aprendizagem são o *Dublin Core Metadata Element Set* (WEIBEL et al., 1998) e MPEG-7 (MARTINEZ, 2003).

Contexto de Usuário (*User*)– Modelos foram pesquisados extensivamente na hipermídia educacional adaptativa, áreas de pesquisa de modelagem. Nesse contexto tem-se: (BRUSILOVSKY e MILLÁN, 2007) (SPECHT, 2000) (NGUYEN e DO, 2008).

- Informações Pessoais Básicas: normalmente inclui informações de identificação, nome, informações de contato, afiliações, informações de autenticação, informações sobre acessibilidade, incluindo capacidades de linguagem e deficiência, além de outras características pessoais, como sexo, idade, profissão e nível de escolaridade;
- Conhecimento: representa os níveis de conhecimento prévio do aluno;
- Interesses: interesses de captura ou preferências dos alunos e são as principais características para apoiar a personalização. Valores que são normalmente armazenados incluem termos de pesquisa do usuário, suas *tags*, comentários e os recursos, que o usuário criou, leu ou classificou;
- Metas de aprendizagem: A distinção é feita frequentemente entre objetivos de curto prazo, onde o aluno tem a intenção de resolver um determinado problema e metas de longo prazo, que estão relacionadas com um curso ou planos para a aprendizagem ao longo do tempo;
- Aprendizagem e estilos cognitivos: Os alunos diferem em sua maneira preferida de aprender. Exemplos para considerar diferentes estilos cognitivos são: visual, textual ou apresentação auditiva de informações. Diferentes estilos de aprendizagem incluem a apresentação de exemplos, a apresentação de conhecimentos teóricos e exercícios práticos;
- Afetiva: A modelagem e uso da informação afetiva é um tópico de pesquisa popular em diversas áreas de pesquisa;

- *Background*: O *background* do usuário está relacionado com as suas experiências anteriores fora do domínio de um sistema específico. Elementos tipicamente modelados incluem a experiência de trabalho em áreas afins, religião e características culturais.

Contexto de Condições Físicas (*Physical Conditions*) – Descreve as condições ambientais em que o sistema ou o usuário está situado, geralmente inclui medidas para o calor, luz e som. Em cenários de aprendizagem, iluminação e ruído, por vezes são critérios importantes a serem considerados;

Contexto de Relações Sociais (*Social Relations*) – Descreve relações sociais, conexões ou associações entre duas ou mais pessoas. As relações sociais podem conter informações sobre amigos, inimigos, vizinhos, colegas de trabalho e parentes (VERBERT et al., 2012).

2.6 CAM - METADADOS DE ATENÇÃO CONTEXTUALIZADA

Como já descrito, CAM permite monitorar as interações do usuário com os ambientes de aprendizagem. O esquema CAM permite modelar a interação de um usuário com os conteúdos digitais. Como CAM foi desenvolvido para descrever muitos tipos de metadados de atenção, os registros CAM de um usuário não descrevem simplesmente os focos de atenção dos usuários, mas sim todo o seu comportamento no uso do computador. CAM pode ser analisado para fornecer uma visão geral sobre onde (com qual aplicativo) e quando uma ação acontece e o que acontece no ambiente.

Análises de CAM permitem a descoberta de popularidade, uso e tendências de ferramentas, também descobrir padrões, como a impopularidade de aplicações. Informações sobre quando uma ação acontece pode ser útil em ambientes controlados, tais como ambientes de aprendizagem formais, onde as atividades são normalmente programadas. Também é bastante útil em ambientes menos controlados ou mistos, para entender quando os alunos são realmente ativos (WOLPERS et al., 2007).

2.6.1 Captação de CAM

Os dados são obtidos por meio da coleta e formalização de observações sobre como e no que o usuário gasta sua atenção e em qual contexto. Em geral, não existe uma definição consensual de atenção. No entanto, Roda e Thomas (2006, p. 6) apontam que, “a maioria dos

pesquisadores referem-se à atenção como o conjunto de processos de habilitação e orientação para a seleção de entrada de informação perceptual". E essa interpretação aplica-se ao manuseio diário de conteúdo digital. Para avaliar o padrão CAM, o computador do usuário é utilizado como principal fonte de observações. Quando há referência a conteúdo digital, o termo "documento" é utilizado. Um documento pode ser uma página Web, um arquivo de texto, imagens de um JPEG, um e-mail, uma sessão de bate-papo, um arquivo de música MP3, dentre outros.

O usuário lida com os documentos de uma forma particular que é específica apenas para ele. Ao capturar e generalizar seu comportamento, padrões emergentes de seu comportamento podem ser usados para descrevê-lo. Por exemplo, tais padrões podem ser derivados a partir de dados observacionais de como o usuário recebe os documentos, por exemplo, via e-mail, *chat* ou se ele encontrou-os por meio de pesquisa na Web.

As observações podem ser generalizadas em padrões comportamentais. Padrões comportamentais descrevem, em geral, como um usuário lida com informação. A comparação dos padrões de comportamento de vários usuários permite o agrupamento de usuários semelhantes. Assim, pode-se prever com precisão futuras etapas e metas do usuário, podendo até utilizar mecanismos de recomendação entre usuários semelhantes (WOLPERS et al. 2007).

Os aplicativos precisam ser desenvolvidos para capturarem as observações da atenção do usuário. Esses aplicativos precisam integrar o ambiente de trabalho diário do mesmo, sem perturbá-lo ou interrompe-lo. Além disso, os aplicativos precisam capturar as observações de aplicações existentes de uso diário, por exemplo, o *Microsoft Office*, Navegadores, Clientes de E-mail ou WINAMP *Music Player*. Assim, fornecem continuamente observações que geram um fluxo de informações sobre as interações do usuário com suas respectivas aplicações.

As observações de usuários precisam ser capturadas em um formato generalizado que permite fusão e transformação dos vários fluxos de observações. Ao fundir as observações, é possível contextualizar cada observação e identificar quais atividades o usuário efetuou simultaneamente em um curto período de tempo. Por exemplo, um contexto que pode ser identificado é a música que o usuário ouvia enquanto escreveu um e-mail, com qual conteúdo e para quem.

Outro exemplo é com que palavras-chave o usuário encontrou documentos relevantes que realmente queria. Por isso, fala-se de observações contextualizadas da atenção do usuário (WOLPERS et al., 2007).

Há várias abordagens para capturar observações sobre a atenção do usuário, por exemplo, no âmbito dos projetos europeus Nepomuk (2006), Aposdle (LINDSTAEDT, 2006), Gnowsis (SAUERMANN, 2005), além de outros como, (VÖLKEL e HALLER, 2006), (HOLZ et al., 2006), (BRAUN e SCHMIDT, 2006), (GAROFALAKIS et al., 2006), (BROISIN et al., 2005), (RODA e THOMAS, 2006). Nessas abordagens o foco recai mais sobre a classificação das observações de acordo com uma ontologia pré-definida. A ontologia pré-definida geralmente descreve um conjunto específico de atividades que está relacionado com algumas tarefas ou processos.

Outras abordagens se concentram na coleta e observações sobre o usuário, monitorando os cursos chaves, movimentos do mouse, número de cliques, etc. Essas abordagens fornecem uma grande quantidade de dados altamente refinados. Esses dados de granulação fina são muito valiosos para analisar a forma como um usuário usou um site (WEINREICH et al., 2006) ou um sistema de informação.

Wolpers et al., (2007) descrevem um esquema de CAM para capturar as observações. Baseia-se em AttentionXML (NAJJAR et al., 2005), que é uma especificação de estudo para capturar dados sobre como as pessoas usam a informação em navegadores, páginas Web, *feeds* de notícias e *blogs*. Esses dados são então analisados e fornecem informações estatísticas sobre seus interesses e atividades ao longo do tempo. Além disso, os interessados utilizam os dados para prever tendências, recomendar itens, dentre outros.

AttentionXML é um esquema baseado em XML que agrupa as observações capturadas de acordo com os usuários. Cada usuário tem uma ou mais fontes de alimentações *(feeds)*. Um *feed* pode ser um fluxo, cliques dentro de um navegador, *feeds* de notícias, etc. Além disso, alguns dados específicos do usuário são capturados como, a data e a hora em que chamou a atenção do usuário e quando o usuário descartou alguma informação. Cada *feed* consiste em um ou mais itens que são descritos por meio de uma série de propriedades, como, o identificador global (*guid*), o título (*title*), quando foi a última leitura, quando foi atualizada, duração, o seu tipo (*type*) *MIME*.

AttentionXML é direcionado para capturar observações dos usuários relacionadas a comportamento de navegação e informações de consultas em blogs e *feeds* de notícias. Não permite a captura de observações relacionadas com atividades como pesquisar e baixar arquivos na Web, ler, escrever e editar documentos, ouvir música, mensagens de comunicação (por exemplo, *chats*), etc. Além disso, não permite captar o contexto dentro do qual uma atividade ocorreu.

Wolpers et al., (2007) propõem uma extensão para AttentionXML que permite ampliar significativamente os tipos de observações que podem ser capturadas e descritas no contexto onde ocorreram. Essa extensão permite capturar o tipo de evento (incluindo todas as propriedades relevantes) que ocorre com cada item. O tipo de evento permite classificar o evento que pertence a cada item. Portanto, quebra-se o pressuposto implícito de AttentionXML, de que cada item é um *feed* de notícias, uma *wiki* ou *blog*. Essa abordagem permite capturar observações sobre atividades do usuário em qualquer tipo de ferramenta, não apenas de um *browser* ou leitor de notícias como no AttentionXML.

2.6.2 Coleta de Fluxos de CAM

Coletam-se observações no formato CAM a partir de todos os aplicativos disponíveis do computador do usuário. Identificam-se as fontes ao longo das dimensões: `localização` do aplicativo de geração de observação e seu `tipo social`. A `localização` descreve onde os dados CAM são gerados, seja na área de trabalho do usuário ou no servidor. O `tipo social` descreve como as observações se relacionam com o usuário sozinho ou em suas interações sociais com outros usuários. Por exemplo, a aplicação *Microsoft Office* ou um navegador foca somente em um usuário, enquanto um cliente de e-mail e *chat* fornece dados CAM que descrevem suas relações sociais com outros usuários. Esta classificação é usada para identificar os vários tipos possíveis de correlações de fluxos de dados observacionais (WOLPERS et al., 2007).

O foco principal da coleta está em aplicativos que são usados diariamente pelo usuário, como *Microsoft Office*, Navegadores. Extensões dos navegadores da Web, como *Slogger* (BAKER e BOAKES, 2008) já permitem a captação do comportamento da navegação do usuário em ambiente Web (VERBERT, et al., 2005). Como é visualizado na Figura 10, *framework* CAM.

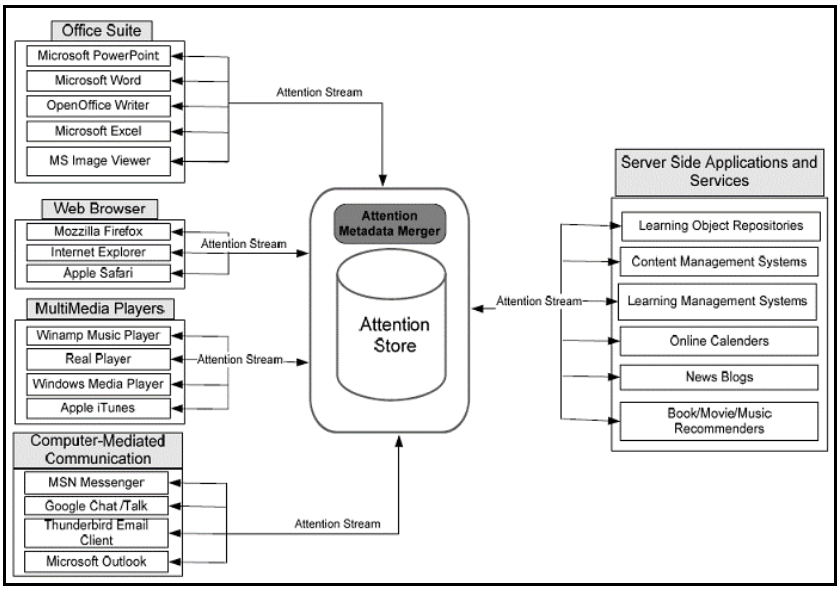


Figura 10 - O *Framework* CAM (WOLPERS et al., 2007)

O *framework* CAM trabalha para coletar, armazenar e fundir os vários fluxos de observações em formato de CAM. Cada aplicativo de *desktop* (lado esquerdo da Figura) gera um fluxo de observações que captura as atividades dentro do aplicativo com *timestamp* (marcação temporal denotando a hora ou data em que certo evento ocorreu) e dados relacionados com o conteúdo. Esses fluxos relacionados com a aplicação são recolhidos e fundidos em um único fluxo por usuário. O formato CAM permite prestar atenção nos fluxos dos metadados em um banco de dados por usuário. A fusão é feita com base na categoria do usuário CAM. Em seguida, os fluxos são armazenados em um CAM *store*.

2.6.3 O Esquema de CAM

A Figura 11 ilustra o esquema CAM desenvolvido para permitir o rastreamento das atividades do usuário em diferentes sistemas.

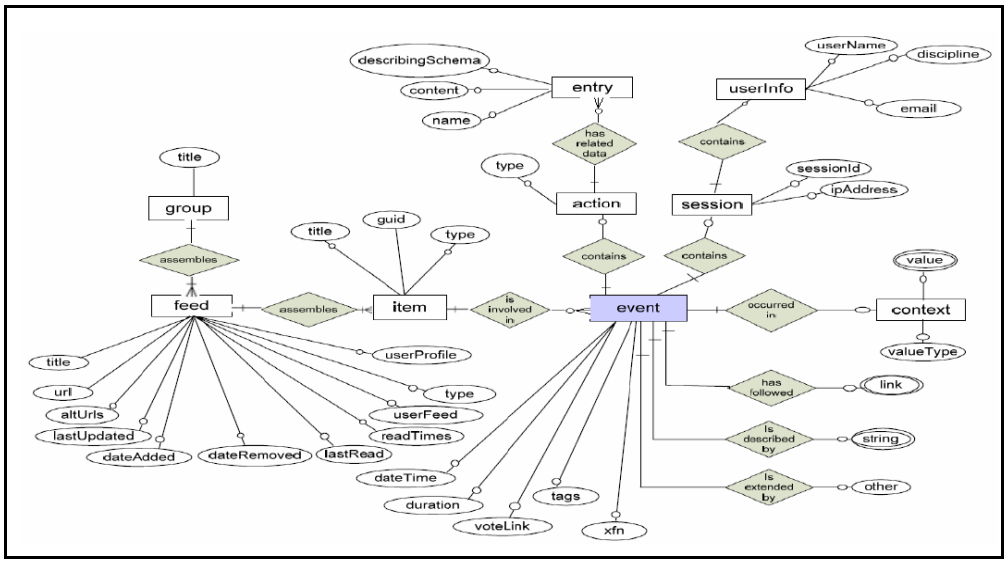


Figura 11 - Os elementos do esquema CAM (WOLPERS et al., 2007)

O esquema CAM é projetado para permitir que as atividades do usuário em todos os sistemas sejam rastreadas, conforme as políticas de privacidade. Como mostrado na Figura 11, CAM recolhe a atenção do utilizador em todos os sistemas de dentro de um elemento de grupo (*group*). O propósito do elemento alimentação (*feed)* é agrupar a atenção do usuário em um sistema específico. O elemento *item* coleta a atenção dada a um documento digital específico, o que acontece é que cada documento digital pode ser acessado em diferentes ocasiões e estar envolvido em diferentes tarefas (leitura, edição, atualização, audição, etc), que exige capturar informações relacionadas a cada evento em que o documento estava envolvido.

Enquanto isso não era possível em AttentionXML, uma extensão do modelo AttentionXML no nível de elemento *item* foi proposta, adicionando o elemento de eventos (*event).* Cada item pode estar envolvido em um ou mais eventos que tenha diferentes informações pertinentes. Por exemplo, o documento pode ser editado em um sistema e depois, em outro evento, o mesmo documento pode ser lido ou atualizado. Cada vez que um documento é acessado, mais metadados de atenção são coletados, como data e hora de acesso, contexto, tempo gasto no trabalho com o documento e dados ativamente criados pelo usuário (*ranking*, anotações e *tags*).

Um professor pode utilizar o mesmo documento em seus cursos *online* para dois grupos diferentes de alunos e em dois contextos diferentes (tempo, aplicação e tópico). Em cada curso, onde o documento é utilizado, diferentes informações de metadados podem ser coletadas, como o tempo gasto em aprendê-lo, o tema (computação, gestão, etc) do curso e a avaliação dos professores sobre a utilidade do documento para cada grupo de estudantes.

Como também é mostrado na Figura 11, no esquema CAM, os elementos, *duration*, *voteLink, xfn* e *tags* são movidos a partir do elemento *item* no elemento *event*. Essa reordenação permite a identificação do tempo que permaneceu com o documento, as marcações especificadas em documento realizadas pelo usuário, relacionamento e experiência sociais do usuário (*voteLink;* informações como simpatia/antipatia) por evento.

No elemento *group*, agrupam-se todos os metadados de atenção de um usuário em todas as aplicações que ele pode trabalhar (tudo em um grupo de instâncias), enquanto o elemento *feed* agrupa a atenção de um usuário em uma ferramenta específica (cada ferramenta em instância de alimentação separada). O grupo de elementos e alimentação do esquema CAM são as mesmas como no AttentionXML (WOLPERS et al., 2007).

Wolpers et al., (2007) descrevem brevemente sobre os elementos do esquema:

`Item:` o elemento *item* agrupa metadados atenção de um documento. Cada documento pode ser envolvido em diferentes ações (ler, ouvir, editar, etc), em datas diferentes, por diferentes períodos e em diferentes contextos.

O elemento *item* tem três sub-elementos que não mudam ao longo das diferentes ações e eventos que documento pode estar envolvido; esses elementos gravam as propriedades do próprio documento. Os dados são coletados para reconhecer e identificar o documento em diferentes sistemas e contextos:

- `Título:` o *title* captura um nome legível dado ao documento, quando o documento é criado ou editado. Este elemento é necessário para permitir que os usuários reconheçam facilmente o documento. Por exemplo, "O aquecimento global - *Wikipedia*, a enciclopédia livre" é um título de um documento sobre o aquecimento global na *Wikipedia*;
- `Guid:` o elemento *guid* representa um identificador global exclusivo para o documento dentro de um determinado contexto. Por exemplo, http://en.wikipedia.org/wiki/Global_warming é o identificador único, que é usado para localizar o documento com o título mencionado em *title* na internet;
- `Tipo:` o *type* de elemento tem o tipo de técnica *MIME* (*Multipurpose Internet Mail Extensions*) do documento. Por exemplo, "html" é o tipo *MIME* correto do documento acima sobre o aquecimento global.

Por outro lado, todos os dados sobre os diferentes eventos do documento podem ser envolvidos e estarem agrupados no elemento evento contendo a atenção dada ao documento, assim como, o tempo gasto com ele, as marcações anexadas ao documento depois de lê-lo e

do contexto em que foi utilizado. Cada dado relacionado a eventos é agrupado em uma instância de evento.

- `Evento:` o elemento *evento* agrupa metadados de atenção para cada evento em que o documento esteve envolvido. Para cada instância de evento os seguintes metadados de atenção são coletados:
  - `Ação:` o elemento *action* fornece informações sobre a ação em que o documento estava envolvido (por exemplo, se foi inserido no sistema local de arquivo ou repositório digital, aberto em uma aplicação de visualização, etc):
    - `Tipo de ação:` o elemento *actionType* detém o tipo de ação (tarefa) em que o documento foi envolvido. Seu valor é normalmente um valor de referência para um valor no espaço de ação. Por exemplo, o URL http: //.../Actiontype/insert pode constituir uma referência para uma ação inserção de valores, se foi a inserção de um documento em um repositório;
    - `Entrada:` o elemento *entry* grava dados dos registros de entrada relacionados a uma ação realizada. Sobre a mesma ação, uma ou mais entradas podem ser gravadas. Por exemplo, se o documento foi encontrado usando uma consulta, uma instância do elemento *entry* pode armazenar os termos da consulta utilizados pelo usuário. Outra entrada pode armazenar a lista de resultados da consulta. Em caso de inserção, também registra o nome do esquema de metadados (por exemplo, IEEE LOM ou *Dublin Core*) utilizado para indexar o documento. Se a ação foi conversa com outra pessoa (bate-papo), esse elemento pode armazenar o nome do parceiro de *chat* em uma instância de entrada e o texto da conversa em outra instância de entrada;
  - `Data Hora:` o elemento *dateTime* mantém a data e a hora em que o evento ocorreu, este elemento mantém todas as marcas de tempo de eventos em que o documento esteve envolvido;
  - `Duração:` o elemento *duration* registra o tempo gasto com o documento (em segundos);
  - `Sessão:` o elemento da sessão contém as informações que são necessárias para identificar a sessão de trabalho:
    - `ID da Sessão:` o elemento *sessionId* tem um identificador único para a sessão;

- `Endereço IP:` o elemento *ipAddress* contém o endereço IP do computador do usuário;
- `Informações do usuário:` o elemento *userInfo* coleta informações sobre o nome do usuário, endereço de e-mail e disciplina científica do usuário que executa a ação.

As informações da sessão *(sessionId* e *ipAddress)* são utilizadas para identificar o usuário, ao longo dos diferentes eventos e tarefas que podem interagir com o documento. Os dados sobre o usuário são coletados por evento, pois o mesmo usuário pode ter, por exemplo, nome de usuário diferente, o endereço IP diferente quando trabalha com o mesmo documento. Trabalhando com um documento a partir do computador no trabalho ou em casa, pode resultar em diferentes endereços IP para o mesmo usuário.

- `Contexto:` o elemento *context* captura as informações que descrevem os ambientes em que o usuário pode interagir. Por exemplo, informações sobre um curso (disciplina e descrição) em que o usuário submeteu um documento. O título e a descrição de um curso sobre Interação Humano-Computador no *Blackboard* ou *Moodle* são sistemas de informação contextual sobre o uso de um documento. Os dados capturados podem ser extraídos a partir das propriedades dos cursos onde os documentos são utilizados; cada curso no *Moodle,* por exemplo, tem um título e uma descrição, esses dados podem ser usados para extrair a informação sobre o contexto. Essa informação é essencial para identificar os diferentes contextos em que o conteúdo digital é usado. O elemento contexto tem dois sub-elementos:
  - `Tipo de valor:` o elemento *valueType* contém referência a um elemento de uma ontologia ou taxonomia que descreve a disciplina que pode ser extraída a partir do elemento de valor. Os tópicos podem servir como termos de pesquisa para identificar a disciplina apropriada com serviços *online* como *Swoogle* (DING et al., 2004);
  - `Valor:` o elemento *value* tem um texto livre que é extraído, por exemplo, a partir do título do curso em aplicações como *Moodle* ou *Blackboard.* Ele descreve os temas de outros documentos envolvidos no trabalho, por exemplo, os temas de todos os documentos envolvidos em um curso. Esse sub-elemento tem valores de entrada multipalavras. Essas entradas em multipalavras podem ser usadas para expressar o mesmo valor em mais de um idioma.

Além da informação obtida nos dois sub-elementos acima, mais informação contextual pode ser extraída a partir de outros elementos. Por exemplo, *event.dateTime*,

event.action.actionType e *event.session.userInfo.discipline* são ricas fontes de informações contextuais. Esses dados permitem identificar padrões interessantes sobre a atenção do usuário dada aos documentos. Por exemplo, usando o elemento *event.action.actionType* sabe-se se o usuário está navegando por uma página da Web, trabalhando com *slides* do *Microsoft Power Point* ou ouvindo música, ou pode fazer tudo ao mesmo tempo. Esses dados podem, por exemplo, permitir identificar as músicas que o usuário ouve quando trabalha com o *Microsoft Power Point*. Usando o elemento *event.dateTime* é possível identificar a música que um usuário ouve pela manhã, até a música que ouve à noite. Além disso, é possível identificar as páginas da Web que um usuário consulta ao trabalhar com *slides* do *Microsoft PowerPoint*. Outros sub-elementos são:

- `Links seguidos:` o elemento *followedLinks* agrupa um conjunto de URLs incluídas no documento seguidos pelo usuário. Isso pode ser um *link* para uma página Web relevante de um documento que está sendo lido;
- `XFN:` o elemento *xfn* acompanha o relacionamento social do autor do documento até a consulta do leitor do documento, se o valor do elemento *event.action.actionType* é lido em um aplicativo de navegador. Em uma ferramenta de *chat*, por exemplo, também pode gravar a relação do usuário com outras pessoas envolvidas em um evento de bate-papo;
- `Vote Link:` o elemento *voteLink* registra o interesse do usuário (gostos e desgostos). O elemento pode ter um dos seguintes valores:
  - `vote-for:` significa “eu gosto do documento”;
  - `vote-abstein:` significa “eu tenho uma opinião neutra”;
  - `vote-against:` significa “eu não gosto do documento”.

Os dados podem ser utilizados, por exemplo, para recomendar um utilizador com os documentos que são semelhantes (de autor semelhante, por exemplo) para os documentos que um utilizador considere positivo. Os documentos que são semelhantes aos documentos voto-contra podem ser omitidos para o usuário. Mais interessantes ainda, esses dados podem ser usados para classificar o documento com base nos votos de um conjunto de usuários. Se muitos usuários votaram em um documento específico, isso significa que o documento é interessante. Outros sub-elementos são:

- `Etiquetas:` o elemento *tags* detém um marcador de texto livre ou palavras-chave que são usadas para descrever o documento. Por exemplo, os metadados de atenção, os dados do usuário e o acompanhamento do usuário, são etiquetas válidas para este elemento;

• `Descrição:` o elemento *description* cobre anotações descritivas que possam ser fornecidas por usuários para expressar a sua experiência com o documento. O usuário usa o espaço de valor do IEEE LOM "*Description*", elemento que é um valor de multipalavras, para fornecer a mesma informação em diferentes idiomas. Este elemento é útil para coletar comentários ou dados descritivos sobre a experiência do usuário no item de leitura;

• `Outros:` O elemento *other* permite fornecer elementos personalizados que não são abrangidos pelo presente esquema.

Um exemplo pode ser observado na Figura 12 do *framework* usado para coletar, transformar e gerenciar os fluxos CAM.

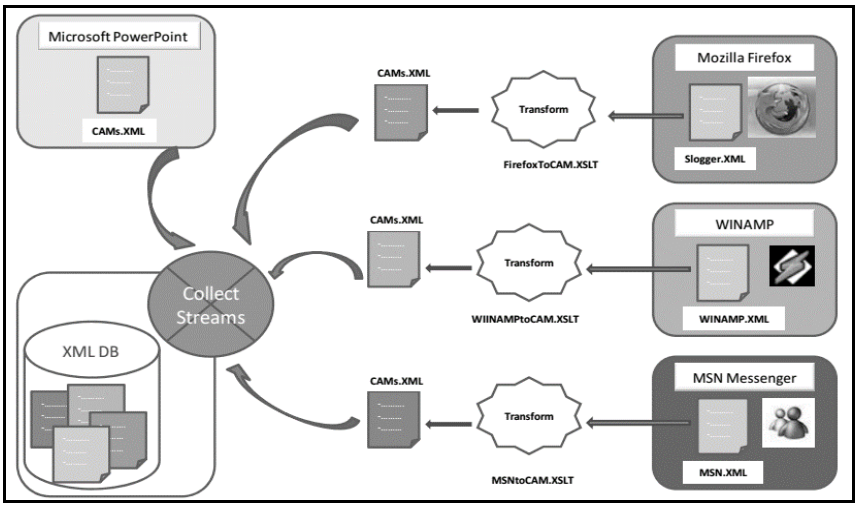


Figura 12 - *Framework* usado para coletar, transformar e gerenciar os fluxos CAM (WOLPERS et al., 2007)

2.6.4 Geração de CAM

O XML detalhado de parte de uma instância CAM de um usuário, utilizando o *Microsoft Power Point* com o *plug-in* ALOCOM para capturar os metadados de atenção, ilustrado na Figura 13.

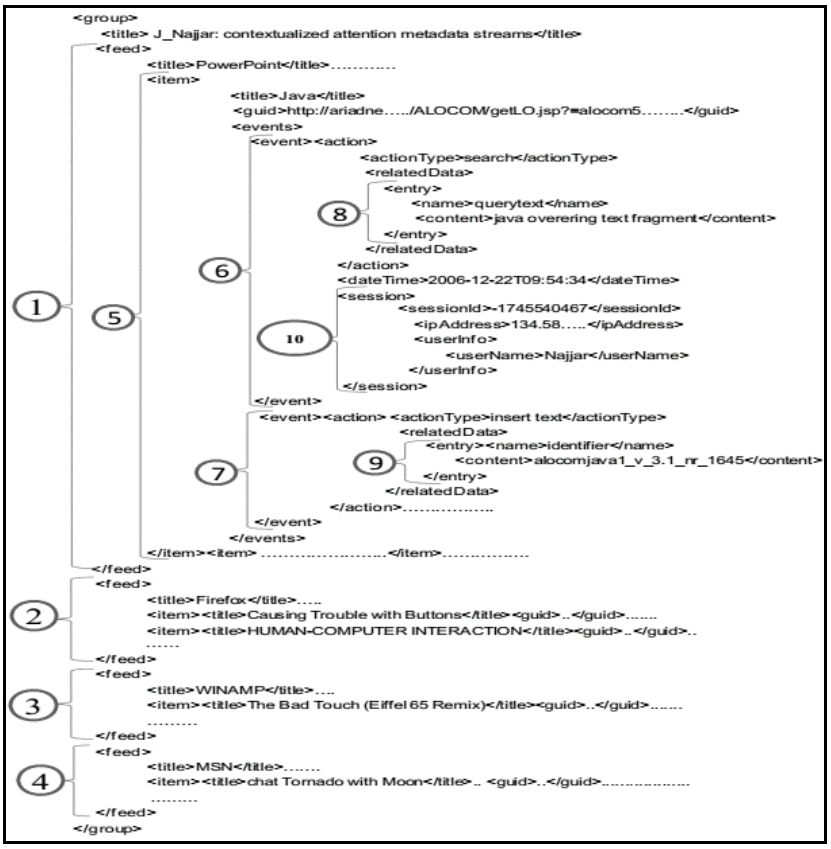


Figura 13 - Parte de uma instância CAM de um usuário (WOLPERS et al., 2007)

Todas as informações de atenção no *Microsoft Power Point* são agrupadas em uma instância de *feed*, (visto no item 1 da Figura 13). Dados sobre cada documento (acessados na apresentação de *slides*) são agrupados em uma instância *item* (visto no item 5 da Figura 13). Para cada documento acessado, os seguintes elementos CAM são capturados dentre outros:

- *group.feed.title:* o título da apresentação. Uma API do *Microsoft Power Point* (Microsoft.Office.Interop.Powerpoint assembly) é usada para recuperar esta propriedade;
- *group.feed.item.guid:* um identificador que é gerado pelo *framework* ALOCOM;
- *group.feed.item.type:* o tipo *MIME* do documento;
- *group.feed.item.lastUpdate:* controle do tempo da ultima vez que salvou;
- *group.feed.item.event.duration:* o período total de tempo de edição em segundos – *"total editing time".*

Nos eventos números 6 e 7 da Figura 13, os metadados de atenção na interação do usuário são armazenados em uma instância separada por evento. No primeiro evento (6 na Figura 13), a ação "*search*" armazenada dados no elemento *actionType.* Os dados relacionados com esta ação "*search"* são armazenados nas instâncias do elemento *entry* (ver 8 na Figura 13). Contudo, essa entrada possui dois sub-elementos; o elemento *name,* que

armazena o tipo de conteúdo em "*querytext*". O elemento *content* armazena o conteúdo real dos dados relacionados com a ação de busca, termo de pesquisa usado para procurar o documento.

No segundo evento (ver 7 na Figura 13), o elemento *action* armazena dados sobre a atividade do usuário de inserir um fragmento de texto no *slide*. Esse fragmento de texto é encontrado usando o *plug-in* ALOCOM. Para esta ação o identificador do documento é armazenado em um elemento *entry* do elemento *action.relatedData* (ver 9 na Figura 13).

- *group.feed.item.evento.session* (ver 10 na Figura 13).
    - *sessionIdentifier:* código para o momento que aconteceu o evento;
    - *ipAddress:* a API System.Net é usada para recuperar esta propriedade;
    - *userName:* a API System.Environment é usada para recuperar esta propriedade;
- *group.feed.item.event.dateTime:* momento que aconteceu o evento. Esse momento é usado para identificar os diferentes eventos que o documento possa estar envolvido.

Os dados gravados são dados do próprio documento (por exemplo, *title, MIME, type, guid)* ou sobre o usuário (*sessionId, ipAddress* e *userName*). A razão para a captura dos dados é dupla: ser capaz de reconhecer os documentos e também os usuários os manusearem. Além disso, para identificar e relacionar (usando o *dateTime* a informações de *session)* os eventos em que o usuário possa ter trabalhado com cada documento.

Também presente na Figura 13, estão a coleção de metadados de atenção do *Browser Firefox* (ver 2 na Figura 13), *Winamp Music Application* (ver 3 na Figura 13) e *MSN Messenger* (ver 4 na Figura 13).

2.6.5 Gerenciamento e uso de CAM

Os fluxos de atenção nas quatro aplicações citadas anteriormente são fundidos em um fluxo XML usando o elemento *group.* Os metadados de atenção para cada ferramenta são representados em uma instância *feed.* O conjunto CAM XML que representa os metadados de atenção dos usuários é armazenado em banco de dados Exist. Declarações XQuery são usadas para identificar e relacionar os diferentes fluxos de atenção e agrupá-los em um documento que é ilustrado a Figura 14:

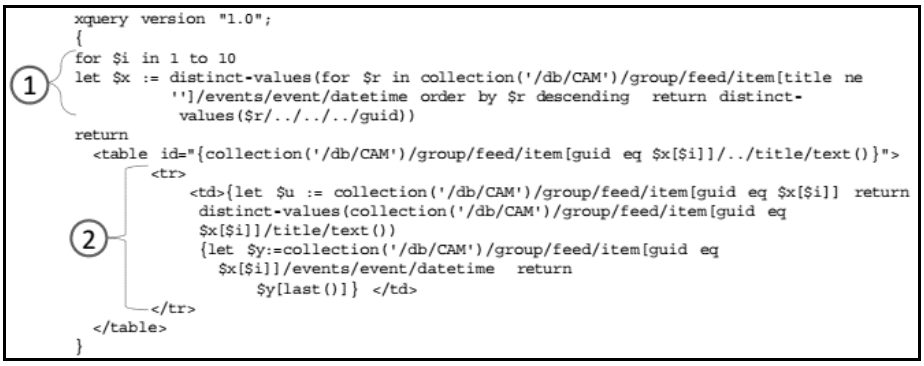


Figura 14 - Um *script* XQuery para recuperar os últimos 10 documentos que os usuários trabalharam(WOLPERS et al., 2007)

O *script* identifica os últimos 10 documentos que os usuários trabalharam em todas as aplicações (usando o elemento *event.dateTime,* ver 1 na Figura 14). Em seguida, lista os títulos e o tempo de acesso de todos os documentos relevantes (ver 2 na Figura 14).

O próximo passo é fazer uso dos metadados de atenção coletadas que rastreia o que o usuário estava dando atenção e estava interessado. A primeira forma de utilização desses dados é na construção de um perfil de atenção do usuário, que represente o interesse real desse usuário com base na sua interação anterior com diferentes ferramentas e documentos com que trabalhou. Para alcançar isso, usuários e documentos devem ser identificados entre as diferentes aplicações.

A identificação do usuário neste contexto (aplicações *desktop*) não é um problema, uma vez que o XML cliente armazena apenas os dados de um usuário no repositório. Mesmo se mais usuários armazenarem seus metadados de atenção no repositório, o elemento de *group* permitiria a identificação clara de cada um.

A Figura 15 ilustra uma ferramenta simples que explora informações gerais sobre a atenção do usuário. A figura 15 mostra os 10 últimos documentos (ver 1 na Figura 15) que o usuário deu atenção; o *script* XQuery mostrado na Figura 14 é usado para recuperar tais documentos. No lado esquerdo da ferramenta, as palavras-chave que representam o interesse do usuário são mostradas (ver 2 e 3 na Figura 15). Essas palavras-chave são extraídas dos elementos capturados sobre cada documento que o usuário trabalhou nas aplicações (*Microsoft Power Point*, *MSN Messenger*, *WINAMP* e *Firefox Browser*), tais como:

- Título do documento;
- Descrições que um usuário dá aos documentos acessados, utilizando o elemento *event.description;*

- Chave que um usuário pode anexar nos documentos acessados, utlizando o elemento *event.tags;*
- Ações relacionadas aos dados sobre os termos de busca utilizados por um usuário para encontrar informações relevantes, mensagens de *chat* de texto, etc, usando o elemento *event.action.relatedData.*

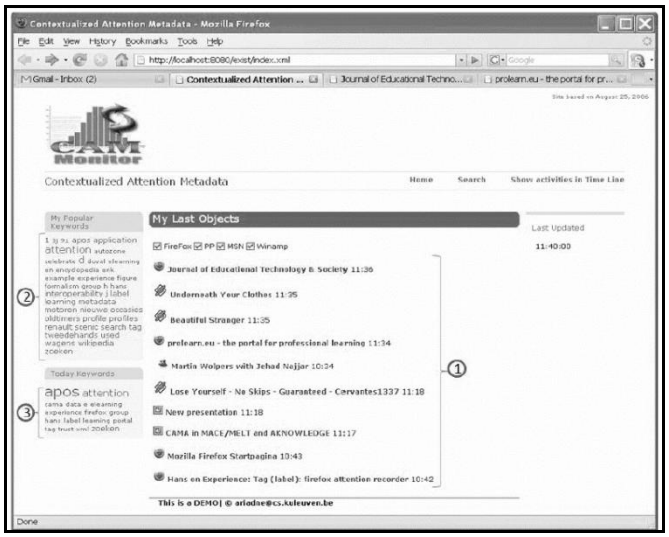


Figura 15 - Imagem da ferramenta Attention Monitor Tool(WOLPERS et al., 2007)

Como mostrado na Figura 15, as palavras-chave representam o interesse do usuário. Os últimos documentos utilizados pelo usuário reflem as tarefas com as quais o usuário esteve envolvido. Os dados permitem a construção de perfis de atenção do usuário. Esses perfis podem mais tarde ser relacionados aos perfis de outras pessoas. Por exemplo, o perfil mantém informações sobre as ferramentas que um usuário utiliza com frequência, o tempo gasto (*duration*) nas ferramentas e nos respectivos documentos, a data e hora de acesso dos usuários às ferramentas e documentos (*dateTime*), os cursos que o usuário realizou e as palavras-chave que representam seus interesses.

Como mencionado anteriormente, os perfis de atenção do usuário representam a atenção do usuário e seus interesses em sistemas diferentes. Pela mineração de informação recolhida a partir de diferentes ferramentas, o perfil pode conter informações sobre o seu padrão de comportamento musical, por exemplo, "sempre que um usuário trabalha com o *Microsoft Power Point*, escuta música clássica, já quando escuta rock, navega na internet." Outro padrão pode ser, "site X e Y são acessados em cerca de 80% das vezes que um usuário inicia uma nova apresentação de *slides* no *Microsoft PowerPoint*". Esses padrões podem ser usados para alertar ou recomendar informações relevantes com base no interesse ou tarefa que o usuário está trabalhando no momento. Além disso, tais padrões permitem a visualização das

atividades do usuário e atenção, para facilmente gerar relatórios de trabalho (WOLPERS et al., 2007).

2.7 USANDO CAM PARA RANQUEAR E RECOMENDAÇÃO DE OBJETOS DE APRENDIZAGEM

Uma das principais razões para capturar e analisar informações sobre as interações entre o usuário e uma ferramenta é melhorar a experiência do usuário (LINDEN et al., 2003). Um exemplo é um sistema de *e-commerce* de livros, onde o mesmo pode registrar os livros comprados anteriormente pelo usuário a fim de recomendar-lhe novas compras similares futuramente, tirando essa tarefa de pesquisa do usuário. Um site de notícias pode registrar o tema das notícias e artigos que um usuário normalmente tem lido a fim de filtrar as notícias de interesse ou não desse usuário (SHEPHERD e WATTERS 2002). O nome genérico aplicado para descrever as informações sobre essas interações foi Metadados de Atenção (NAJJAR et al., 2005).

Quando a informação armazenada não contém apenas a referência para o usuário e a ação que ele executou, mas também registra quando a ação ocorreu por meio de qual ferramenta a ação foi realizada, qual foi o perfil do usuário que executou aquela ação, a qual comunidade que ele pertence, torna-se um registro mais útil e informativo, chamado de Metadados de Atenção Contextualizada (*Contextualized Metadata Attention*– CAM) (WOLPERS et al., 2007).Assim, CAM é uma extensão do AttentionXML.

Ochoa e Duval (2006) acreditam por exemplo, que a busca por Objetos de Aprendizagem (OA) poderia se beneficiar das informações de CAM. A principal razão para isso é a falta de uma forma significativa e escalável de classificar ou recomendar esses objetos aos usuários. A atenção humana (significativa) é processada para a construção de um sistema automatizado (escalável) de classificação e procedimentos de recomendação.

Os usuários interagem diretamente com diversos OA por todo o ciclo de vida de um curso na Web. Gravadores CAM capturam o *timestamp* (momento que ocorreu o evento) e todas as informações de interação, a fim de fornecer informações necessárias para calcular métricas úteis para serem utilizadas em uma ferramenta de gerenciamento de objetos. De acordo com Najjar et al. (2005), essas informações são armazenadas dentro de um registro de ações. As fases do ciclo de vida dos OA são nominadas por Collis e Strijiker (2004) (Tabela 1) da seguinte maneira: criação, rotulação, oferta, seleção, uso e retenção.

Tabela 1 - Proposta de Informações CAM para serem armazenadas por Aplicações de Objetos de Aprendizagem (OCHOA e DUVAL, 2006).

Ciclo de Vida	Ações	Informações Principais	Fonte
Criação	Criando	Autor, Componentes	Ferramentas de autoria, Componentes
Rotulação	Rotulando	Formato de Metadados, Origem, Confiança	Ferramentas de autoria ou Geração de metadados
Oferta	Inserindo	Inserção	Recomendando Objetos de Aprendizagem (LOR) ou compartilhando aplicações
Seleção	Buscando	Consultas, Resultados	Ferramentas de Busca de LOR
	Recomendando	Objetos Recomendados	Recomendador
	Navegando	Tempo	LOR ou Recomendador
	Selecionando	Identificador de Objeto	LOR ou Recomendador
Uso	Publicando	Contexto do LMS	LMS
	Sequenciando	Lista sequenciada de objetos	ID da Ferramenta ou pacote
	Visualizando	Tempo, ferramentas usadas.	Navegando ou Lendo aplicações
	Anotando	Taxa ou revisão	LMS
Retenção	Retendo	Decisão de manter ou apagar	LMS

No contexto das recomendações de REAs em ambientes MOOC, serão utilizadas as fases `seleção e uso.`

`Seleção:` Nesta fase, várias ações devem ser capturadas. A ação `buscando,` quando uma consulta é realizada para encontrar objetos relevantes. Devem-se incluir informações que descrevam a consulta realizada e os objetos devolvidos. A ação `recomendando,` quando o sistema sugere objetos relevantes sem que o usuário realize uma consulta, além da ação do usuário que aciona a recomendação e a ferramenta usada para efetuar a recomendação. A ação `navegando,` quando o usuário analisa o metadado ou a descrição do OA, deve armazenar informações que identificam o registro do metadado navegado e o tempo gasto na revisão do OA. Finalmente, uma ação `selecionando,` quando o usuário escolher o objeto por meio de *download* ou aceite da recomendação. Todas essas ações também devem conter informações sobre o usuário que as executou (OCHOA e DUVAL, 2006).

`Uso:` Esta fase compreende todas as ações que o usuário final executa com o OA durante a sua utilização normal em um LMS (*Learning Management System*). Existem várias ações a serem registradas. A ação `publicando,` quando o instrutor insere um OA em um curso. Essa ação deve conter informações que identificam o objeto publicado e o contexto (curso, aula) onde foi publicado. A ação de `sequenciamento,` quando um ou mais objetos são incluídos em um *design* instrucional. Tal ação deve conter informações sobre a identificação (em forma ordenada) dos objetos integrados. A ação `visualizando,` quando

o objeto é visto ou lido pelo aluno. Essa ação deve conter informações sobre o tempo gasto revendo o material. A ação `anotando`, quando o instrutor ou o aluno adicionar um comentário ou identificador ao objeto. Todas essas medidas também devem armazenar informações sobre o usuário que as executou. Diferentes ferramentas devem ser responsáveis pela geração dos registros de atenção, um LMS para a ação de `publicação`, um LMS *Activity* (DALZIEL, 2006) ou Pacotes SCORM para a ação de `sequenciamento`, um navegador da Web ou editor de documentos para a ação de `visualização` e uma classificação para a ação de `anotando`.

2.7.1 Métricas de Recomendação Baseada em Análise de *Ranking*

Um dos mais famosos e bem sucedidos algoritmos de classificação é o *PageRank* (PAGE, 1998). *PageRank* usa as informações contidas na rede de ligações entre as páginas Web para calcular a importância relativa de uma página. Pode ser resumido como: uma página é importante se está ligada por um elevado número de páginas. Além disso, a importância aumenta se as páginas com *links* importantes também têm um alto nível de classificação. Infelizmente, este algoritmo não poderia ser aplicado diretamente para OA, já que registros LOM têm um campo de `vinculação` que raramente é preenchido. Além disso, LOM reflete apenas em uma relação semântica; isso não implica um "voto" para esse objeto, uma vez que é assumido para páginas Web.

CAM permite criar uma ligação implícita entre o OA e outras entidades relacionadas aos OA: autores, usuários, cursos, etc, por exemplo: Ações `criando` podem ser convertidas em um *link* entre um autor e um objeto, ação `selecionando` pode ser convertida em uma ligação entre um usuário e um objeto, ações `publicando` podem ser conferidas em um *link* entre um curso e um objeto e também entre o usuário e o mesmo objeto. Como resultado desta conversão de CAM para ligações entre diferentes entidades, um grafo k-partido é criado (um grafo com diferentes partições, onde não existem ligações entre os nós da mesma partição). Nesse grafo, cada tipo de elemento (OA, usuário, curso e autor) é considerado uma partição, um exemplo de tal partição está presente do grafo representado na Figura 16 (OCHOA e DUVAL, 2006).

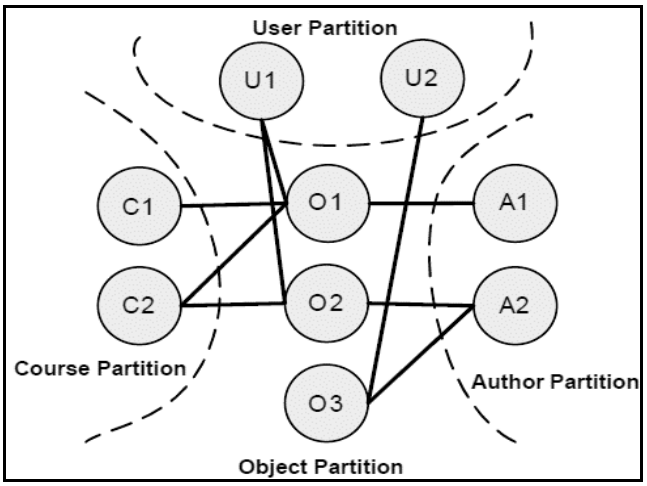


Figura 16 - Representação de CAM em grafo k-partido (OCHOA e DUVAL, 2006)

Uma vez que a informação de CAM é representada como um grafo, pode-se usar algoritmos básicos de grafos para calcular métricas de classificação. Na sequência, existem algumas métricas que podem ser desenvolvidas da seguinte forma (OCHOA e DUVAL, 2006):

- ***Rank* de Popularidade (PR):** Usando as informações contidas na ação `Seleção` (já convertida em um grafo bipartido), pode-se obter o número de vezes que um objeto foi baixado. Para calcular, conta-se apenas o número de ligações incidentes que cada nó de OA tem de nós na **Partição Usuários**. Essa métrica é apenas para colocar os objetos mais baixados em primeiro lugar na lista de resultados, ilustrado na Equação 1.

$$\mathbf{PR(Object) = inDegree\ (Object)} \qquad (1)$$

- ***Rank* Manual (MR):** Usando a informação que é armazenada na ação `Anotando`, o número de vezes que um objeto tem sido positivamente (ou negativamente) avaliado poderia ser considerada para calcular uma métrica. Um grafo bipartido (partições **Usuários e Objetos**) é criado. O procedimento pesará o *link* como 1, se for uma taxa positiva, -1 se for uma taxa negativa. O valor é usado apenas para avaliar se a taxa é um "voto" positivo ou negativo, porque diferentes usuários e sistema têm diferentes escalas de avaliação. O comentário só pode ser considerado se o valor da sua positividade ou negatividade é incluído na ação de `Anotando`, conforme ilustrado na Equação 2.

$$\mathbf{MR\ (Object) = inDegree_{Positive}\ (object) - inDegree_{Negative}\ (object)} \qquad (2)$$

Estas métricas podem ser calculadas *offline*, pois não usam uma consulta específica. Calculam uma importância média ou relevância dos objetos de aprendizagem com base na sobrecarga de informação de atenção. Estas métricas, e outras, que podem ser desenvolvidas posteriormente, poderiam ser integradas em um *ranking* de métricas final, chamada de Métrica de Popularidade Composta **(MPC)** e pode ser calculada como a soma ponderada dos valores das métricas individuais. Por exemplo, o Google integra mais de 100 diferentes métricas simples, a fim de gerar seus resultados (OCHOA e DUVAL, 2006). Ilustrado na Equação 3.

$$\mathbf{MPC = \alpha PR + \beta MR} \qquad (3)$$

Os coeficientes ponderados (alfa, beta, etc) devem ser estimados para fornecer uma boa ordenação de resultado. Métodos para fazer essas estimativas são descritas em (RADLINSKI e JOACHIMS, 2005) e (FAN et al., 2004). Além disso, taxas manuais devem ser usadas com cuidado, pois a ação `Anotando` é uma informação opcional e não poderia existir para todos os objetos envolvidos no cálculo.

2.7.2 Métricas de Similaridade para Recomendação

Uma propriedade de um grafo bipartido é que ele pode ser dobrado ao longo de uma de suas divisórias, gerando um grafo normal, com apenas uma entidade e as relações entre seus nós. Por exemplo, se existe um grafo bipartido de usuários que fizeram *download* do OA, é possível dobrar neste a partição de OA e ter-se-á um grafo onde os usuários estão ligados entre si. Este novo grafo pode ser utilizado para calcular a semelhança entre os usuários com base nos padrões de *download*. Na Figura 17, é possível ver uma representação do resultado da dobradura.

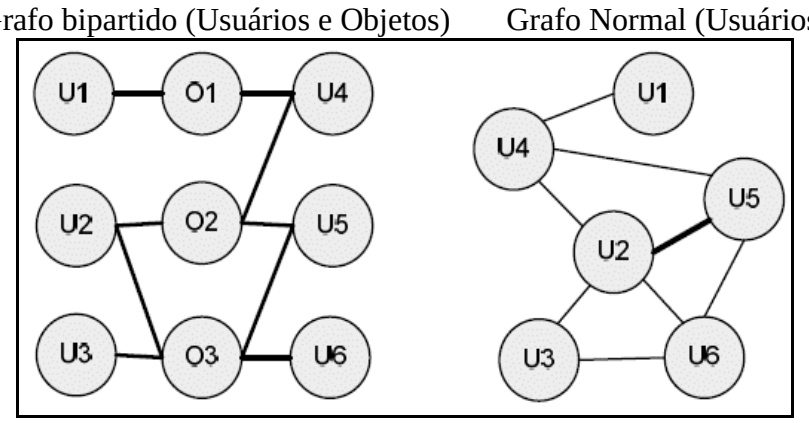


Figura 17 - Grafo Bipartido dobrado e desdobrado (OCHOA e DUVAL, 2006)

A primeira parte da figura representa um grafo bipartido com partições de **Usuário** e **Objetos**. O grafo mostra que, por exemplo, o Usuário 5 tinha baixado Objeto 2 e 3 e o Usuário 1 só tinha baixado o Objeto 1. A segunda parte da figura ilustra a versão dobrada do grafo. No novo grafo, os usuários têm uma ligação entre si se forem ligados a um mesmo objeto no grafo desdobrado. Por exemplo, o Usuário 1 e Usuário 4 estão ligados porque ambos têm Objetos baixados, Usuário 2 e 5 têm laços mais fortes porque ambos têm baixado objetos 2 e 3. (NASCIMENTO et al., 2003).

No contexto deste trabalho duas métricas de similaridade que podem ser calculadas usando as informações contidas em ações de CAM (detalhadas anteriormente segundo Ochoa e Duval (2006) são:

- **Similaridade de objetos baseada em número de *downloads*:** Criar um grafo bipartido com as informações das ações `Selecionando` (quando um usuário baixar um OA) e dobrar sobre a **Partição de Usuários**. Uma ligação entre dois Objetos no grafo final significa que esses objetos foram baixados pelo mesmo usuário. A força da semelhança é o número de usuários que tenham baixado os dois objetos;
- **Usuários semelhantes baseados em *download:*** Criar um grafo bipartido com as informações da ação `Selecionando` e dobrar sobre a **Partição Objetos**. Uma ligação entre dois usuários significa que baixaram o mesmo objeto. A força de similaridade é o número de objetos que os usuários têm em comum.

Após o cálculo das métricas de similaridade obtidas a partir dos grafos, pode-se aplicar esses valores em ferramentas de recomendação (OCHOA e DUVAL, 2006). Por exemplo: se um usuário encontra um objeto útil, um *link* para objetos semelhantes poderia ser fornecido (similar ao que a Amazon faz com livros (LINDEN et al., 2003)).

Recomendação Contextual: CAM são considerados não apenas como uma fonte de dados de histórico, mas também como um fluxo contínuo de atenção contextualizada de informação, pode-se usar CAM na ordem de segundos ou minutos para gerar recomendações com base no que o usuário está concentrado no momento. Por exemplo, o sistema de recomendação poderia usar as informações armazenadas caso o usuário tenha inserido um objeto dentro de um curso em um LMS, o LMS irá gerar um registro CAM com informações contextuais sobre qual objeto foi inserido e em qual aula do curso. O sistema de recomendação poderia usar essa informação para apresentar ao usuário objetos para serem inseridos ou outros que tenham sido usados em cursos semelhantes, com base no tema do curso ou em métricas de similaridade.

O sistema de recomendação pode também apresentar objetos que se adequem ao aplicativo que o usuário está usando em um dado momento, com base nas informações sobre o objeto (registro LOM). Por exemplo, se o usuário está trabalhando na ferramenta *Microsoft Power Point*, apresentações, *slides*, pequenos textos, imagens e diagramas serão recomendados. Se está trabalhando com um Pacote SCORM, OA completos serão apresentados.

Técnicas de recomendações contextuais foram analisadas em vários campos (GOOGLE, 2006) (FAN et al., 2004). Blinkx (2006) é um exemplo deste tipo de aplicação, recomenda páginas Web, vídeos e notícias com base no conteúdo atual da tela do usuário. Uma aplicação semelhante pode ser desenvolvida em um LMS, por exemplo, onde o sistema poderia recomendar materiais a serem adicionados pelos instrutores para adicionar a cada lição, ou poderia recomendar ao aluno materiais semelhantes ou complementares ao que o instrutor tenha adicionado para o curso.

Algoritmos de recomendação contextuais: Existem diferentes métodos para incorporar informação contextual no processo de recomendação (ADOMAVICIUS e TUZHILIN, 2005) incluindo recomendação via consulta orientada a contexto e método de pesquisa, pré-filtragem contextual, pós-filtragem e modelagem contextual. Alguns Sistemas de Recomendação (SR) contam com recomendações via consulta orientada ao contexto e método de pesquisa. Esses sistemas cruzam dados contextuais com metadados de recursos para obter os recursos apropriados. Por exemplo, MOBIlearn (LONSDALE et al., 2004) corresponde ao local e tema atual de interesse para descrições de recursos de aprendizagem. TANGO (OGATA, 2004) corresponde ao nível do aluno e sua localização, com as descrições de recursos de conhecimento. Outros sistemas contam com um método de consulta baseado em ontologias para filtrar os recursos adequados, incluindo APOSDLE (STERN et al., 2010).

A principal diferença dos sistemas de recomendação tradicionais, é que os dados de contexto adicionais são usados para recuperar os recursos relevantes a partir de um repositório.

Alguns outros sistemas utilizam um algoritmo de pós-filtragem para filtrar os resultados de um algoritmo de recomendação tradicional baseado em informação contextual. Por exemplo, COLDEX (BALOIAN et al., 2004) filtra os resultados em um algoritmo de recomendação tradicional baseado no contexto de computação. PLCR 2 (BERRI et al., 2006) filtram os resultados com base tanto na computação quanto nas limitações de tempo. O algoritmo de pós-filtragem CALS (YAU e JOY, 2007) também leva em conta as condições físicas. C- LINK (SCHIRRU et al., 2010 e ZHAO et al., 2008) e o de recomendação 3A (EL HELOU et al., 2010) usam uma técnica de modelagem contextual que adapta-se a um algoritmo de recomendação tradicional levando em consideração a informação contextual. O algoritmo de recomendação 3A adapta uma versão do algoritmo *PageRank* do Google para o *framework* de modelagem especial, que considera contexto de relação. C- LINK usa tentativa de aprender as preferências contextualizadas do usuário por meio do modelo de perfis. Tais sistemas adaptam algoritmos tradicionais e incorporam as preferências multidimensionais do usuário, por exemplo, que recursos que um usuário gostou quando estava trabalhando em uma determinada atividade.

De modo geral, há necessidade de explorar de forma mais abrangente trocas entre as diferentes abordagens, a fim de desenvolver uma melhor compreensão sobre quais os algoritmos usar e como combiná-los (ADOMAVICIUS e TUZHILIN, 2005).

2.8 SISTEMA DE PESQUISA FEDERADA UTILIZANDO *WIDGET*

A pesquisa federada pode ser usada como pesquisa em vários locais e fontes de informações (GOVAERTS et al., 2011). Em sistemas de pesquisa federada, a tarefa é procurar um grupo independente de coleções, para efetivamente mesclar os resultados que tais sistemas retornam para consultas, como ilustrado na Figura 18.

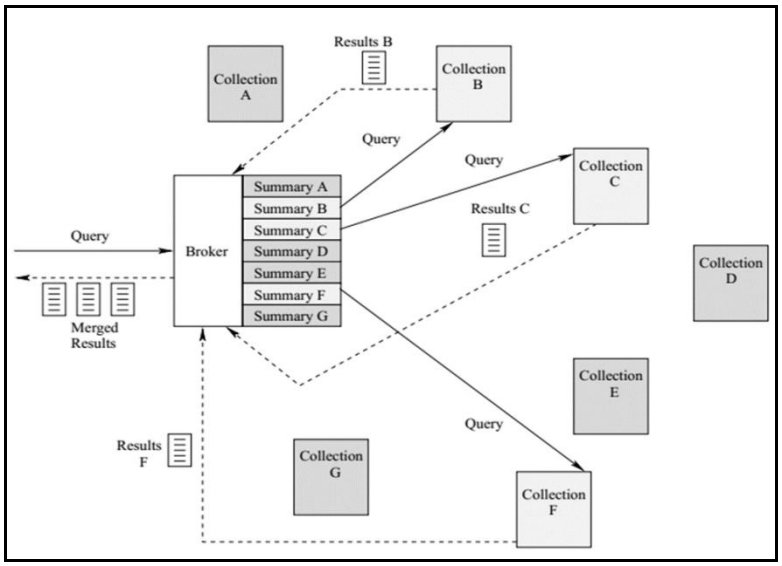


Figura 18 - Arquitetura de um Sistema de pesquisa federada (GOVAERTS et al., 2011)

A Figura 18 mostra a arquitetura de um sistema de busca federada. A peça central é agente (*Broker*) que recebe consultas dos usuários e as envia para as coleções de dados ou repositórios que são considerados com maior probabilidade de conter respostas das consultas feitas.

As coleções destacadas na Figura 18 são selecionadas para consulta. Para realizar consultas adequadas, o agente precisa armazenar algumas informações importantes (resumo ou representação) sobre as coleções disponíveis.

Uma técnica que pode ser utilizada para obter informações sobre as coleções em tais ambientes, é enviando uma amostragem (consultas) para cada coleção. A informação recolhida a partir do número de documentos de resposta a essas perguntas é usada para construir um conjunto de representações; esse conjunto orienta a avaliação e classificação das coleções. As coleções selecionadas recebem a consulta dos seus próprios índices pelo agente. Na etapa final, os resultados retornados pelas coleções selecionadas são apresentados ao usuário.

A pesquisa federada desempenhou um papel chave no fornecimento de tecnologia para a pesquisa agregada e *crawling* para a Web profunda. Há muitas situações em que a informação é distribuída por meio de diferentes fontes/servidores. *Peer-to-peer* e pesquisa personalizada são dois exemplos onde a pesquisa federada tem sido utilizada com sucesso para pesquisa em vários locais independentes (LU, 2007; THOMAS, 2008).

Dentre os sistemas de pesquisa federada existentes, dois merecem destaque por sua completude no contexto educacional. O projeto MERLOT - *Multimedia Educational Resource for Learning and Online Teaching,* que se refere ao desenvolvimento cooperativo e

gratuito de recursos baseados na Web para que professores, alunos e profissionais possam facilmente encontrar e disponibilizar materiais digitais de aprendizagem com suas respectivas avaliações e indicações de usos mais apropriados. Sua pesquisa federada é realizada entre os bancos de dados de dezenas de universidades, faculdades isoladas, associações de pesquisadores, bibliotecas digitais e empresas americanas, podendo ser selecionadas as áreas de *Multimedia Educational Resource for Learning and Online Teaching* ou *Digital Resource Collections for Physics and Astronomy Education*. (EL HELOU et al., 2010).

Outro sistema é o *EduSource*, sob coordenação do Grupo Canarie do Canadá, cuja proposta é a criação de uma robusta infraestrutura tecnológica para o compartilhamento nacional de repositórios de objetos de aprendizagem. Sua fase inicial foi o inventário das ferramentas, sistemas, protocolos e práticas disponíveis no país, de modo a definir componentes para um *framework* que garantisse a interoperabilidade entre as várias instituições. Em 2010 existiam cerca de onze instituições participando desse projeto e disponibilizando seus conteúdos para a busca federada (EL HELOU et al., 2010).

No contexto deste trabalho específico de dissertação a pesquisa federada é aplicada por meio de *widget*. O *widget* em questão é do tipo alto nível já que é implementado e funcionará como buscador usando o *SeeOER* (GAZZOLA et al., 2014) e visualizador das recomendações no ambiente MOOC, por meio do *browser* do usuário.

2.9 *SEEOER* – MECANISMO DE BUSCA NA WEB POR REAS

O *SeeOER* é derivado da arquitetura de um mecanismo de busca na Web em grande escala (GAZZOLA et al., 2014). Portanto, sua arquitetura possui dois processos principais, o processo de indexação e o processo de consulta, ilustrados na Figura 19.

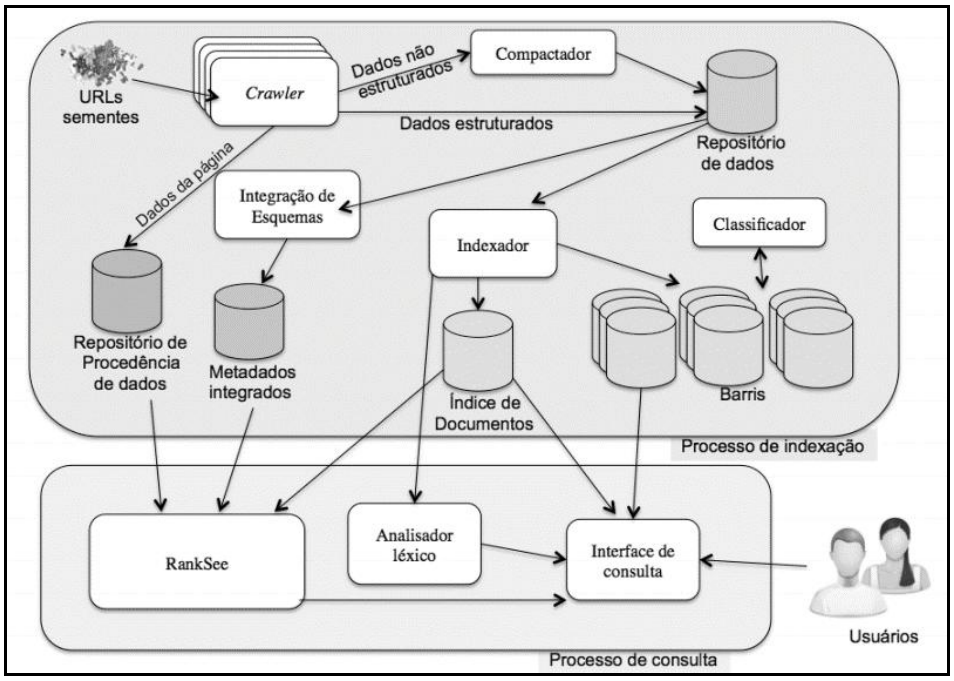


Figura 19 - Arquitetura do *SeeOER* (GAZZOLA et al., 2014).

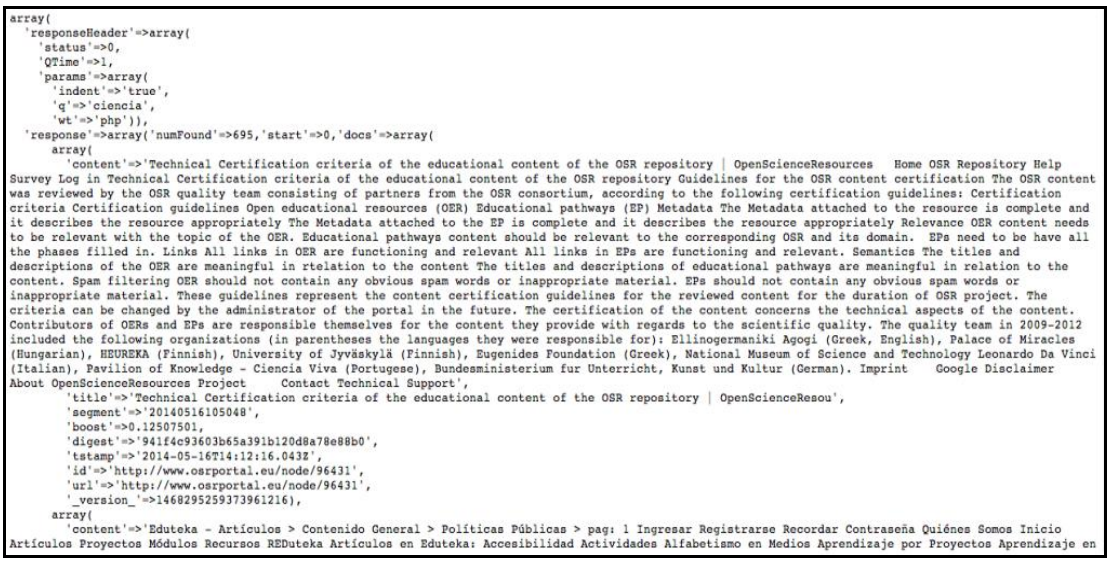
A busca inicia com uma lista de URLs sementes a serem rastreadas pelo *Crawler*, o qual faz o *download* das páginas referentes a esses URLs de forma distribuída. Os dados estruturados representam os metadados, enquanto que os dados não estruturados representam o próprio material, o qual pode estar em diversos formatos (por exemplo, HTML, PDF, DOC). Os dados não estruturados são enviados ao compactador, o qual comprime e armazena-os no repositório, de forma que cada URL seja associado a um número de identificação único. Os dados estruturados são armazenados no repositório, mas de forma que possam ser uniformizados futuramente. Os dados das páginas, como origem do servidor, última atualização, entre outros dados, são armazenados no Repositório de Procedência de dados. O componente de integração de esquemas tem como objetivo criar um esquema integrado representativo, que represente a maioria dos metadados. Esse esquema representativo é armazenado no repositório de metadados integrados. Por fim, o componente *RankSee* terá como funcionalidade recuperar e ranquear os REAs, levando em consideração características intrínsecas de REAs podendo auxiliar na busca e atendam as requisições dos usuários (GAZZOLA et al., 2014).

O `Indexador` tem várias funções. Lê o repositório de dados, descompacta os documentos e os analisa. Cada documento é convertido em um conjunto de ocorrências de texto chamado de *hits*. Cada *hit* possui a palavra, a posição no documento, uma aproximação do tamanho da fonte e a capitalização (maiúsculo ou minúsculo). Esses *hits* são distribuídos na forma de um índice para frente (*forward index*) em um conjunto de *"barris"*. O

`Classificador` realiza, dentre outras operações, a geração de um índice invertido que é usado pela `Interface de consulta` conjuntamente com o Analisador Léxico e o *RankSee*.

Um dos componentes chave do mecanismo de busca na Web é o *Crawler*. Na arquitetura do *SeeOER* foi proposto um *Crawler* que obtivesse metadados das páginas dos repositórios REAs, como também os dados de procedências das páginas capturadas (GAZZOLA et al., 2014).

Para o contexto deste trabalho, far-se-á uso da API de consulta dos resultados indexados do *SeeOER*, onde o acesso por esses resultados pode ser realizado por meio do *widget*. O retorno dessa API possui vários formatos de respostas, os quais são: i) Ruby; ii) JSON; iii) Python e iv) PHP. O retorno já vem sinteticamente e semanticamente na opção do formato pedido ao *SeeOER*. A Figura 20 ilustra o retorno do *SeeOER* para linguagem PHP.



```
array(
  'responseHeader'=>array(
    'status'=>0,
    'QTime'=>1,
    'params'=>array(
      'indent'=>'true',
      'q'=>'ciencia',
      'wt'=>'php')),
  'response'=>array('numFound'=>695,'start'=>0,'docs'=>array(
      array(
        'content'=>'Technical Certification criteria of the educational content of the OSR repository | OpenScienceResources   Home OSR Repository Help
Survey Log in Technical Certification criteria of the educational content of the OSR repository Guidelines for the OSR content certification The OSR content
was reviewed by the OSR quality team consisting of partners from the OSR consortium, according to the following certification guidelines: Certification
criteria Certification guidelines Open educational resources (OER) Educational pathways (EP) Metadata The Metadata attached to the resource is complete and
it describes the resource appropriately The Metadata attached to the EP is complete and it describes the resource appropriately Relevance OER content needs
to be relevant with the topic of the OER. Educational pathways content should be relevant to the corresponding OSR and its domain.  EPs need to be have all
the phases filled in. Links All links in OER are functioning and relevant All links in EPs are functioning and relevant. Semantics The titles and
descriptions of the OER are meaningful in rtelation to the content The titles and descriptions of educational pathways are meaningful in relation to the
content. Spam filtering OER should not contain any obvious spam words or inappropriate material. EPs should not contain any obvious spam words or
inappropriate material. These guidelines represent the content certification guidelines for the reviewed content for the duration of OSR project. The
criteria can be changed by the administrator of the portal in the future. The certification of the content concerns the technical aspects of the content.
Contributors of OERs and EPs are responsible themselves for the content they provide with regards to the scientific quality. The quality team in 2009-2012
included the following organizations (in parentheses the languages they were responsible for): Ellinogermaniki Agogi (Greek, English), Palace of Miracles
(Hungarian), HEUREKA (Finnish), University of Jyväskylä (Finnish), Eugenides Foundation (Greek), National Museum of Science and Technology Leonardo Da Vinci
(Italian), Pavilion of Knowledge - Ciencia Viva (Portugese), Bundesministerium fur Unterricht, Kunst und Kultur (German). Imprint    Google Disclaimer
About OpenScienceResources Project     Contact Technical Support',
        'title'=>'Technical Certification criteria of the educational content of the OSR repository | OpenScienceResou',
        'segment'=>'20140516105048',
        'boost'=>0.12507501,
        'digest'=>'941f4c93603b65a391b120d8a78e88b0',
        'tstamp'=>'2014-05-16T14:12:16.043Z',
        'id'=>'http://www.osrportal.eu/node/96431',
        'url'=>'http://www.osrportal.eu/node/96431',
        '_version_'=>1468295259373961216),
      array(
        'content'=>'Eduteka - Artículos > Contenido General > Políticas Públicas > pag: 1 Ingresar Registrarse Recordar Contraseña Quiénes Somos Inicio
Artículos Proyectos Módulos Recursos REDuteka Artículos en Eduteka: Accesibilidad Actividades Alfabetismo en Medios Aprendizaje por Proyectos Aprendizaje en
```

Figura 20 - Retorno da API do *SeeOER* no formato de saída PHP usando http://usp.seeoer.com/?q=ciencia&wt=php&indent=true (GAZZOLA et al., 2014).

A Figura 20 mostra os resultados da expressão de busca "ciência". São retornados para esta expressão 695 documentos, por padrão são retornados 10 documentos por intervalo, começando pelo zero (*start=0*). É possível alterar esse intervalo de 10 documentos, para isso é necessário incluir na consulta à API a quantidade de linhas de retorno usando o parâmetro *rows*. Por exemplo, rows=100 retornaria o intervalo de 100 documentos por consulta. Quanto maior o valor do parâmetro *rows* da quantidade de linhas de retorno passadas, maior será o tempo de latência de resposta entre a consulta e a reposta do *SeeOER*.

A consulta é feita por meio da seguinte URL http://usp.seeoer.com/ adicionando os parâmetros de busca, os quais são: expressão de busca (*q*), início (*start*), intervalo (*rows*),

indentação (*indent*) e formato. Todos os parâmetros são opcionais, exceto o *q*. É possível incluir os seguintes parâmetros na API nas linhas de retorno *rows*. Por exemplo: http: //usp.seeoer.com/?q=ciencia&wt=php&rows=600. A API retornaria os 600 resultados da consulta "ciência" no formato PHP.

Também é possível incluir a indentação usando o parâmetro *indent*. Por exemplo: http://usp.seeoer.com/?q=ciencia&wt=php&rows=600&**indent=true**, como mostrado na Figura 21, a reposta está indentada para facilitar a identificação do retorno ao usuário.

Os parâmetros válidos para formatos de respostas são:

- Formato Ruby) wt=ruby
- Formato Python) wt=python
- Formato JSON) wt=json
- Formato PHP) wt=php

Qualquer outro formato que não estejam contidos nessa lista acima e que forem colocadas neste parâmetro, a saída será mal formatada (GAZZOLA et al., 2014).

É importante salientar que após a indexação dos resultados, o SR irá predizer ao usuário um conjunto de índices, por meio do campo *URL*, resultantes da busca, na forma de links para que o mesmo possa baixá-los, como é mostrado pontilhado na Figura 21.

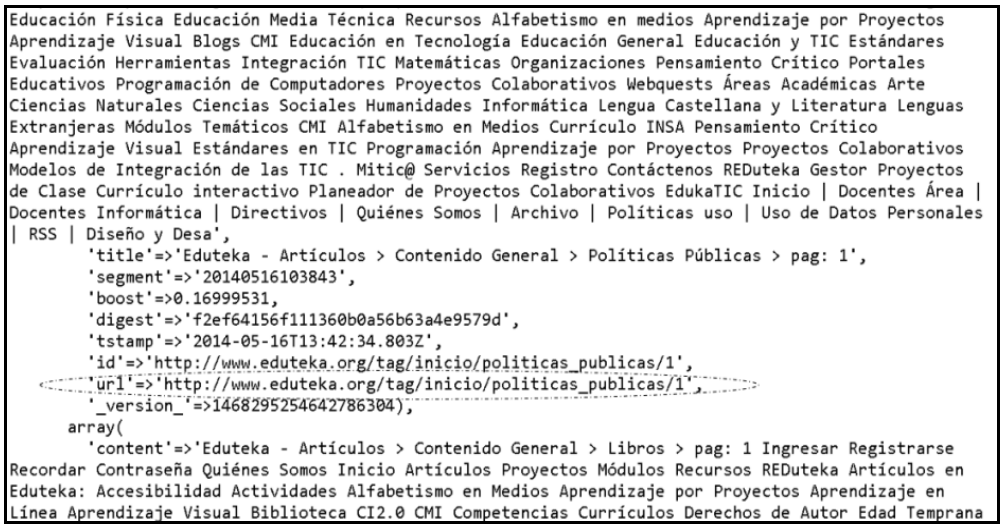


Figura 21 - Campo URL utilizado pelo *OERecommender* para ser mostrado ao usuário em forma de *link*

2.10 SISTEMAS DE RECOMENDAÇÃO

SR são compostos de técnicas e ferramentas de *software* que fornecem sugestões de itens que podem ser úteis aos usuários (RESNICK e VARIAN, 1997) (BURKE, 2007) (MAHMOOD e

RICCI, 2009). O desenvolvimento de SR iniciou-se a partir de uma observação simples: indivíduos muitas vezes dependem de recomendações fornecidas por outras pessoas para a tomada de decisões diárias (MAHMOOD e RICCI, 2009) (MCSHERRY e MIRONOV, 2009). Conforme Jannach et al. (2011), os SR têm se revelado nos últimos anos um meio valioso para lidar com o problema da sobrecarga de informação, pois indica aos usuários novos itens, ainda não conhecidos que possam ser relevantes. As ações do usuário e *feedbacks* podem ser armazenadas no banco de dados de recomendação e podem ser utilizados para a geração de novas recomendações em suas próximas interações.

As sugestões referem-se a vários processos de tomada de decisão, tais como, os itens para comprar, a música para ouvir ou as notícias *online* para ler. Item é o termo geral usado para designar o que o sistema recomenda aos usuários. O SR normalmente se concentra em um tipo específico de item e, consequentemente, seu *design*, sua interface gráfica para o usuário e a técnica usada para gerar as recomendações são personalizadas para fornecer sugestões úteis e eficazes (RICCI et al., 2011).

Um exemplo é um sistema de recomendação de livros que auxilia os usuários a escolherem um livro para ler. No site popular, Amazon.com, é utilizado um SR para personalizar a loja *online* para cada cliente (JANNACH, 2006). Essas recomendações geralmente são personalizadas para usuários individuais ou grupos de usuários. Além disso, há também recomendações não personalizadas, que são mais simples de gerar e, normalmente, são destaque em revistas ou jornais (RICCI et al., 2011).

Em sua forma mais simples, recomendações personalizadas são oferecidas como listas de classificados de itens. O SR tenta prever quais os produtos ou serviços são mais adequados com base nas preferências do usuário. A fim de concluir uma tarefa, os SRs coletam a partir dos usuários as suas preferências, as quais são explicitamente expressas.

2.10.1 Classificação dos Sistemas de Recomendação

Adomavicius e Tuzhilin (2005) classificam os sistemas de recomendação como: Sistemas de Recomendação baseados na Filtragem de Conteúdo, Sistemas de Recomendação baseados na Filtragem Colaborativa e Sistemas de Recomendação Híbridos.

Este trabalho aborda um SR baseado na Filtragem Colaborativa, e aborda também: Sistemas de Recomendação baseados em Modelos com descrição de alguns métodos, sendo eles: **(i)** Similaridade entre Itens; **(ii)** Algoritmos baseados em Similaridade por Cosseno; **(iii)**

Algoritmos baseados em Pearson; (**iv**) Algoritmos baseados em Cosseno Ajustado; e **(iv)** Tamanho de Modelo.

Os detalhes dos SR são descritos a seguir:

`Sistema de Recomendação baseado na Filtragem de Conteúdo:` Analisam o histórico da interação entre usuário e o ambiente, com o objetivo de se obter quais produtos/serviços foram adquiridos pelo usuário no passado (FELFERNIG e BURKE, 2008). Eles focam na recomendação de informações textuais, como documentos, sites, *blogs* e fóruns. Utiliza-se do perfil do usuário, tentando recomendar itens que sejam similares aos que o usuário gostou no passado. O foco desses sistemas é aprender as preferências do usuário e filtrar dentre os novos itens, aqueles que mais se adequarem às suas preferências. Em um SR baseado em Conteúdo, o usuário fornece, de forma implícita ou explícita, suas preferências e restrições, e o sistema cruza essas descrições com os itens contidos em um catálogo de itens (JANNACH et al., 2011). Como Adomavicius e Tuzhilin (2005) explicam, a utilidade u(c, s) de um produto s para um usuário c tem seu cálculo baseado na utilidade u(c, s_i), tal que s_i são os itens pertencentes ao conjunto S, adquiridos em algum momento do passado.

Embora essa técnica seja muito utilizada, têm algumas limitações (JANNACH et al., 2011):

- Os atributos precisam ser cadastrados manualmente para os itens, mesmo com a atual tecnologia, mídias como som e vídeo apresentam dificuldades de serem analisadas automaticamente para a extração automática dos atributos;
- Nessa técnica, são encontrados apenas itens parecidos com os já conhecidos pelo usuário, não é possível recomendar o item se o usuário ainda não avaliou itens similares;
- Os métodos não são capazes de avaliar as descrições dos itens quanto a questões subjetivas, como qualidade.

`Sistema de Recomendação baseado na Filtragem Colaborativa:` A Filtragem Colaborativa é uma técnica que utiliza os históricos das interações entre usuários e itens buscando encontrar relacionamentos entre os mesmos. Seu algoritmo consiste em montar uma matriz de pontuações, em que as linhas representam os usuários e as colunas representam os itens, de modo que sejam identificados grupos de usuários com as pontuações aproximadas.

Nesse contexto, quanto mais próximas as pontuações, mais semelhantes são os perfis dos usuários (BERKONSKY et. al., 2008). Em termos formais, a técnica tenta prever a

utilidade *u(c,s)* do item para o usuário, com base na utilidade desse mesmo produto para um conjunto de usuários $c_i \in S$ que possuem características similares às suas (BURKE, 2002).

Um problema dessa técnica é que quando se tenta gerar recomendações sem a existência prévia de um histórico do usuário. Esse problema é conhecido como "Partida Fria de Usuário" ou *Cold Start User* (SCHAFER et al., 2007), o qual acontece quando o SR não possui informações suficientes sobre o usuário para gerar as recomendações.

- Sistemas de Recomendação baseados em Modelos: Sistemas baseados em modelos, usam o conjunto de avaliações para aprender um modelo, que é usado então para fazer as predições e recomendações. Modelos são entidades que sintetizam o comportamento dos dados. Sistemas baseados em modelos foram criados para resolver os problemas dos algoritmos baseado em memória (LINDEN et al., 2003; SARWAR et al., 2001). Esses sistemas analisam a estrutura da matriz A que relaciona usuários e itens para encontrar relações entre os itens. A ideia por trás dessa estratégia vem da intuição de que o usuário se interessaria por itens similares aos itens bem avaliados por ele e evitar os itens similares aos itens que ele não gostou no passado (LINDEN et al., 2003; SARWAR et al., 2001). Além disso, essa técnica não precisa identificar a vizinhança de usuários similares que apresenta o gargalo de desempenho dos algoritmos baseados em memória. Em consequência, tende a produzir recomendações muito mais rapidamente do que os baseados em memória (LINDEN et al., 2003; SARWAR et al., 2001).

  Para esta seção, item tem o mesmo significado que REA para o contexto deste trabalho.

  Diferentes técnicas foram sugeridas para abordar as recomendações baseadas em modelo. Uma abordagem probabilística é vista em Breese et. al., (1998) enquanto que Sarwar et al., (2001) apresentam uma abordagem mais tradicional que explora a correlação item-a-item (baseado em itens).

  Nas abordagens baseadas em itens, analisa-se o conjunto de itens avaliados pelo usuário alvo, $I_a \in I$, computa-se a similaridade entre esses itens e um item *i* e depois seleciona-se uma lista com os *k* itens mais similares $\{i_1, i_2, ..., i_k\}$. Ao mesmo tempo guarda-se a lista de similaridades correspondente $\{s_{i1}, s_{i2}, ..., s_{ik}\}$. A predição é então obtida por uma média ponderada dos itens do usuário, I_a , sobre cada um dos itens *i*.

- Similaridade entre Itens: A ideia básica para encontrar a similaridade entre dois itens *i* e *j* é isolar os usuários que tenham avaliados ambos *i* e *j* e depois aplicar uma métrica de similaridade para calcular a similaridade $s_{i,j}$, como representado na Figura 22.

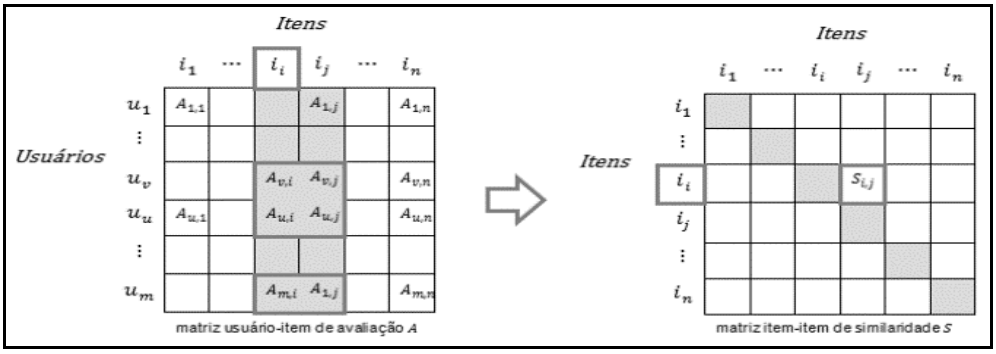


Figura 22 - Encontrando a Similaridade

A métrica da similaridade entre *i* e *j* ($S_{i,j}$) é feita usando apenas os pares de avaliação de usuários em comum.

Nesse exemplo os pares vêm dos usuários *v, u* e *m.*

Assim pode-se calcular a similaridade entre itens usando diferentes algoritmos de similaridade, segue abaixo alguns desses algoritmos:

- Algoritmos baseados em Similaridade por Cosseno: A similaridade entre dois itens, sim(i,j), é dada por uma pequena modificação, Equação 4:

$$sim(i,j) = \cos(\vec{i},\vec{j}) = \frac{\vec{i}.\vec{j}}{\|\vec{i}\|_2 \times \|\vec{j}\|_2} = \frac{\sum_{u \in U_{ij}} A_{u,j} A_{u,j}}{\sqrt{\sum_{u \in U_{ij}} A_{u,i}^2} \sqrt{\sum_{u \in U_{ij}} A_{u,j}^2}} \quad (4)$$

- Algoritmos baseados em Pearson: O coeficiente de Pearson também é usado para calcular similaridades entre itens sendo obtidas pela Equação 5:

$$sim(i,j) = \frac{\sum_{u \in U_{ij}} (A_{u,i} - \bar{A}_i)(A_{u,j} - \bar{A}_j)}{\sqrt{\sum_{u \in U_{ij}} (A_{u,i} - \bar{A}_i)^2} \sqrt{\sum_{u \in U_{ij}} (A_{u,j} - \bar{A}_j)^2}} \quad (5)$$

Onde $\bar{A}_i$ e $\bar{A}_j$ representam a média das avaliações de $_i$ e $_j$ respectivamente.

- Algoritmos baseados em Cosseno Ajustado: Assim como o coeficiente de Pearson, o cosseno ajustado, leva em consideração a diferença entre as

notas dos usuários. Essa métrica (6) de similaridade foi proposta no trabalho de Sarwar et al., (2001).

$$sim(i,j) = \frac{\sum_{u \in U_{ij}} (A_{u,i} - \bar{A}_u)(A_{u,j} - \bar{A}_u)}{\sqrt{\sum_{u \in U_{ij}} (A_{u,i} - \bar{A}_u)^2} \sqrt{\sum_{u \in U_{ij}} (A_{u,j} - \bar{A}_u)^2}} \quad (6)$$

Onde $\bar{A}_u$ é a média de todas as avaliações do usuário $_u$.

- Tamanho do Modelo**:** Os algoritmos baseados em modelos têm uma grande vantagem com relação aos baseados em memória, pois podem ser divididos em duas fases, uma *offline* e outra *online*, a fase de cálculo de similaridade entre os itens pode ser calculada e armazenada em uma estrutura. No momento da recomendação, um simples acesso a essa estrutura busca pelas similaridades necessárias tomando o processo de recomendação bastante rápido (DESHPANDE e KARYPIS, 2004; SARWAR et al., 2001). Outra vantagem é que a escalabilidade deste modelo independe do tamanho da base de usuários, o que faz com que seja uma boa alternativa para os ambientes de MOOC, onde concentram-se grandes quantidades de usuários.

  A estrutura que armazena a similaridade precisa apenas ser atualizada de tempos em tempos para refletir as mudanças nas relações entre os itens. Porém essas relações são estáveis e não mudam tão frequentemente quanto a relação entre usuários que é obtida por meio de grafos, conforme visto na Seção 2.7.1. As similaridades podem ser vistas como uma matriz quadrada de tamanho $n = |I|$. Apesar de economizar tempo, esse modelo requer um custo computacional de pior caso na ordem de $O(n^2)$.

  Para este trabalho, utilizar-se-á o algoritmo *Slope One* para realizar as recomendações ao usuário. Ele é um algoritmo baseado em modelos e suas predições são calculadas a partir da comparação entre avaliações de usuários a certos REAs.

`Sistemas de Recomendação Híbridos`: SR híbrido são sistemas que combinam duas ou mais técnicas de recomendação para poder indicar itens/serviços aos

usuários. Segundo Resnick (2000), existem diversas formas de combinar Filtragem Colaborativa e Filtragem baseada em Conteúdo, a saber:

- Desenvolvendo os métodos separadamente e depois combinando as predições;
- Incorporando algumas características do modelo baseado em conteúdo no modelo colaborativo;
- Incorporando algumas características do modelo colaborativo no modelo baseado em conteúdo;
- Construindo um modelo único que possua características das duas técnicas ao mesmo tempo.

2.11 CONSIDERAÇÕES FINAIS

Neste capítulo foram apresentados os principais conteúdos que fundamentaram teoricamente a realização deste trabalho. Nos próximos capítulos serão abordados: a proposta e, posteriormente, a avaliação do SR de REAs em ambientes MOOC.

3 – PROJETO DO *OERECOMMENDER*

3.1 CONSIDERAÇÕES INCIAIS

Este capítulo apresenta o projeto de um SR para MOOCs, denominado *OERecommender*, que é uma extensão das funcionalidades dos ambientes MOOCs atuais.

As bases referenciais principais para a criação deste trabalho foram os estudos de Ochoa e Duval (2006) que propuseram recomendar OA considerando informações contextuais dos usuários para esse fim, muito semelhante ao contexto atual deste projeto, que é a recomendação de REAs com base no contexto do usuário. E os estudos de Wolpers et al. (2007) que descrevem uma abordagem de CAM, embasando este trabalho sobre quais e como gerenciar os metadados de atenção contextualizada produzidos pelos usuários na interação com o ambiente.

Govaerts et al. (2011) desenvolveram uma arquitetura de pesquisa federada e serviço de recomendação social por meio de *widget*, que foi base para o desenvolvimento da arquitetura com seus componentes e seus relacionamentos para um SR. Outro importante trabalho utilizado foi o de Gazzola et al. (2014), que propuseram um mecanismo de busca na Web por REAs e disponibilizaram uma API para fazer as busca dos mesmos. Essa API é utilizada no trabalho aqui proposto como meio de realizar a etapa de busca por REAs utilizando palavras-chave; e também Sarwar et al. (2001) com algoritmos de recomendação de filtragem colaborativa baseada em itens, dentre os algoritmos e métodos utilizados por Sarwar et al. (2001), foi possível decidir qual o mais apropriado para o contexto deste trabalho.

Para descrever as partes do projeto do *OERecommender*, inicialmente, apresenta-se como foi concebido o modelo conceitual, suas funcionalidades e arquitetura. A Seção de modelo conceitual apresenta os elementos dos ambientes MOOC atuais e seus relacionamentos, incluindo os novos elementos necessários para incorporar um SR. Na Seção de arquitetura, são citados seus elementos e relacionamentos. Em seguida, são descritas as funcionalidades, as ações dos elementos e a produção dos artefatos desses elementos para comporem o SR, incluindo a API e os algoritmos incorporados para a geração de *string* de busca e seus resultados, a métrica de popularidade composta entre os REAs e a execução do algoritmo de ordenação desses REAs, métrica de similaridade entre usuários e algoritmo de comparação de REAs. E, por fim, o algoritmo de recomendação e a forma de exibição dos REAs ao usuário.

3.2 MODELO CONCEITUAL

A concepção do modelo conceitual e a descrição dos seus principais elementos foram fundamentais para conceber um SR em ambientes MOOC, proposto por este trabalho. Ochoa e Duval (2006) acreditam que a busca por OA seria auxiliada pelas informações de CAM, pois elas adicionam uma forma interessante de classificar ou recomendar esses objetos aos usuários.

Segundo Ochoa e Duval (2006), os usuários interagem diretamente com diversos OAs ao longo do ciclo de vida de um MOOC. Por exemplo, coletores de CAM captam o *timestamp* do momento em que ocorreu o evento e as informações de interação do usuário com o MOOC para apoiar o cálculo de métricas a serem utilizadas posteriormente em uma ferramenta de gerenciamento de REAs. De acordo com Najjar et al. (2005), essas informações são armazenadas em um registro de ações. As fases do ciclo de vida dos OA são tomadas a partir de uma enumeração proposta por Collis e Strijiker (2004), já descritas anteriormente, sendo as seguintes ações: `criando, rotulando, ofertando, selecionando, usando` e `anotando.`

Ochoa e Duval (2006) utilizaram com êxito eventos de CAM em todas as fases do ciclo vida do OA, para um SR em um contexto semelhante ao aqui proposto No entanto, aqui, foram utilizadas as fases `selecionando` e `anotando`, fases necessárias para serem utilizadas no *OERecommender*, um SR com propósito de recomendar OA, aqui identificados como REAs, disponíveis em repositórios na Web. A fase `selecionando`**,** representa a forma com que o usuário realiza consultas e/ou obtem resultados automaticamente sobre quais REAs lhe são interessantes em um determinado momento. E na fase `anotando`, representa a forma com que o usuário avalia esse REA com uma nota de satisfação.

A Figura 23 apresenta o modelo conceitual proposto para representação dos elementos comuns nos ambientes MOOCs. A definição da estrutura do modelo foi concebida por meio de um mapeamento dos ambientes existentes atualmente.

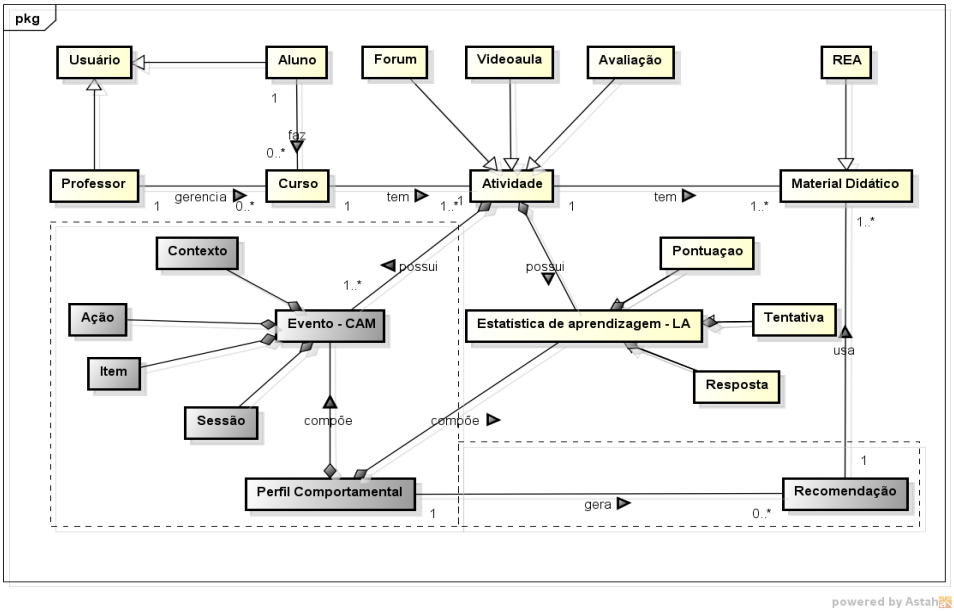


Figura 23 - Modelo Conceitual do *OERecommender*

A Figura 23 é composta por diversos componentes inter-relacionados, os componentes sombreados e destacados por uma linha pontilhada fazem parte do incremento para *OERecommender*.

Os elementos correspondentes ao *OERecommender* estão agregados como Evento - CAM associado ao conceito de atividade, compondo o conceito de Perfil Comportamental, para posteriormente representar como forma de armazenar as Recomendações. Os referidos elementos estão destacados em pontilhado na Figura 23, a saber:

- O elemento `Evento - CAM` – é composto de informações sobre as ações dos usuários conforme descrito a seguir (WOLPERS et al., 2007):
    - Item: agrupa metadados de atenção de um documento. Cada documento pode estar envolvido em diferentes ações como ler, ouvir e editar, em datas diferentes, por diferentes períodos e em diferentes contextos. O elemento item é composto de mais três elementos:
        - `Título:` captura um nome legível dado ao documento, quando o documento é criado ou editado. Este elemento é necessário para permitir que os usuários reconheçam facilmente o documento;
        - `Identificador:` representa um identificador global exclusivo para o documento dentro de um determinado contexto;
        - `Tipo:` tem o tipo de técnica *MIME* do documento. Por exemplo o HTML.

- Contexto: este elemento captura as informações que descrevem os ambientes em que o usuário pode interagir. Por exemplo, informações sobre um curso (disciplina e descrição) em que o usuário submeteu um documento, conforme ilustrado na Figura 11 da Seção 2.6.3. Dentro do elemento contexto, têm-se dois sub-elementos que são:
  - Tipo de valor: o elemento *valueType* contém referência a um elemento de uma ontologia ou taxonomia que descreve a disciplina que pode ser extraída a partir do elemento de valor;
  - Valor: o elemento *value* tem um texto livre, ele descreve os temas de outros documentos envolvidos, podendo expressar, por exemplo, o mesmo texto em mais de um idioma.
- Ação: o elemento *action* fornece informações sobre a ação em que o documento estava envolvido (por exemplo, se ele foi inserido no sistema local de arquivo ou em repositório digital):
  - Tipo de ação: o elemento *actionType* detém o tipo de ação (tarefa) em que o documento foi envolvido. Por exemplo, o URL http: //.../Actiontype/insert pode constituir uma referência para uma ação inserção de valores;
  - Entrada: o elemento *entry* grava dados dos registros de entrada relacionados a uma ação realizada.
- Sessão: o elemento sessão contém as informações que são necessárias para identificar a sessão de trabalho.
  - ID da Sessão: o elemento *sessionId* tem um identificador único para a sessão;
  - Endereço IP: o elemento *ipAddress* contém o endereço IP do computador do usuário;
  - Informações do usuário: o elemento *userInfo* coleta informações sobre o nome do usuário, endereço de e-mail e disciplina científica do usuário que executa a ação.
- Atividade: refere-se a toda e qualquer atividade no ambiente MOOC, por exemplo, vídeo aula, fórum, avaliações, etc. Esse elemento é fundamental para que ocorram as recomendações de forma eficiente, pois é onde ocorrem as maiores interações do usuário com o ambiente, consequentemente, onde são coletadas a maior quantidade de metadados.

O conceito Perfil Comportamental é composto de Eventos – CAM e Estatísticas de Aprendizagem – LA. Representa as ações realizadas juntamente com informações de aproveitamento pelo usuário no ambiente MOOC. Este perfil é utilizado para gerenciar as recomendações que o SR fará ao usuário. Por fim, o conceito Recomendação é responsável por armazenar as recomendações dos REAs aos usuários.

3.3 ARQUITETURA DO *OERECOMMENDER*

A arquitetura do *OERecommender* é apresentada na Figura 24. Sua concepção foi embasada em uma arquitetura desenvolvida por Govaerts et al., (2011), que utilizaram um sistema de pesquisa federada juntamente com o serviço de recomendação em um contexto semelhante ao aqui proposto. A arquitetura é composta por um processo com sete etapas, sendo a 1ª - geração de *string* de busca por meio de extração de palavras-chave; 2ª – *string* de busca por REAs em motor de buscador na Web utilizando API do *SeeOER*, ambas detalhadas na Seção 3.4.1; 3ª – resultados dessa *string* de busca, detalhados na Seção 3.4.2; 4ª – ordenação desses REAs por meio de algoritmo de ordenação e métrica de popularidade composta, descritas com mais detalhes na Seção 3.4.3; 5ª – algoritmo de comparação, comparando registros de CAM de REAs e usuários, e também calculando métricas de similaridade entre usuários, detalhados na Seção 3.4.4; 6ª – algoritmo de recomendação, realizando a recomendação dos REAs mais relevantes ao usuário por meio de *widget* utilizando o algoritmo *Slope One*, descritos na seção 3.4.5 e; 7ª – resultado reordenado das recomendações apresentado ao usuário por meio de *widget* e explicado na Seção 3.4.6.

Além de serem adicionadas etapas enumeradas, far-se-á a utilização de um cenário exemplo, como forma de melhorar o entendimento das etapas do *OERecommender.*

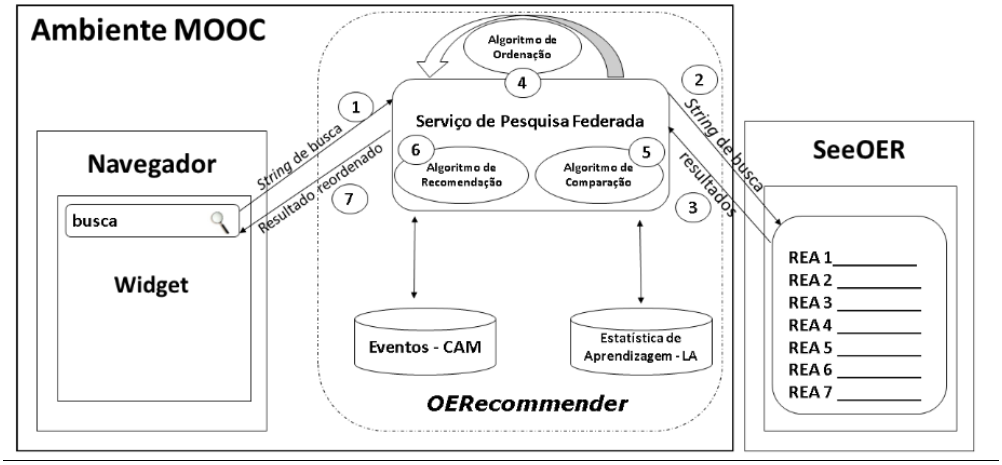


Figura 24 - Arquitetura do *OERecommender*

O cenário exemplo é composto por um MOOC sobre Introdução à Programação Python, em que o usuário Frodo está matriculado e começará suas aulas hoje. Frodo está utilizando um Sistema Operacional Linux e um navegador Mozila Firefox. O curso é ofertado para qualquer pessoa que tenha acesso à Internet e possui 5.400 alunos matriculados e ativos. A primeira vez que Frodo acessa o MOOC, aceita os termos de privacidade e instala a extensão *Slogger* em seu navegador (BAKER e BOAKES, 2008). Essa extensão permite a captação dos metadados do comportamento de navegação de Frodo na Web. Por meio dessa extensão é possível gravar fluxos XML de CAM do usuário e armazená-lo em uma base de dados Exist (EXIST, 2006).

A primeira atividade de Frodo é ler as informações sobre o curso, seu tempo de duração, que será de 5 semanas, entender como funciona o programa e adquirir os conhecimentos prévios para um bom aproveitamento. O formato do curso terá apresentação de palestras em vídeo aula, com avaliações *(quizzes)* entre as palestras. Ao final de cada semana será feita uma avaliação com a entrega de um miniprojeto por meio de *upload* no ambiente MOOC para avaliação por seus pares.

A primeira atividade é assistir a vídeo aula sobre "Introdução à lógica de Programação" e realizar sua primeira avaliação, um *quizz* que Frodo deve responder.

3.4 FUNCIONAMENTO DO *OERECOMMENDER*

O processo para recomendar REAs segue algumas etapas que foram enumerados na Figura 24. O *OERecommender* utiliza filtragem colaborativa para encontrar similaridade entre os itens, como nos tracionais sistemas de recomendação (RESNICK e VARIAN, 1997; ADOMAVICIUS e TUZHILIN, 2003; BARCELLOS et al., 2007; BURKE 2007; JANNACH et al., 2011) que seguem os métodos baseados em filtragem de conteúdo, colaborativa ou híbrida. O maior diferencial do *OERecommender* é fazer uso de informações contextuais para encontrar a similaridade entre usuários para posteriormente prever qual REA é relevante para ser recomendado ao usuário alvo. Usuário alvo é o termo adotado ao usuário que irá receber as recomendações. Essas informações contextuais são identificadas por semelhança entre atributos de instâncias de CAM, armazenadas em uma base de dados XML.

Reportando novamente ao exemplo, para iniciar o processo de recomendação a Frodo, o sistema captura os metadados e gera instâncias de CAM a partir da instalação da extensão *Slogger* (BAKER e BOAKES, 2008) em seu navegador. Por exemplo, ao interagir com as atividades do MOOC, com o tema Programação Python, Frodo produz instâncias que estão

sendo armazenadas em um banco de dados, uma instância decorrente dessa interação é demonstradas na Figura 25. É possível perceber que alguns atributos não têm valor atribuído, pois tratam-se de valores a serem atribuídos de acordo com a interação do usuário no ambiente.

```
<group>
    <title> Id_Frodo </title>
    <feed>
        <title> Firefox Mozila </title>
        <item>
            <title> Programação Python </title>
            <guid> ... </guid>
            <events>
                <event><action>
                            <actionType> ... </actionType>
                            <relatedData>
                                <entry>
                                    <name> Text
                                    <content> Introdução à Programação Python </content>
                                </entry>
                            </relatedData>
                        </action>
                        <dateTime> 2015-01-10T09:49:40</dateTime>
                        <session>
                            <sessionId> 10 </sessionId>
                            <ipAddress> 189.39.2.219 </ipAddress>
                            <userInfo>
                                <userName> Frodo </userName>
                            </userInfo>
                        </session>
                </event>
            </events>
        </item>
    </feed>
</group>
```

Figura 25 - Instância de CAM gerada e armazenada na base de dados XML.

3.4.1 A Geração de *String* de Busca

Na etapa 1, as *strings* de busca por REA são geradas por meio de palavras-chave, encontradas nas instâncias de CAM em que o Frodo está interagindo. Primeiro, o *OERecommender*, captura o valor do atributo *group.title* identificando unicamente este usuário e, em seguida, captura o *group.feed.item.title* identificando o título da atividade que ele está realizando. Este título é utilizado como termo para a busca na API do *SeeOER*, demonstradas na Etapa 2 da Figura 26.

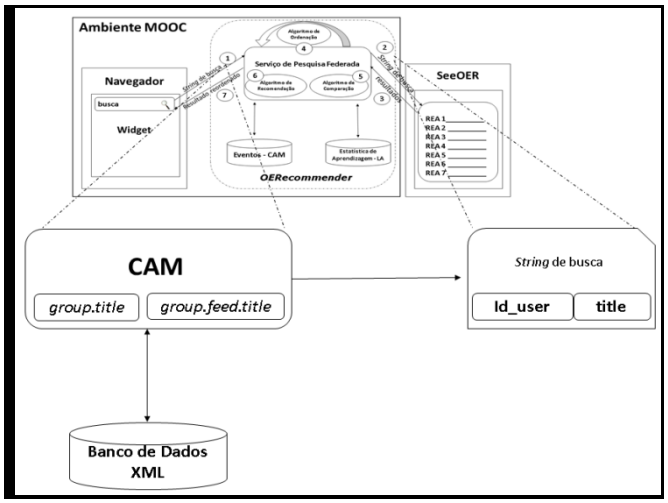


Figura 26 - Geração de *string* de busca

No cenário exemplo, Frodo está no ambiente MOOC realizando atividades relacionadas à "Introdução à Programação Python", portanto, o atributo de CAM *group.title* (*Id_Frodo*) é o identificador de usuário e o atributo *group.feed.item.title é Python*, Python será o termo para o *OERecommender* realizar a busca por meio do buscador *SeeOER*. Essa busca ocorre por meio de um serviço chamado de pesquisa federada utilizando a API do próprio *SeeOER* para realizar essa tarefa. Como já detalhado anteriormente, pesquisa federada permite realizar pesquisas em locais remotos (GOVAERTS et al., 2011).

3.4.2 O Resultado da Busca

A Figura 27 ilustra a 3ª Etapa, o retorno dos índices encontrados nos repositórios de REAs na Web, *o SeeOER* retorna uma ordenação pré-estabelecida do próprio buscador, trazendo somente o campo dos *URLs* de cada REA para posteriormente apresentar ao usuário.

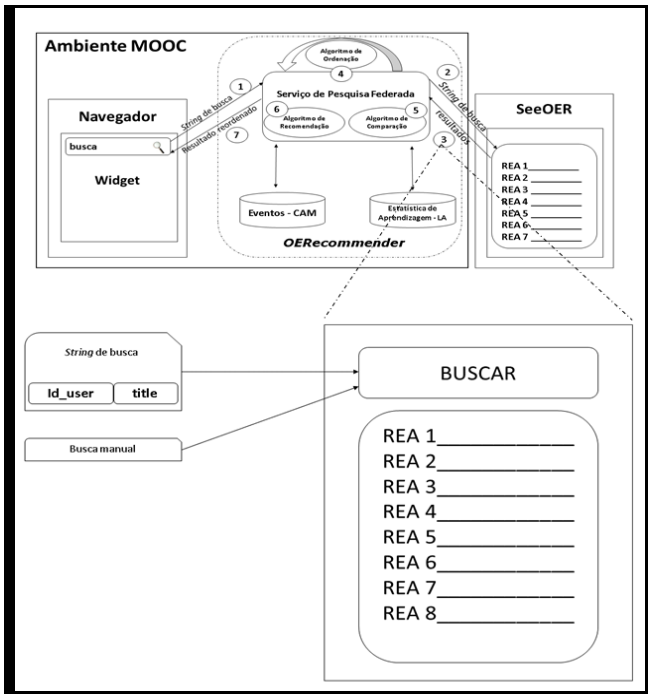


Figura 27 - REAs encontrados pelo buscador

O *SeeOER* é uma arquitetura desenvolvida para um mecanismo de busca na Web por REAs que visa a resolução de conflitos em nível de esquema e em nível de instância, oriundos do uso de diferentes padrões de metadados, repositórios e plataformas de REAs (GAZZOLA et al., 2014).

O *OERecommender* utiliza as estatísticas de aprendizagem para fazer uma primeira filtragem dos REAs. Por exemplo, após retornar os índices da API do *SeeOER* com os REAs encontrados sobre "Introdução à Programação Python", o *OERecommender* verificará o percentual de aproveitamento de Frodo até o momento, pois é interessante que Frodo receba recomendações de REAs já baixados e/ou avaliados por outros usuários que tiveram aproveitamento tão bom quanto ou melhor que ele.

3.4.3 O Algoritmo de Ordenação

A 4ª Etapa é a etapa da aplicação do algoritmo de ordenação dos resultados, a qual é dividida em dois passos. Primeiro passo, encontrar a similaridade entre os usuários por meio de grafos e métricas de classificação e, o segundo passo, a aplicação do algoritmo de ordenação.

- `Similaridade entre usuários`: o *OERecommender* primeiramente encontra a similaridade entre Frodo e os demais usuários que já utilizaram esses REAs. Para

isso, executa um passeio aleatório aplicado em um grafo bipartido, conforme mostra a Figura 28.

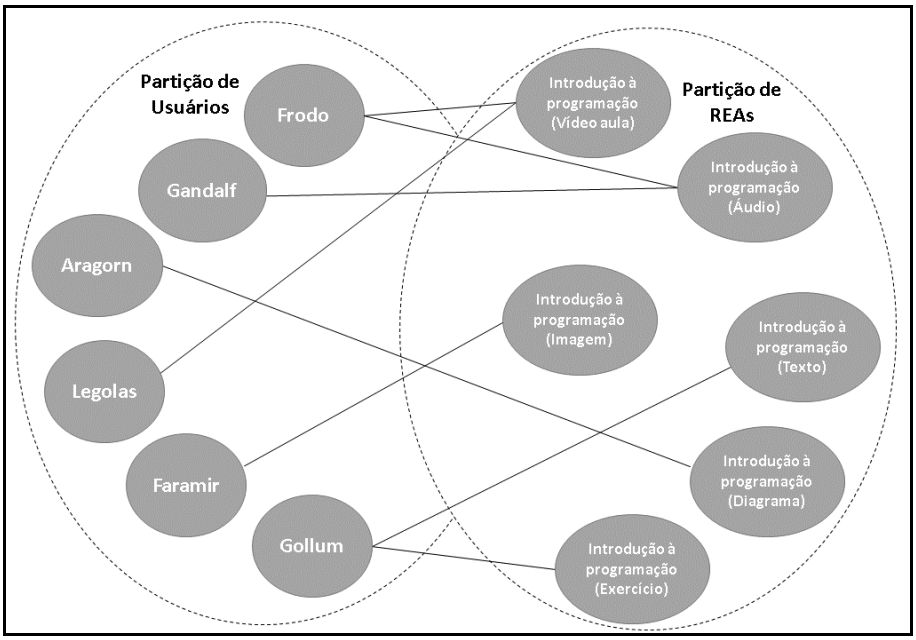


Figura 28 - Grafo bipartido entre Usuários e REAs

A Figura 28 ilustra uma partição de nós que representam os **Usuários**, destacados na esquerda do grafo. São os usuários, que baixaram e/ou avaliaram os REAs retornados na busca e, à direita, são representados os **REAs** retornados da busca, criando assim a correlação entre os usuários (OCHOA e DUVAL, 2006) por meio da dobradura de grafos, processo detalhado na Seção 2.7.2. Para encontrar essa correlação, os parâmetros avaliados nas instâncias de CAM da base de dados são: Primeiramente o parâmetro *grup.feed.item.guid* que identifica unicamente esse REA e, posteriormente, o parâmetro *group.title* que identifica o usuário que utilizou esse REA.

A correlação entre as partições representa a quantidade de vezes que o REA foi baixado e/ou avaliado pelos usuários. Para realizar esse cálculo o *OERecommender* usa duas métricas de classificação que são (OCHOA e DUVAL, 2006):

- `Rank de Popularidade (PR)`**:** Usando as informações da ação `Selecionando` armazenadas na base de CAM, pode-se obter o número de vezes que um REA foi baixado. Para calcular, conta-se apenas o número de ligações incidentes que cada nó da partição dos REAs tem nos nós da partição dos Usuários. Essa métrica é apenas para colocar os REAs mais

baixados em primeiro lugar na lista de resultados, métrica ilustrada na *Equação 7:*

$$\mathbf{PR(REA) = {}_{Numero}Downloads\ (REA)} \quad (7)$$

- `Rank Manual (MR)`**:** Usando as informações da ação `Anotando`, é possível obter o número de vezes que o REA foi avaliado. Essa avaliação segue uma escala (5,4,3,2,1), onde o valor "5" é que o usuário avaliou o REA como "Ótimo"; "4– Bom"; "3 – Regular"; "2 – Ruim" e; "1 – Péssimo". Esses valores são implícitos ao usuário, para o qual somente aparecerão uma escala de cinco estrelas, para marcar a quantidade de estrelas que ele avalia o REA. Uma avaliação de valor zero para nível de métrica, representa a ausência de avaliação daquele REA. Um grafo bipartido (partições **Usuários** e **REA**s) é criado. Essa métrica calcula os valores como pesos para encontrar relações entre usuários. O valor real para fim de cálculo da métrica é implícito ao usuário e é usado apenas para avaliar se a avaliação é um "voto" ótimo (peso 3), bom (2), regular (1), ruim (-1) ou péssimo (-2), pois diferentes usuários têm diferentes avaliações, métrica ilustrada na *Equação 8:*

$$\mathbf{MR\ (REA) = Avaliação_{(positiva)}\ (REA) - Avaliação_{(negativa)}\ (REA)} \quad (8)$$

Essas métricas calculam a relevância dos REAs. A soma das duas métricas pode ser calculada como a soma ponderada dos valores das métricas individuais. A ponderação adotada pelo *OERecommender* é peso de 20%(α) sobre a métrica que envolve o número de vezes que o REA foi baixado e 80%(β) sobre o peso da métrica dos pesos atribuídos pelos usuários para aquele REA. Isso deve-se ao fato de que nem sempre que o usuário baixa um REA, ele o utiliza. Já nos casos de avaliação da qualidade, normalmente fez uso do REA para posteriormente avaliá-lo. A combinação dessas duas métricas é chamada de `Métrica de Popularidade Composta (MPC)`, ilustrada na *Equação 9* (OCHOA e DUVAL, 2006):

$$\mathbf{MPC = (\alpha PR) + (\beta MR)} \quad (9)$$

Métodos para fazer essas estimativas são descritos em (RADLINSKI e JOACHIMS, 2005) e (FAN et al., 2004). Além disso, marcações manuais devem ser usadas com cuidado, pois a ação `Anotando` que exige interação explícita do usuário é uma informação opcional e pode não existir para todos os REAs envolvidos no cálculo.

- `Aplicação do Algoritmo de Ordenação`: ocorre após recuperar os REAs, calcula o valor dos pesos de relevância dos mesmos e encontra os usuários mais similares a Frodo, o usuário alvo. Assim, o sistema deve ordenar os REAs mais relevantes de acordo com o aproveitamento dos usuários semelhantes e os resultados da `MPC`. Para isso, o sistema utiliza uma forma de pós-filtragem, onde primeiramente ordena de forma decrescente os REAs, utilizando um algoritmo simples de ordenação de vetores ilustrado no Quadro 1, posteriormente essa ordenação será utilizada como vetor de entrada nas próximas etapas do *OERecommender*.

Quadro 1 - Algoritmo de Ordenação

```
1: inicialização
2:      Se N ⊂ B e k >= l faça
3:           vetor_REA[N]
4:           para I ← 1 até N faça
5:                eleito ← vetor_REA[N];
6:                j ← n-1;
7:                enquanto (( j >=0 ) e (eleito > vetor_REA [j] )) faça
8:                     vetor_REA[j + 1] := vetor_REA[j];
9:                     j ← j – 1;
10:               Fim_enquanto;
11:               Vetor_REA [j + 1] ← eleito;
12:          fim_para
13:     Senão B ← B ∩ N
14: fim
```

`Algoritmo de Ordenação`. Ordena o vetor REA por relevância.
`Entrada`: *vetor_REA* é o vetor com os índices dos REAs encontrados na busca, *i, j,* e *eleito* são variáveis utilizadas no algoritmo; *N* é um conjunto de REAs retornados da busca; *B* é o conjunto de registros de CAM desses REAs já armazenados no banco de dados; *k* é o percentual de aproveitamento de Frodo; *l* é o percentual de aproveitamento dos usuários que baixaram ou avaliaram esses mesmos REAs em outro momento e estão armazenados em registros de CAM na base de dados.

O `Algoritmo de Ordenação` verifica se existem registros de CAM desses índices já baixados e/ou avaliados em outro momento por Frodo ou por outros

usuários no banco de dados. Então verifica se os mesmos tiveram um aproveitamento igual ou superior a Frodo (linha 2). Se **SIM**, o vetor é ordenado de forma decrescente, calculando a métrica **MPC**. Sabendo-se quantas vezes o mesmo foi baixado e/ou avaliado é possível reordenar a primeira sequência retornada do motor de busca, colocando em ordem decrescente os REAs que tiveram maior valor no **MPC** no início do vetor (linha 3 a 10).

Caso os REAs retornados da busca **NUNCA** tiverem sido baixados e/ou avaliados, ou o percentual de aproveitamento desses usuários encontrados na base de dados forem inferiores ao de Frodo, o algoritmo não altera a sequência original dos índices retornados pelo motor de busca do repositório (linha 11). A Figura 29, apresenta um exemplo do funcionamento do Algoritmo de Ordenação.

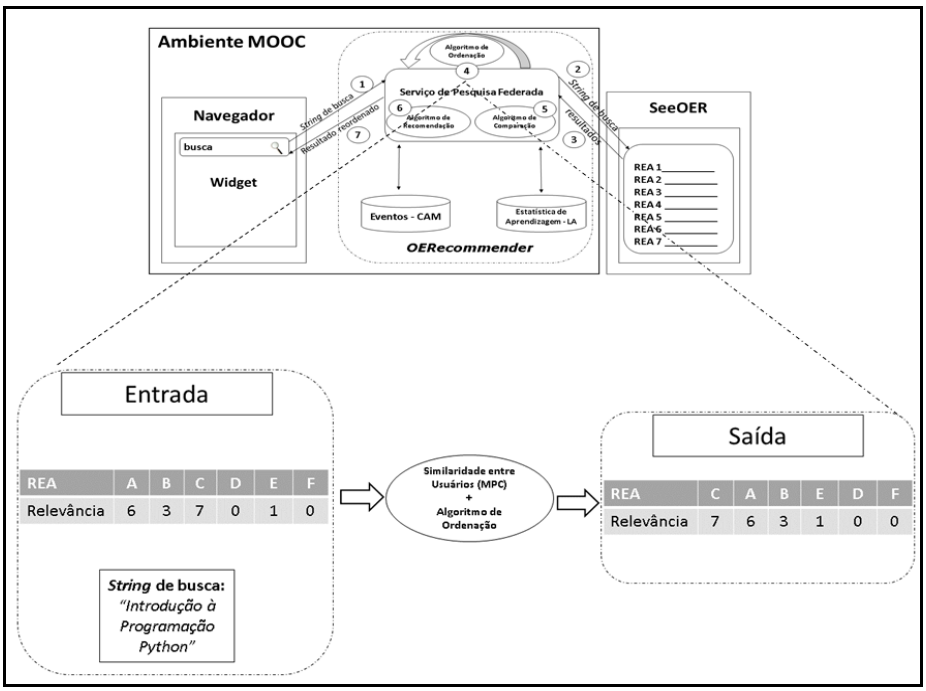


Figura 29 - Exemplo de aplicação do Algoritmo de Ordenação

Com a Figura 29 é possível visualizar a entrada do algoritmo, que contém os termos de busca e a saída do algoritmo após serem reordenados por meio da **MPC** e pelo Algoritmo de Ordenação.

3.4.4 Algoritmo de Comparação

A 5ª Etapa é a etapa da aplicação do algoritmo de comparação. Isso ocorre após o vetor de REAs estar ordenado por relevância, assim, compara as informações contextuais por meio dos valores dos atributos de instâncias de CAM em que Frodo está interagindo. Para isso, o algoritmo percorre o vetor de REAs já ordenado por relevância e faz comparações. Compara todos os registros de CAM do banco de dados que contém os REAs retornados do algoritmo de ordenação. Após encontrar os registros de CAM, compara todos os atributos dos usuários que baixaram e/ou avaliaram esses REAs e assim encontrará os REAs que possuem maior similaridade dos demais usuários com Frodo, para reordenar novamente os resultados.

O motivo para reordenar novamente os REAs está no fato de que, para se conseguir uma comparação de contexto, é necessária a comparação dos valores dos atributos de instâncias de CAM entre esses usuários. Fazendo isso, seria evitado, por exemplo, de recomendar REAs somente de um usuário que tenha maior similaridade, mas não significando necessariamente que tais REAs estejam no contexto mais similar ao do usuário alvo.

`Algoritmo de Comparação`: Compara os atributos de registros de CAM.
`Entrada`: *vetor_REA* é o vetor que retornou os REAs encontrados na *string* de busca, *Base_A* é o conjunto de todos os valores dos atributos dos registros de CAM de outros usuários na base de dados; Base_*C* é o conjunto de todos os valores dos atributos de CAM de Frodo; *P* é quantidade de atributos iguais entre os registros da base com os REAs de busca.

O algoritmo ilustrado no Quadro 2, reordena o vetor, após fazer uma interseção de todos os atributos dos registros de CAM dos usuários que já baixaram e/ou avaliaram esses REAs com os atributos dos registros de CAM de Frodo (linha2), retorna o tamanho dessa interseção (linha 3).

Quadro 2 - Algoritmo de Comparação

```
1: Tamanho_intersecao (vetor_REA, Base_C)
2:        P ← Base_A ∩ Base_C
3:        retorna P.tamanho
```

Na Figura 30, observa-se o um exemplo de aplicação do Algoritmo de Comparação, comparando os REAs ordenados vindos do algoritmo de ordenação com as informações de contexto dos registros de CAM de outros usuários.

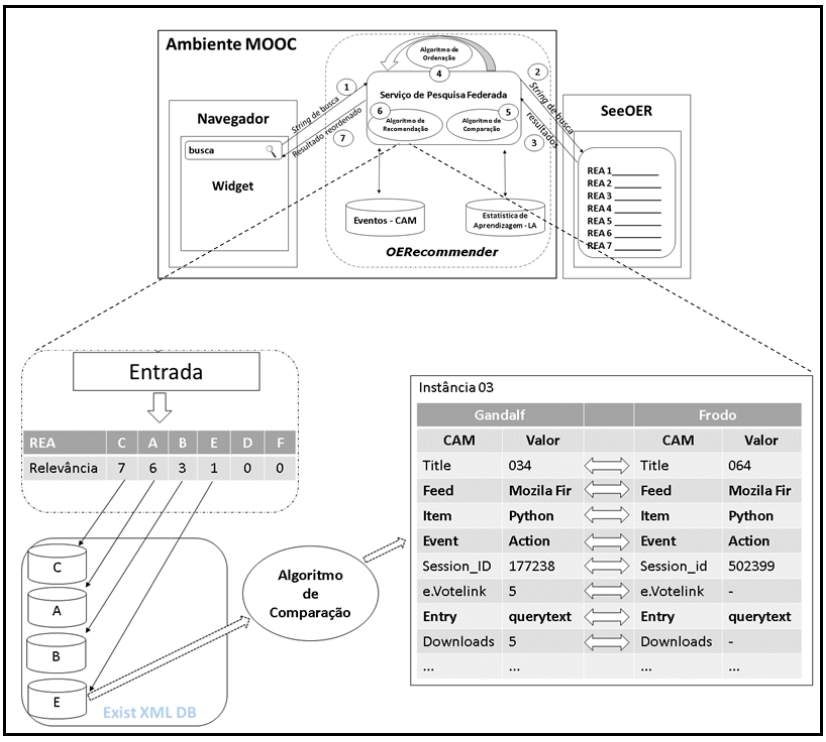


Figura 30 - Exemplo de funcionamento Algoritmo de Comparação

Na Figura 30, é mostrada a aplicação do algoritmo de comparação entre o usuário Gandalf e Frodo com alguns REAs (C, A, B, E, D e F) já ordenados pelo algoritmo de ordenação. As informações de CAM nos registros dos usuários estão armazenadas no banco de dados XML. O algoritmo de comparação compara todas as instâncias de CAM referentes ao mesmo REA, procurando igualdades entre valores dos atributos. Os REAs do vetor comparam o REA-E entre o usuário Gandalf (que já o baixou e/ou avaliou) e Frodo (que possivelmente irá baixar e/ou avaliar) para encontrar possíveis semelhanças de contexto e, posteriormente, mensurar o peso das semelhanças entre os dois usuários.

Cada similaridade encontrada entre os atributos da instância de Frodo e dos demais usuários, o algoritmo de comparação incrementa no peso da correlação um valor "+1" entre os usuários. A Figura 31 ilustra que foram encontradas "4" similaridades entre os usuários por meio dos atributos de CAM: *feed, item, event, entry*. Isso quer dizer que o peso da similaridade entre eles é "4" e a distância é igual a "1", pois como é ilustrado nessa figura, utiliza-se a técnica de dobradura de grafos, descrita na Seção 2.7.2 (OCHOA e DUVAL, 2006).

O cálculo da métrica de similaridade entre os usuários é feito conforme a *Equação 10*:

$$Métrica\ de\ Similaridade\ (MS) = \sum \frac{(Peso)}{(Distância)} \qquad (10)$$

Quanto mais próximo a "0" for a métrica de similaridade, menos similares são os dois usuários. No exemplo da Figura 31, foram encontrados quatro valores de atributos que têm valores similares (peso) e distância de valor "1" entre Gandalf e Frodo em uma instância, resultando em um valor de similaridade entre perfis igual a **"4".**

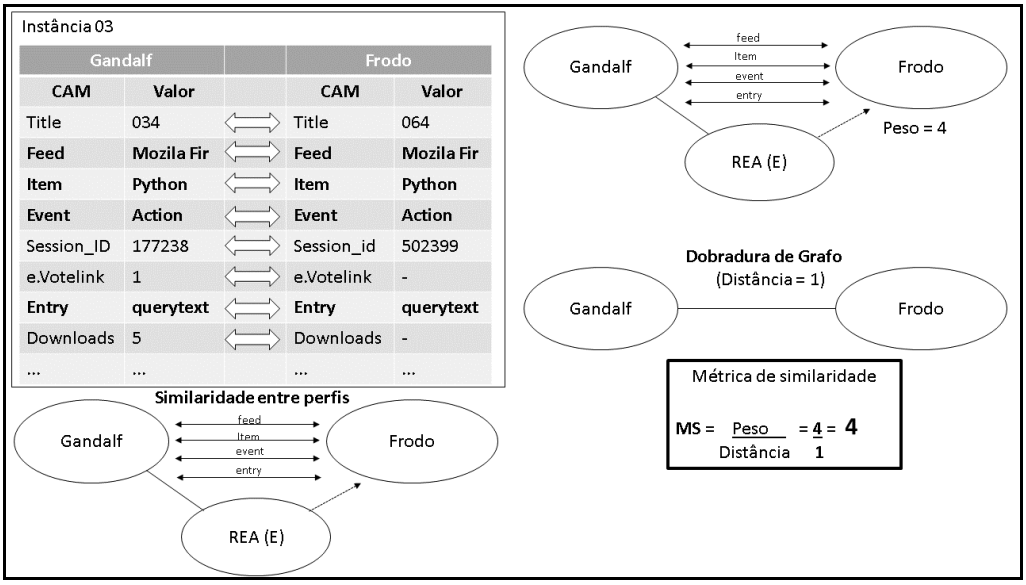


Figura 31 - Exemplo de similaridade entre perfis

A Figura 32 ilustra uma comparação entre dois REAs retornados da busca do algoritmo de ordenação. A comparação é entre o REA-E e o REA-A. Para o REA-E foi encontrado somente em uma única instância, mas para o REA-A foi encontrado em seis instâncias que já o baixaram e/ou o avaliaram. Assim, é possível observar que ao contrário do cálculo da métrica para o REA-E, o REA-A apresenta seis cálculos para encontrar o valor médio dos pesos, já que todos os valores dos atributos têm o mesmo peso ("1"). Assim, percebe-se que um maior número de instâncias foi sinônimo de uma maior similaridade, pois o REA-E tem uma similaridade entre perfis igual a "**4"**. Mas para o REA-A obteve-se um valor de similaridade de "**4,16".**

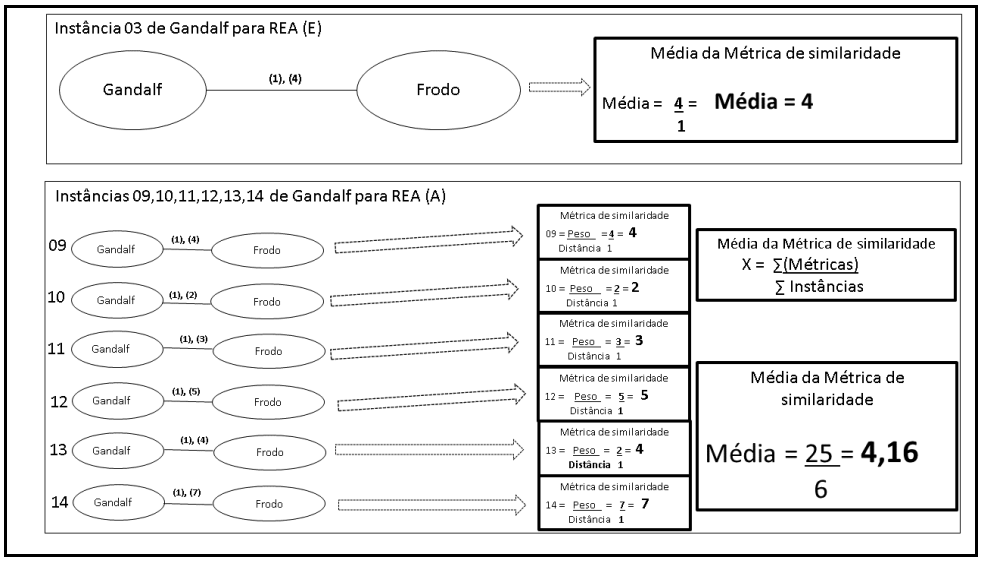


Figura 32 - Comparação entre instâncias de Gandalf e Frodo para o REA-A e REA-E

Na Figura 33, observam-se as métricas comparativas entre o REA-E e o REA-B, nas quais é visível que o número de instâncias não foi sinônimo de uma maior similaridade entre os perfis, já que no REA-B tem três instâncias e a média da similaridade é **"2,66"** enquanto no REA-E foi de **"4".**

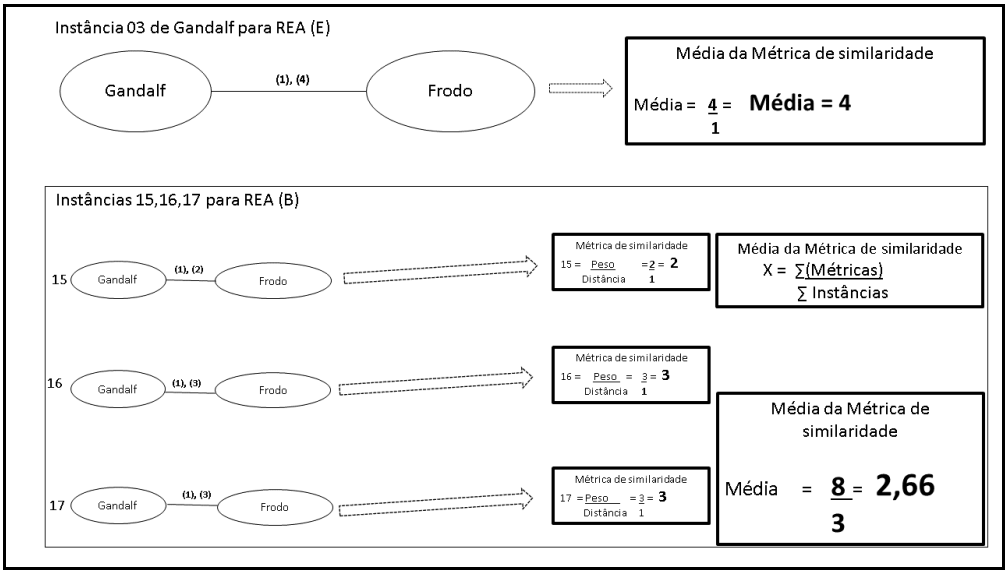


Figura 33 - Comparação entre instâncias de Gandalf e Frodo para o REA-B e REA-E

A Figura 34 mostra o cálculo da métrica para o REA-C e a média da métrica de similaridade de **"5,14",** provando que esse REA é o que tem maior relevância dentre todos, pois tem uma maior quantidade de atributos similares entre Gandalf e Frodo.

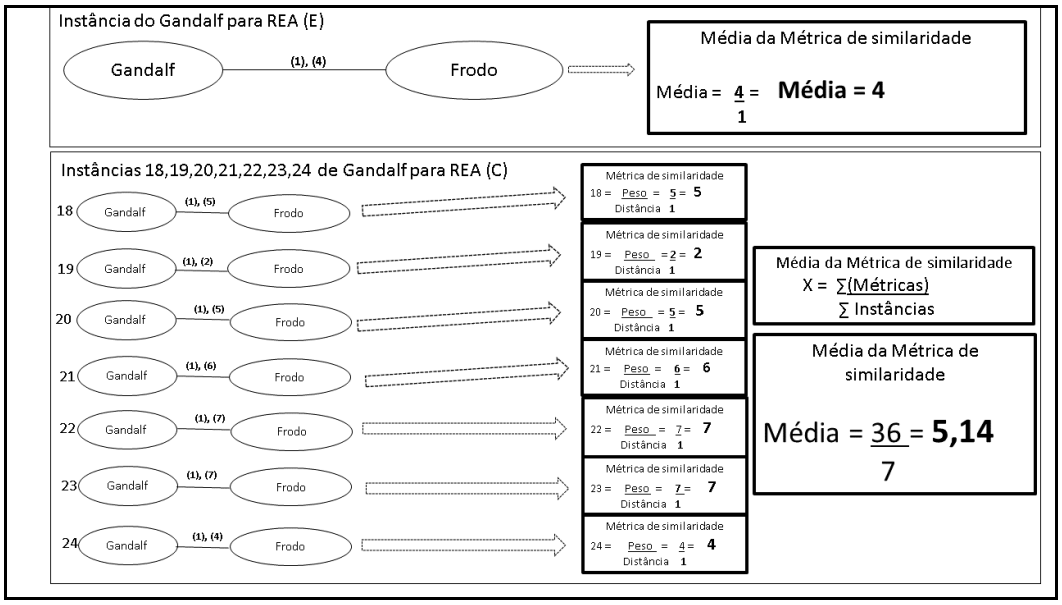


Figura 34 - Comparação entre instâncias de Gandalf e Frodo para o REA-C e REA-E

Na Figura 35, é possível visualizar o algoritmo de comparação reordenando os REAs após calcular a similaridade entre os perfis dos usuários. Percebe-se que houve a troca de posição entre o REA-B com o REA-E após retorno do algoritmo de ordenação.

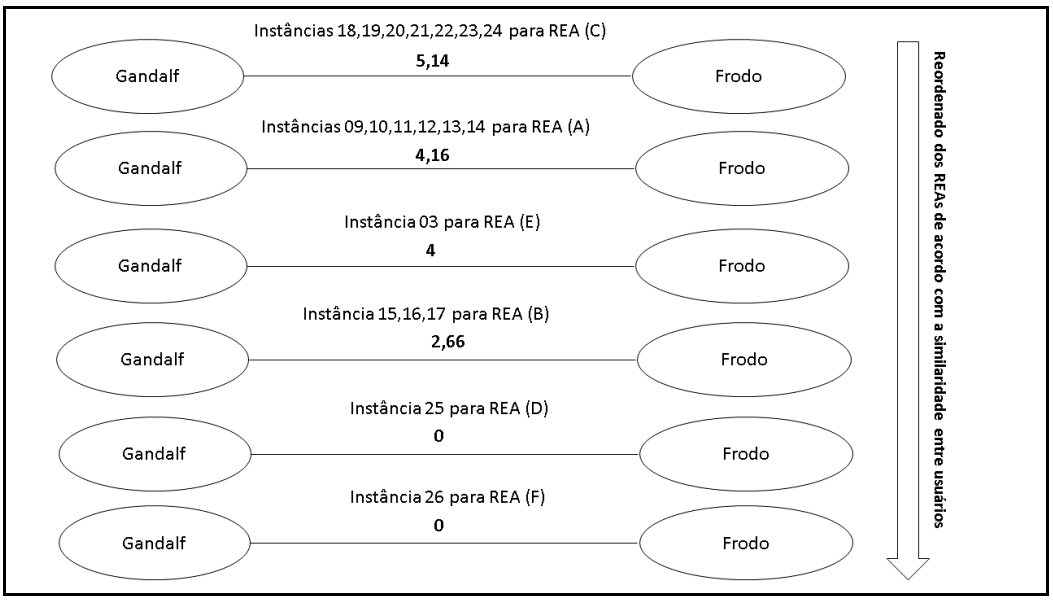


Figura 35 - Reordenação dos REAs de acordo com similaridade entre Gandalf e Frodo, analisadas por meio de instâncias de CAM

A Figura 36, mostra como o vetor dos REAs é novamente reordenado após os REAs serem submetidos ao algoritmo de comparação.

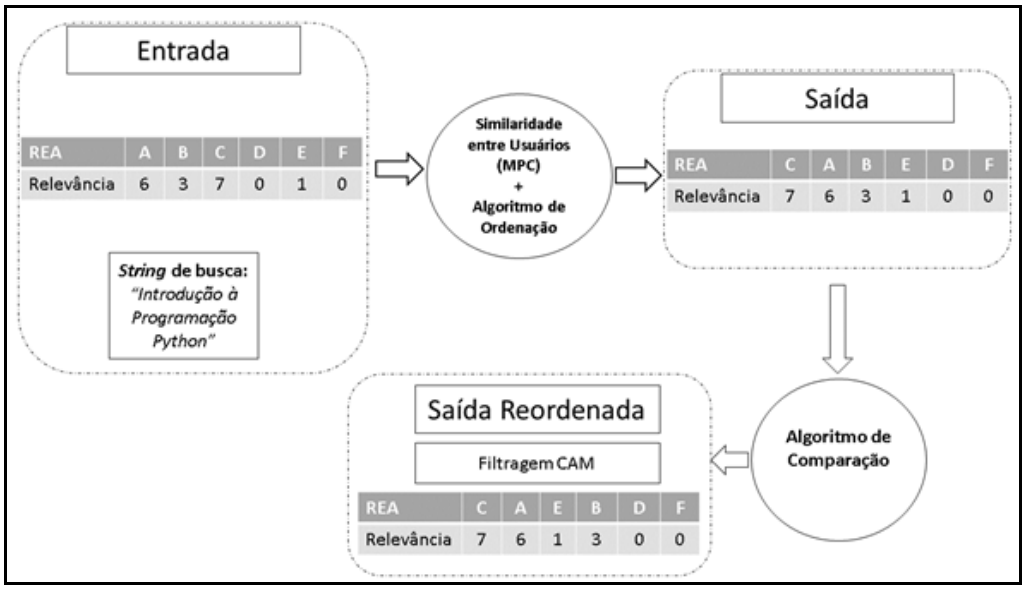


Figura 36 - Nova saída reordenada após Algoritmo de Comparação

Frodo ainda não recebeu as recomendações, os processos realizados até o momento encontraram a similaridade entre os usuários e reordenaram os REAs de acordo com sua relevância. Para que as recomendações sejam apresentados a Frodo, é necessária a submissão dos dados à um algoritmo de recomendação, para que seja possível fazer uma recomendação por meio de similaridade entre itens. Nem sempre os usuários que são mais similares têm as mesmas preferências com relação aos itens.

3.4.5 Algoritmo de Recomendação

A 6ª Etapa é composta de um algoritmo de recomendação, neste projeto, optou-se pela escolha do algoritmo *Slope One*. A escolha foi devido ao método de recomendação ser fácil de implementar, com teoria simples, apresentando bons resultados práticos, sendo altamente escalável (LEMIRE e MACLACHLAN, 2005). Suas predições são calculadas a partir da comparação entre avaliações de usuários a certos itens.

O algoritmo opera supondo que um usuário tenha dado notas não binárias a itens. Essas notas são colocadas em uma matriz de **Usuários x Itens**, de tal forma que cada linha corresponda às notas de um usuário *j* a *N* itens. Se um usuário *j* não tiver dado notas a um item *i*, o elemento $x_{i,j}$ fica igual a 0. A Figura 37 representa a matriz de um conjunto de notas:

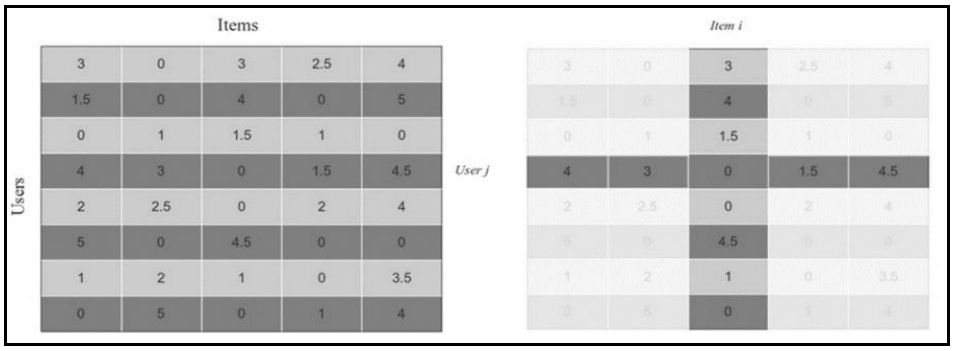


Figura 37 - Matriz dos Itens x Usuários (LEMIRE e MACLACHLAN, 2005)

Observando a matriz de notas vê-se que uma linha *j* da matriz representa as notas dadas por um usuário *j* a todos os itens no espaço definido. Uma coluna *i r*epresenta as notas recebidas pelo item *i* dos diferentes usuários. A partir dessa matriz, pode-se obter relações entre os dados. É possível gerar uma interpolação matemática e predizer qual seria a nota dada por um usuário *j* ao item *i* que ele ainda não avaliou (LEMIRE e MACLACHLAN, 2005).

Diferentemente de outras abordagens colaborativas, o *Slope One* cria uma relação linear entre os dados, usando a seguinte fórmula $f(x) = x + b$, onde a variável x representa as avaliações feitas pelo utilizador, enquanto b representa o desvio.

Exemplificando, um usuário *A* que deu nota 2 para um item *i*, e nota 4 a um item *j* e supondo ainda que exista um usuário *B* que deu nota 3 para o item *i*, por meio do *Slope One*, calcular-se-ia a predição da nota que o usuário *B* daria ao item *j* da seguinte forma, conforme ilustrado na *Equação 11*:

(Nota de A a item i - Nota de A a item j) = (Nota de B a item i – Predição de B em j) Logo:

$$(2-4) = (3-?) \qquad (11)$$
$$? = 2 + 3$$
$$? = 5$$

De acordo com a predição do algoritmo, o usuário *B* daria uma nota 5 ao item *j*.

Para análise de mais dados, obter-se-ia a média das diferenças entra as notas dos usuários. A equação geral para cálculo das predições segue descrita na *Equação 12*:

$$\frac{P(A,i) = (R(A.j) + Diff(i,j)) + (R(A.k) + Diff(i,k)) + \ldots + (R(A.z) + Diff(i,z))}{N} \qquad (12)$$

Onde *Diff (i,j)* é a média das diferenças de avaliações entre os itens *i* e *j* para os outros usuários, *R (A, j)* é quanto o usuário *A* deu de nota ao item *j*, supondo que haja *N* itens e que os itens variem de *i* a z .

Importante salientar que o tamanho da vizinhança considerada é tipicamente limitado a um tamanho específico, portanto, nem todos os vizinhos são levados em consideração para a previsão. No contexto específico deste trabalho, serão os 4 vizinhos com maior peso de similaridade e contextos similares com Frodo, o usuário alvo.

Um pseudocódigo do algoritmo *Slope One* é demonstrado no Quadro 3. É separado em três passos:

Quadro 3 - Algoritmo de Recomendação

Passo 1: Leitura das entradas

Leitura das informações contidas no arquivo de entrada:

Para file_input: nome do arquivo de dados.

Retorna: dicionário com as informações do arquivo.

A chave de cada elemento é o id do usuário e o valor é um novo dicionário com os REAs (chaves) e as avaliações (valores) para cada REA.

Passo 2: Pré-Processamento das matrizes

Pré-processamento e cálculo da matriz de diferenças.

Para Informações_Usuários: dicionário de informações retornado pelo método 'read_input()'.

Passo 3: Predição:

Predição a partir da matriz de diferença.

Para user_prefs: Um dicionário com as avaliações de alguns REAs.

Retorna: dicionário com as 'predições' para a(s) avaliação(ões) do(s) outro(s) REA(s) não informado(s) no parâmetro user_prefs.

3.4.6 Reordenação dos Resultados

Após a realização das reordenações dos REAs, o *OERecommender* finaliza seu processo predizendo à Frodo por meio de *widget*, os cinco REAs mais interessantes. Esse é a 7ª e última etapa do *OERecommender*, e ilustrado na Figura 38.

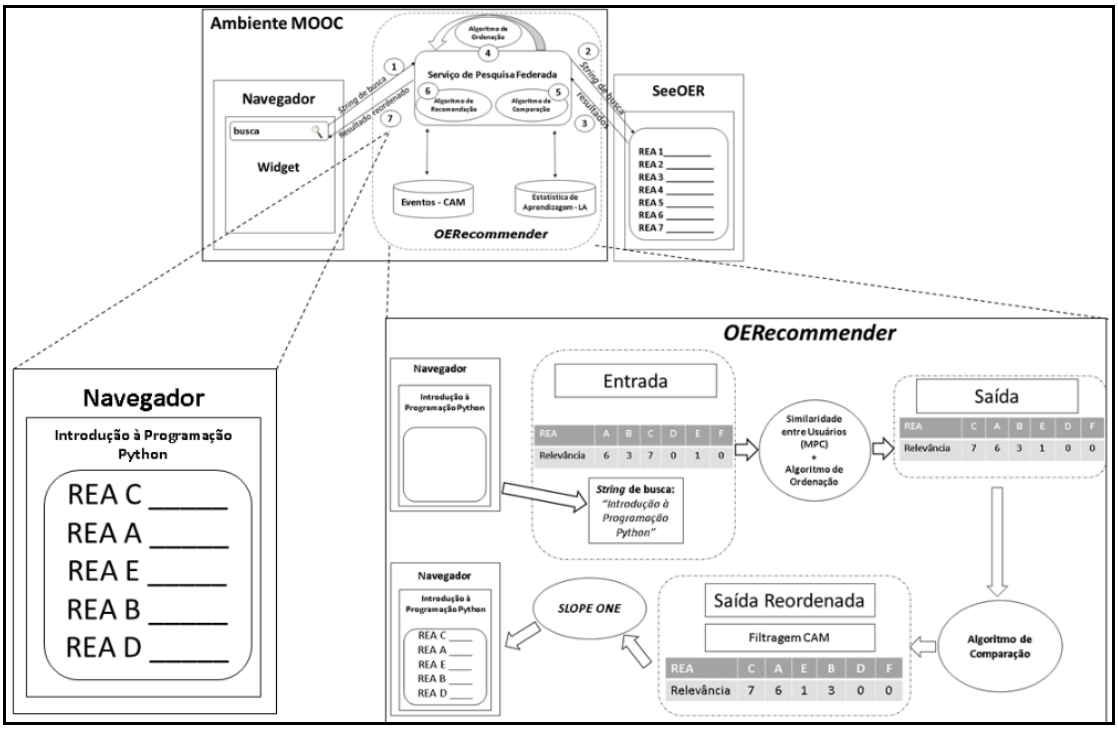


Figura 38 - Predição dos 5 REAs mais relevantes sendo exibidos à Frodo, após processo completo do *OERecommender*

3.5 CONSIDERAÇÕES FINAIS

O projeto do *OERecommender* utiliza técnicas de filtragem colaborativa baseada em modelos. Essa técnica utiliza o conjunto de avaliações dos usuários para aprender um modelo, que é usado então para fazer as predições e recomendações. Faz uso de métricas não supervisionadas para encontrar similaridades e recomendar REAs aos usuários. Porém, essas métricas não supervisionadas ocasionam um problema, ao iniciar um novo curso em ambientes MOOC, novos usuários irão sofrer com recomendações imprecisas, dado que ainda não foram realizadas avaliações dos REAs e a matriz estará esparsa. Assim, o *OERecommender* terá como parâmetro somente as informações contextuais dos usuários para encontrar as similaridades entre os mesmos e realizar as recomendações.

Esse problema de matriz esparsa é caracterizado na literatura como o problema da "partida fria de usuário" (SCHEIN et al., 2002), pois o algoritmo precisa treinar e, com o passar do tempo, a tendência é que as recomendações sejam cada vez mais eficazes.

A opção pela escolha da técnica baseada em modelo deve-se ao cenário atual dos MOOCs, que concentram grande quantidade de usuários. Caso utilizasse a técnica baseada em memória, sua implementação tornar-se-ia muito "caro" computacionalmente e considerando

que a atualização das matrizes para encontrar similaridades ocorreria a cada novo evento de CAM realizado. Esse foi o principal motivo pela escolha do método baseado em modelos.

4. AVALIAÇÃO QUALITATIVA DO *OERECOMMENDER*

4.1 CONSIDERAÇÕES INICIAIS

A avaliação do *OERecommender* foi realizada por meio da prototipação de cenários fictícios em que os algoritmos envolvidos foram executados. A discussão qualitativa dos resultados é realizada com base em parâmetros característicos de MOOC e de SR.

4.2 CENÁRIO PARA AVALIAÇÃO

O cenário fictício utilizado para avaliação é de um MOOC a ser disponibilizado no ambiente do Google Course Builder (code.google.com/p/course-builder). Algumas telas foram criadas para simular o ambiente real.

A dinâmica em que o curso foi conduzido foi no formato xMOOC. O curso é de *Programming* e tem por objetivo o ensino de programação de computadores para alunos iniciantes, não há pré-requisito para matricular-se no curso e está sendo disponibilizado de forma gratuita.

As etapas utilizadas na simulação dos cenários foram: as etapas do funcionamento do *OERecommender* e a tabulação dos dados obtidos durante a simulação, representados por meio de tabelas. Nesta simulação, para caracterizar os usuários foram utilizados nomes fictícios, assim facilitando o entendimento do leitor nas etapas da recomendação dos REA.

Os pseudocódigos dos algoritmos foram descritos nas Seções 3.4.3, 3.4.4 e 3.4.5 e a implementação do algoritmo de recomendação foi realizada em linguagem Python.

Inicialmente foram concebidos dois cenários fictícios descritos nas seções seguintes. O primeiro cenário é destinado ao usuário Frodo que fará o MOOC de *Programming* usando e classificando REA que o *OERecommender* lhe recomendará. O segundo cenário é do usuário Gandalf que recebe recomendações de REA que foram classificados por Frodo no primeiro cenário. Cada cenário foi dividido em vários fragmentos e os seus relacionamentos são ilustrados na Figura 39.

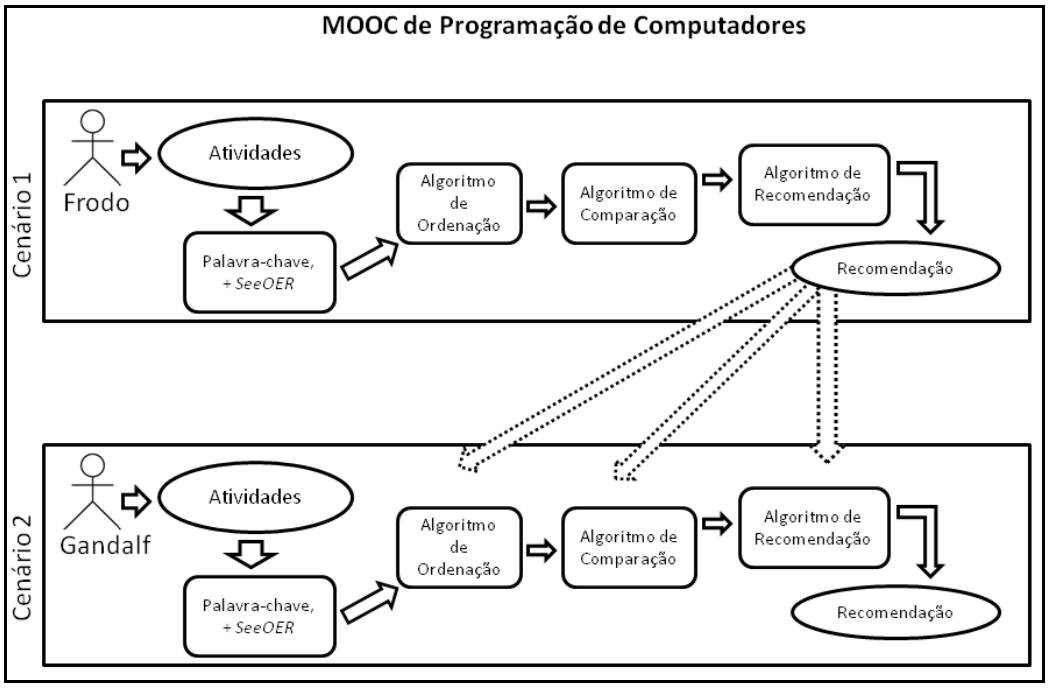


Figura 39 - Cenários para explicação do *OERecommender*

Na Figura 39, as elipses representam ações de interação explícitas do usuário com o MOOC. Os retângulos representam a execução dos algoritmos, além disso, a sequência entre os componentes de cada cenário é ilustrado por meio de setas contínuas e as ligações de inferência entre componentes dos diferentes cenários são representadas pelas setas tracejadas.

4.3 PRIMEIRO CENÁRIO

O usuário Frodo está matriculado no MOOC de *Programming* e no decorrer de suas interações com o curso, receberá as recomendações e classificará alguns dos REA. De forma hipotética, este cenário mostra como funciona a recomendação de REA em uma atividade intermediária, já que em recomendações iniciais tem-se o problema da "partida fria", já detalhada na Seção 3.5.

O processo de recomendação consiste de sete etapas que são sumariamente descritas a seguir. Essas etapas envolvem quatro algoritmos que foram descritos nas Seções 3.4.3 a 3.4.5 respectivamente, a saber: análise de similaridade entre os usuários; ordenação de REA; comparação de atributos de registros de CAM e; o algoritmo de recomendação *Slope One*.

4.3.1 Etapas 1, 2 e 3 - Geração da *String* e Realização de Busca Automática e Resultados do *SeeOER*

Nas etapas iniciais, a *string* de busca por REA é gerada por meio de palavras-chave contidas nas instâncias de CAM com as quais o usuário está interagindo. Inicialmente, o *OERecommender* captura o valor do atributo *group.title* identificando unicamente esse usuário, que no cenário deste trabalho é Frodo e terá o ID=14. Em seguida, captura o *group.feed.title* identificando o título da atividade que está realizando, que é "*programming"*. Esses atributos são oriundos de eventos de CAM e armazenados em um banco de dados XML (WOLPERS et al., 2007; NAJJAR et al., 2005).

Após a realização dessa busca, o *SeeOER* devolve ao *OERecommender* o conjunto de índices encontrados, detalhados na Seção 3.4.2. Para facilitar a compreensão, foi utilizada a palavra-chave aqui especificada para realização de busca diretamente no *SeeOER* no formato *php*. O URL utilizado para fazer a busca já incluindo a palavra chave foi:

*http://usp.seeoer.com/?q=**programming**&wt=php&indent=true*

Os resultados retornados pelo *SeeOER* tiveram 1.113 índices indexados, dos quais foram tabulados, conforme ilustrado na Tabela 2. Foram utilizados os cinco primeiros índices do total dos resultados: *1) SI / Coursera – Programming for Everybody; 2) Everything maths & Science; 3) Introduction to Programming; 4) Introduction to programming in Java; 5) J2EE Tutorial*, pois a apresentação do total dos resultados nesse tipo de formato seria repetitiva. Nesta tabela, foram tabulados os seguintes resultados: a identificação do usuário que efetuou a busca, o REA que foi encontrado, a especificação do REA e o repositório onde foi encontrado.

Tabela 2 - REAs selecionados após busca para exemplificação

Usuários	REA	Especificação do REA	Repositório REA
Frodo	*SI / Coursera - Programming for Everybody*	**Tipo de material:** Exercícios, OA, Palestras, etc. **Formato:** html **Licença:** Creative Commons 4.0 Attribution **Publicado em:** 24 fev, 2014 **Revisado:** -	Open.Michigan
	Everything maths & Science	**Tipo de material:** Texto, Palestras, Exercícios **Formato:** html / Text **Licença:** Creative Commons **Publicado em: -** **Revisado: -**	Siyavula
	Introduction to Programming	**Tipo de material:** texto **Formato:** html **Licença:** Creative Commons 3.0 Attribution **Publicado em:** 16 Mar, 2010 **Revisado:** 19 Abr, 2010	Cnx.org
	Introduction to programming in Java	**Tipo de material:** Tutorial **Formato:** HTML / Texto **Licença:** Creative Commons Attribution Noncommercial-ShareAlike 3.0 **Publicado em:** 18 Jul, 2002 **Revisado:** 12, Dec, 2014	MERLOT
	J2EE Tutorial	**Tipo de material:** Tutorial **Formato:** HTML / Texto **Licença:** Creative Commons **Publicado em:** 30 dec, 2001 **Revisado:** 30 nov, 2011	MERLOT

4.3.2 Etapa 4 – Ordenação dos Resultados

O algoritmo de ordenação dos resultados recebe os índices oriundos do *SeeOER* e faz a reordenação dos índices de acordo com o nível de relevância, o qual foi descrito com detalhes na Seção 3.4.3.

Para o cenário utilizado neste trabalho, assume-se que esses REA já foram avaliados por diversos outros usuários, dentre eles, um em específico o usuário Faramir e as informações já estão armazenadas no banco de dados.

Nesse caso, há uma similaridade entre o usuário Faramir e o usuário Frodo que é o usuário alvo do cenário, ela é obtida por meio da técnica de dobradura de grafos, detalhada na Seção 3.4.3. Antes mesmo de Frodo realizar as atividades do MOOC de *Programming*, Faramir já havia realizado algumas atividades deste mesmo MOOC e recebido recomendações do *OERecommender*, nas quais havia feito *download* de alguns REAs e avaliado um deles, conforme ilustrado na Figura 40, ilustrando também a similaridade entre Frodo e Faramir.

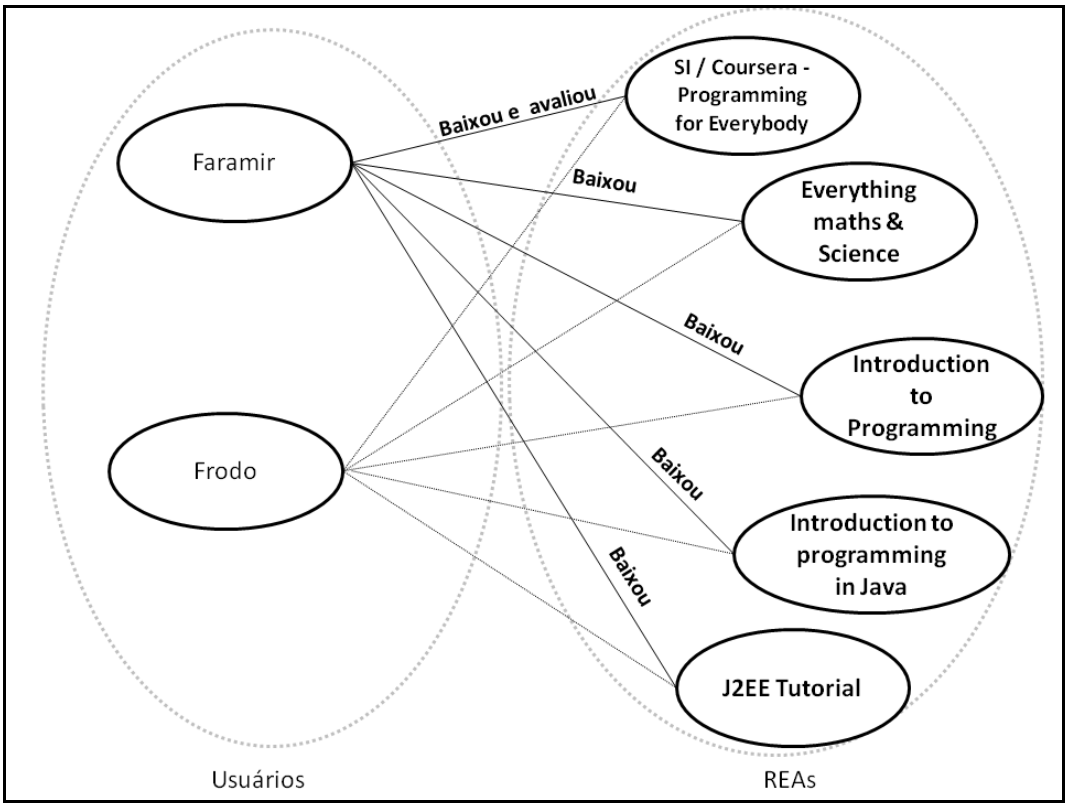


Figura 40 - Grafo bipartido representando similaridade entre usuários Faramir e Frodo

No grafo ilustrado na Figura 40, estão armazenadas no banco de dados, instâncias com o atributo *item.event.actionType* com valor da ação = `selecionando`, com isso, iniciam-se relações entre o usuário e os REAs. Na Tabela 3 é possível visualizar o valor dado aos REAs pelo usuário Faramir após baixá-los e avaliá-los.

Tabela 3 - Valores atribuídos pelo usuário Faramir aos REAs gravados no banco de dados.

Usuários	REA	Parâmetro	Ação	Valor atribuído ao parâmetro
Faramir	*SI / Coursera - Programming for Everybody*	Event.action.actionType	Selecionando	1
		Event.voteLink	Anotando	3
	Everything maths & Science	Event.action.actionType	Selecionando	1
	Introduction to Programming	Event.action.actionType	Selecionando	1
	Introduction to programming in Java	Event.action.actionType	Selecionando	1
	J2EE Tutorial	Event.action.actionType	Selecionando	1

Os parâmetros mostrados na Tabela 3 são analisados pelo algoritmo de ordenação, que calcula as métricas do **MPC** para cada REAs com os valores atribuídos nesses parâmetros para reordená-los. Essas métricas foram detalhadas nas Equações 7, 8 e 9 da Seção 3.4.3. O cálculo da **MPC** para o REA: *SI / Coursera - Programming for Everybody*, é realizado e exibido na Equação 13:

$$\begin{aligned} \textbf{MPC}\ (\textit{SI / ... for Everybody}) &= \textbf{(}\alpha\textbf{PR)} + \textbf{(}\beta\textbf{MR)} \\ \textbf{MPC}\ (\textit{SI / ... for Everybody}) &= \textbf{(}1*0{,}2\textbf{)} + \textbf{(}3*0{,}8\textbf{)} \\ \textbf{MPC}\ (\textit{SI / ... for Everybody}) &= \textbf{2,6} \end{aligned} \qquad (13)$$

Para o restante dos REA usados neste cenário, o cálculo do **MPC** é igual a 0,2 já que foram baixados uma única vez, mas não foram avaliados pelo usuário Faramir, como é possível visualizar na Figura 41.

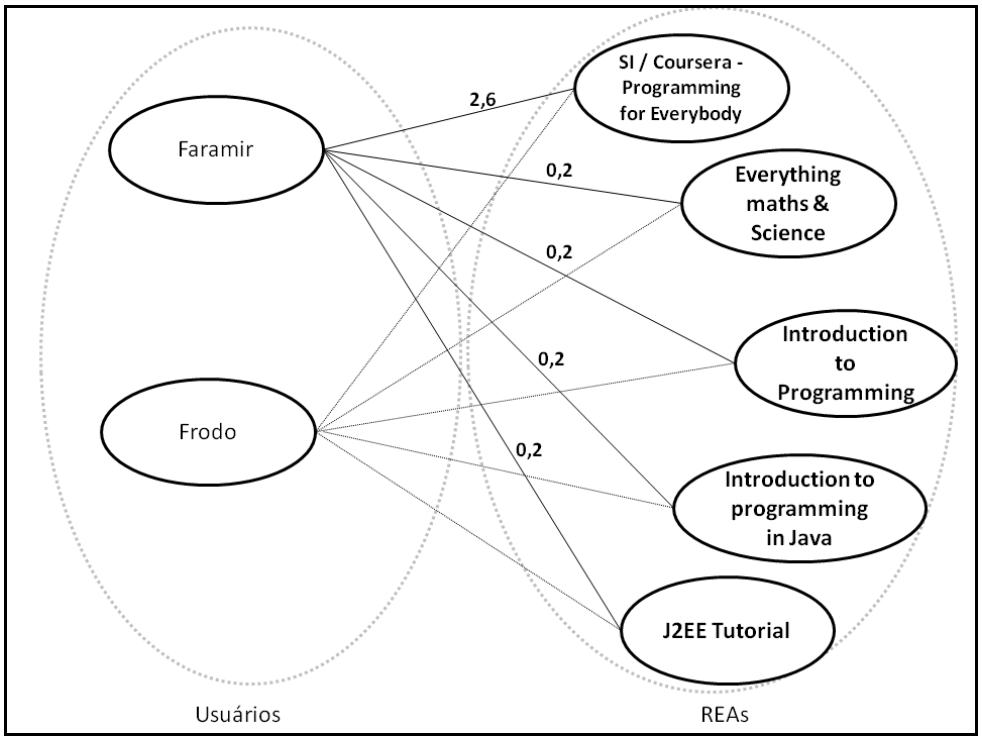


Figura 41 - Grafo bipartido com pesos para usuário Faramir

Ao final do cálculo do **MPC** o algoritmo produz como saída uma nova ordenação dos índices, conforme mostra a Tabela 4.

Tabela 4 - Saída do algoritmo de ordenação

REAs retornados do *SeeOER*	Valor da MPC	Nova ordenação
SI/Coursera - Programming for Everybody	2,6	*SI/Coursera - Programming for Everybody*
Everything maths & Science	0,2	*Everything maths & Science*
Introduction to Programming	0,2	*Introduction to Programming*
Introduction to programming in Java	0,2	*Introduction to programming in Java*
J2EE Tutorial	0,2	*J2EE Tutorial*

4.3.3 – Etapa 5 - Comparação de Instâncias

A etapa 5 é onde ocorre a aplicação do algoritmo de comparação que tem por finalidade fazer uma comparação entre os atributos das instâncias de CAM para encontrar a similaridade contextual entre usuários. O algoritmo de comparação de instâncias foi detalhado na Seção 3.4.4.

No cenário atual, ao receber os índices reordenados, o *OERecommender* faz uma nova reordenação dos resultados usando como parâmetro os atributos das instâncias entre os usuários similares que usaram os mesmos REA. Com isso, é possível obter a maior quantidade de atributos de CAM em comum entre Faramir e Frodo. Esse algoritmo é importante, pois comparará os contextos entre os usuários para, posteriormente, recomendar os REA. Um exemplo da comparação de uma instância é exibido na Tabela 5.

Tabela 5 - Comparação de instância em comum entre Faramir e Frodo para o REA: *SI / Coursera - Programming for Everybody*

Atributos		**Valores**	**Atributos**		**Valores**
Group	Title	Faramir	Group	Title	Frodo
Feed	**title**	**Mozila Firefox**	Feed	**title**	**Mozila Firefox**
	url	http:// moocexemplo.org/id=faramir.....		url	http:// moocexemplo.org/id=frodo.....
Item	**Title**	***SI / Coursera - Programming for Everybody***	Item	**Title**	***SI / Coursera - Programming for Everybody***
	Guid	**http://goo.gl/F7plLr**		**Guid**	**http://goo.gl/F7plLr**
	Type	**Html**		**Type**	**hmtl**
Event	DateTime	2015-01-10T08:49:40	Event	DateTime	2015-01-10T10:01:10
	Duration	80		Duration	-
	FollowedLink	**http://open.umich.edu/education/si/coursera-programming-everybody/winter2014/materials**		**FollowedLink**	**http://open.umich.edu/education/si/coursera-programming-everybody/winter2014/materials**
	Xfn	-		Xfn	-
	voteLink	-		voteLink	-
	Tags	**Programming text**		**Tags**	**Programming text**
	Description	**Inglês**		**Description**	**Inglês**
	Other	-		Other	-
Context	**Value**	**Swoogle**	Context	**Value**	**Swoogle**
	valueType	**MOOC**		**valueType**	**MOOC**
Session	sessionId	10	Session	sessionId	14
	ipAddress	189.39.2.219		ipAddress	201.12.2.145
UserInfo	userName	Faramir	UserInfo	userName	Frodo
	Email	faramir.exemplo@exemplo.com		Email	frodo.exemplo@exemplo.com
	Discipline	**Programming**		**Discipline**	**Programming**
Action	ActionType	http: //.../Actiontype/select	Action	ActionType	-
Entry	**Name**	**Text**	Entry	**Name**	**Text**
	Content	**Programming**		**Content**	**Programmin**
	describing Schema	-		describing Schema	-

Após comparar os atributos, obteve-se doze atributos com valores iguais, os quais estão em destaque na Tabela 5 e esses valores são utilizados para encontrar a similaridade

entre os contextos dos usuários e assim reordenar novamente os resultados. Importante lembrar que, quanto mais atributos com valores em comum tiverem, maior a similaridade contextual entre os usuários. Com essa nova ordenação é possível deixar os REAs mais similares no topo da lista de recomendação. A Tabela 6 mostra os atributos em comum entre Faramir e Frodo para os cinco REA do cenário.

Tabela 6 - Quantidade de atributos em comum entre Faramir e Frodo

	SI / Coursera - Programming for Everybody		*Everything maths & Science*	*Introduction to Programming*	*Introduction to programming in Java*	*J2EE Tutorial*
	Ação		**Ação**	**Ação**	**Ação**	**Ação**
	Seleção	Anotação	Seleção	Seleção	Seleção	Seleção
Atributos	12	12	11	11	11	11
Média	12		11	11	11	11

Comparando todas as instâncias em comum armazenadas no banco de dados, nota-se que o REA: *SI/Coursera – Programming for Everybody* é o que mais apresentou atributos em comum entre os usuários, e assim maior similaridade. Isso se deve ao fato de que o usuário Faramir executou duas ações para esse mesmo REA, baixou-o com ação `selecionando` e avaliou-o com a ação `anotando`. Esses valores de similaridade contextual são responsáveis por reordenar novamente os REAs, e essa ordenação está em ordem decrescente, conforme ilustrado na Tabela 7.

Tabela 7 - Reordenação dos REAs após execução do algoritmo de comparação

REAs	**Média da Métrica de Similaridade**
SI / Coursera - Programming for Everybody	12
Everything maths & Science	11
Introduction to Programming	11
Introduction to programming in Java	11
J2EE Tutorial	11

4.3.4 Etapa 6 e 7 - Recomendação e Reordenação dos REAs

Nas etapas 6 e 7 do *OERecommender* são efetuadas as recomendações e reordenações dos REAs.

Neste trabalho, optou-se pelo algoritmo *Slope One* (LEMIRE e MACLACHLAN, 2005), como já detalhando anteriormente, por ser considerado um método de recomendação fácil de implementar, com teoria simples, apresentando bons resultados práticos, e sendo

altamente escalável. Suas predições são calculadas a partir da comparação entre avaliações de usuários a certos itens. O *Slope One* trabalha com um método de matriz, já detalhado na Seção 3.4.5.

Neste cenário, será usada uma matriz de cinco **REAs** por cinco **Usuários**. Dado que são dois usuários utilizados no cenário, os demais serão inseridos aleatoriamente como forma de enriquecer a exemplificação. Portanto, para inserção desses dados no algoritmo, criou-se um dicionário de dados em um arquivo texto com os respectivos REAs, usuários e as notas que foram atribuídas aos mesmos. Porém, somente não é inserido valor ao REA que se deseja saber a predição, para os dados obtidos no cenário deste trabalho, a matriz resultante é exibida na Tabela 8.

Tabela 8 - Matriz das avaliações dos Usuário para os REAs

	Everything maths & Science	*SI / Coursera - Programming for Everybody*	*Introduction to Programming*	*Introduction to programming in Java*	*J2EE Tutorial*
Frodo	0	0	0	0	0
Faramir	0	3	0	0	0
Usuário 1	-1	3	2	1	0
Usuário 2	0	2	-2	0	1
Usuário 3	2	1	-1	-1	1

Ao inserir o dicionário de dados no algoritmo *Slope One*, é calculada a previsão da avaliação que Frodo daria para cada um dos REA, para tal, o algoritmo foi executado cinco vezes. Os resultados obtidos após a submissão do dicionário de dados ao algoritmo estão ilustrados na Tabela 9.

Tabela 9 - Predição calculada pelo algoritmo *Slope One* após submissão do dicionário de dados

Everything maths & Science	*SI / Coursera - Programming for Everybody*	*Introduction to Programming*	*Introduction to programming in Java*	*J2EE Tutorial*
-0.37	2.12	-1	-0.68	-0.06

O *OERecommender* finaliza sua execução exibindo por meio de *widget*, os REAs que seriam mais interessantes ao usuário alvo, no cenário deste trabalho a disposição dos REAs seria da seguinte maneira: no topo da lista de recomendações: *SI/Coursera – Programming for Everybody*, em segundo, *J2EE Tutorial*, em terceiro, *Everything maths & Science*, em quarto, *Introduction to programming in Java* e, por fim, *Introduction to Programming*,

conforme predição feita pelo *Slope One*. O *widget* com os resultados apresentados a Frodo é ilustrado na Figura 42, com um esboço similar ao de uma tela real do ambiente GCB.

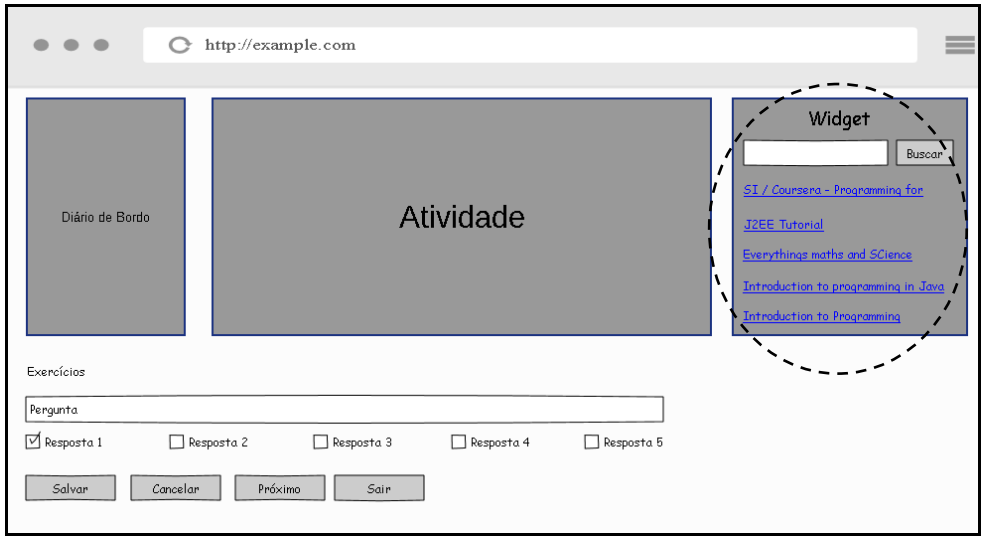


Figura 42 - Esboço de tela da recomendação de REA a Frodo por meio de *widget*

Para o usuário Frodo, esses são os REAs que foram apresentados, tendo como parâmetro a avaliação e o contexto similar de outros usuários. Assume-se que Frodo fez o *download* e avaliou dois desses cinco REAs que lhe foram sugeridos. Primeiro, *SI/Coursera – Programming for Everybody*, avaliando-o com nota 3 de satisfação e *J2EE Tutorial*, avaliando-o com nota 1 de satisfação. Esses dados serão usados para a explanação do segundo cenário.

4.4 SEGUNDO CENÁRIO

Para o segundo cenário, o usuário que faz as interações com o MOOC é Gandalf. Como já definido inicialmente, Gandalf também está fazendo o MOOC de *Programming*.

4.4.1 Etapa 1, 2 e 3 - Geração da *String* e Realização de Busca Automática e Resultados do *SeeOER*

Inicialmente a *string* de busca que ocorre para Gandalf é *programming*, igual à *string* gerada pelo sistema ao usuário Frodo no primeiro cenário.

Nessa primeira interação, cria-se uma relação do usuário Gandalf a esses REAs, e assim, novas instâncias no banco de dados vinculando o usuário ao REA são criadas. Nesse

cenário, adota-se também apenas o uso de cinco REA retornados da busca, para facilitar a compreensão da explicação, que são os mesmos retornados no primeiro cenário.

4.4.2 Etapa 4 – Ordenação dos Resultados

Com o retorno dos índices oriundos do *SeeOER*, o algoritmo reordena os índices de acordo com o nível de relevância. Para esse cenário, assume-se que esses REA já foram avaliados por diversos outros usuários, inclusive por Frodo e, as informações já estão armazenadas no banco de dados.

Portanto, tem-se uma similaridade entre o usuário Frodo e o usuário Gandalf que agora é usuário alvo no segundo cenário. Algumas avaliações foram feitas pelo usuário Frodo anteriormente o que o torna similar a Gandalf e essa similaridade é representada pelo grafo bipartido ilustrado na Figura 43.

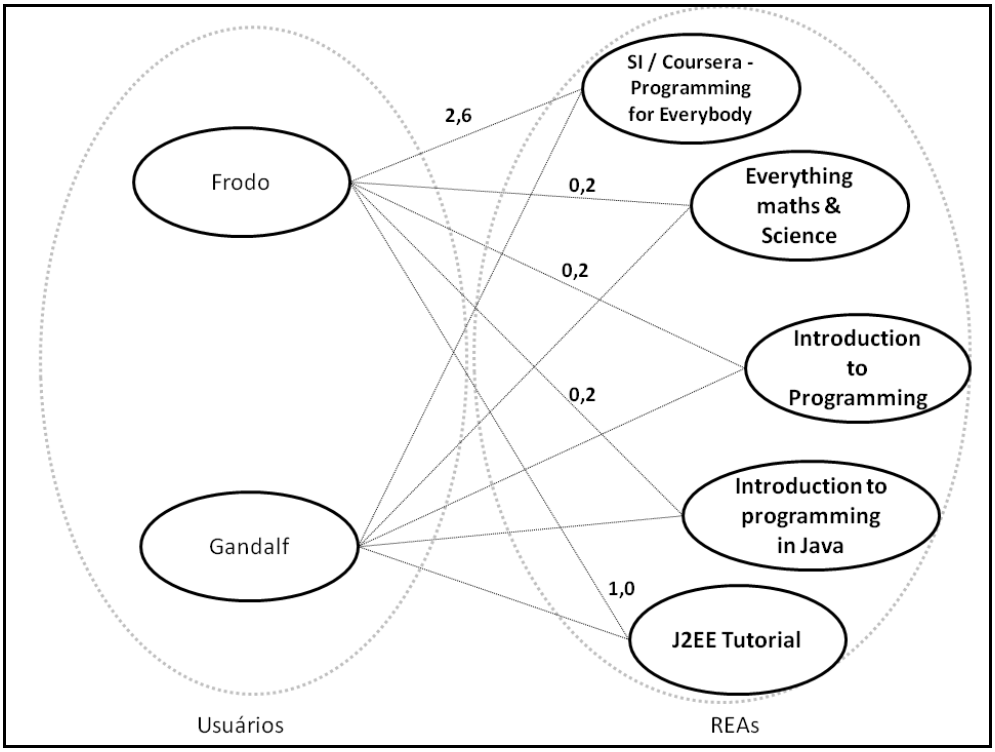


Figura 43 - Similaridade entre Frodo e o Gandalf

Ao final da execução do algoritmo de ordenação, os REA que já foram avaliados e baixados por outros usuários são ordenados tendo como base os pesos das arestas representados na Figura 43. Os pesos são obtidos pelo cálculo do `MPC` já descrito anteriormente e são somados a cada nova interação do usuário com o REA. A representação da ordenação, referente ao segundo cenário é exibida na Tabela 10.

Tabela 10 - Dados obtidos após algoritmo de ordenação

REAs retornados do *SeeOER*	Valor da MPC	Nova ordenação
SI/Coursera - Programming for Everybody	5,2	*SI/Coursera - Programming for Everybody*
J2EE Tutorial	1,2	*J2EE Tutorial*
Introduction to Programming	0,4	*Introduction to Programming*
Introduction to programming in Java	0,4	*Introduction to programming in Java*
Everything maths & Science	0,4	*Everything maths & Science*

4.4.3 Etapa 5 - Comparação de Instâncias

Nesta etapa ocorre a aplicação do algoritmo de comparação. Assim, o *OERecommender* reordena novamente os resultados com base nos atributos similares das instâncias. Ao final dessa comparação, visualiza-se na Tabela 11 a quantidade de atributos em comum entre Frodo e Gandalf para os cinco REAs do cenário.

Tabela 11 - Quantidade de atributos em comum entre Frodo e Gandalf

	SI / Coursera - Programming for Everybody		*Everything maths & Science*	*Introduction to Programming*	*Introduction to programming in Java*	*J2EE Tutorial*	
	Ação		**Ação**	**Ação**	**Ação**	**Ação**	
	Selecionando	Anotando	Selecionando	Selecionando	Selecionando	Selecionando	Anotando
Atributos	12	12	11	11	11	12	12
Média	12		11	11	11	12	

4.4.4 Etapa 6 e 7 - Recomendação e Reordenação dos REAs

Ao receber os dados reorganizados do algoritmo de comparação, é possível gerar uma matriz para ser utilizada no algoritmo *Slope One*. Neste cenário será utilizada uma matriz cinco **REAs** por cinco **Usuários**. Para enriquecer a exemplificação serão adicionados alguns usuários fictícios na matriz.

O dicionário de dados gerado para ser utilizado no algoritmo *Slope One* foi criado. Para tanto, conforme explicado no cenário anterior, somente não é inserido valor ao REA que se deseja saber a predição, a matriz resultante é exibida na Tabela 12

Tabela 12 - Matriz das avaliações dos Usuários para os REAs

	Everything maths & Science	*SI / Coursera - Programming for Everybody*	*Introduction to Programming*	*Introduction to programming in Java*	*J2EE Tutorial*
Frodo	-0.37	2.12	-1	-0.68	-0.06
Faramir	0	3	0	0	0
Gandalf	0	0	0	0	0
Usuário 1	0	2	-2	0	1
Usuário 2	2	1	-1	-1	1

Ao submeter o dicionário de dados ao algoritmo *Slope One*, as previsões das avaliações de Gandalf aos REAs que lhe foram sugeridos são mostradas na Tabela 13.

Tabela 13 - Previsão das avaliações de Gandalf aos REAs

Everything maths & Science	*SI / Coursera - Programming for Everybody*	*Introduction to Programming*	*Introduction to programming in Java*	*J2EE Tutorial*
-0.75	2.68	-1.06	-1.68	0.81

Ao final da execução do *Slope One*, o *OERecommender* exibe por meio de *widget* os REAs que seriam mais interessantes a Gandalf. No topo da lista de recomendações: *SI/Coursera – Programming for Everybody*, em segundo, *J2EE Tutorial*, em terceiro, *Everything maths & Science*, em quarto, *Introduction to programming* e, por fim, Introduction *to Programming in Java*. Um esboço similar ao de uma tela real do ambiente GCB é exibido na Figura 44.

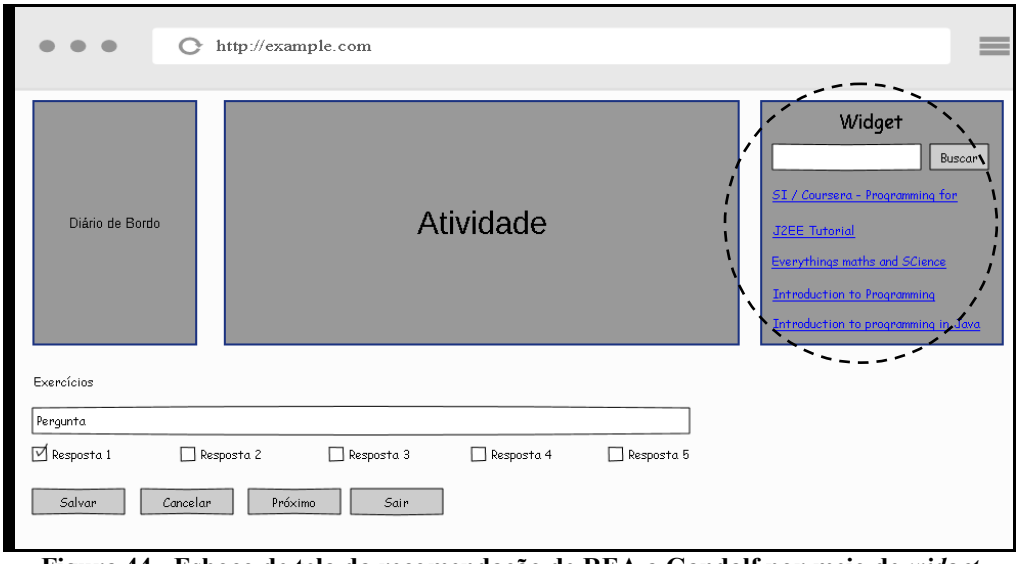


Figura 44 - Esboço de tela da recomendação de REA a Gandalf por meio de *widget*

Quanto mais interações entre usuários com o MOOC e, consequentemente, com os REA recomendados, maiores serão os acertos de recomendações para o próprio usuário e novos usuários que farão o MOOC.

4.5 DISCUSSÕES

A prototipação em ambiente controlado pode ser considerada como insuficiente, porém faz-se necessária para fundamentar teórica e metodologicamente uma posterior implementação.

Os resultados das recomendações obtidos nos cenários fictícios evidenciam que é viável a implementação do *OERecommender*, pois em contextos semelhantes essas técnicas e algoritmos obtiveram bons resultados.

É necessária a continuidade do trabalho para aprimorar os resultados previamente obtidos, já que os mesmos podem apresentar alguns problemas com falso positivo ou falso negativo.

A escolha do algoritmo de recomendação *Slope One* mostrou-se viável, pois sua implementação apresentou um bom desempenho na prototipação, porém vale lembrar, que os dicionários de dados foram preparados para o algoritmo e isso colaborou para seu êxito. Métodos estatísticos precisam ser utilizados para realizar uma simulação mais completa.

É possível ocorrer problemas de falso positivo e também falso negativo na avaliação do usuário com relação ao REA. Ele pode gostar do REA e avaliá-lo como ruim e pode não gostar do REA e avaliá-lo como bom. A avaliação desse tipo de variável foge do escopo deste trabalho.

Para todos os efeitos, este é um trabalho inicial para um extenso trabalho posterior. É recomendável que a implementação de todo o *OERecommender* seja realizado em ambiente real, possibilitando posteriormente comprovar sua efetividade por meio de experimentos.

4.6 TRABALHOS RELACIONADOS

Alguns trabalhos foram usados como base, embora esses não tenham sido concebidos para ser diretamente utilizados em MOOC. Hsu (2008) descreve um SR para um curso *online* de inglês capaz de recomendar materiais que auxiliam no processo de aprendizagem. Essas recomendações são geradas por meio de filtragem baseada em conteúdo, filtragem colaborativa e técnicas de mineração de dados. Experimentos realizados durante um ano mostraram que a maioria dos estudantes aceitaram as recomendações de materiais indicados e lições sugeridas pelo SR, provando sua viabilidade e eficiência. Este trabalho possui características similares ao trabalho desenvolvido por Hsu (2008), principalmente no que tange aos métodos de filtragem dos materiais para posterior recomendação e, também, na forma de sugerir materiais que realmente sejam de interesse dos usuários alvo. Wolpers et al., (2007); Najjar et al., (2005) retratam o desenvolvimento do modelo CAM que permite monitorar o comportamento do usuário com o computador, em que, como e quanto tempo ele utilizou determinado ambiente e armazenou essas informações em uma base de dados para posterior utilização. Esse trabalho foi importante para o desenvolvimento do *OERecommender*, pois gerencia as informações contextuais das atividades que o usuário realiza no MOOC servindo assim como artefato de entrada para o processo de busca por REAs realizadas pelo *SeeOER* (GAZZOLA et al., 2014) em diversos repositórios na Web. Ochoa e Duval (2006) utilizaram métodos de similaridade entre usuários por meio de dobradura de grafos, método esse que foi replicado no *OERecommender*, pois já havia sido testado pelos autores e com bons resultados em contextos similares, por isso, foram utilizados na concepção deste SR.

O trabalho aqui proposto se difere dos acima referenciados, pois recomenda REAs em ambientes MOOC. Para tal, o *OERecommender* considera informações contextuais gerenciando os metadados de CAM necessários para realizar a busca por REAs na Web, sendo auxiliado pelo *SeeOER* (GAZZOLA et al., 2014) um buscador específico para encontrá-los. Esses fatores somados tornam o *OERecommender* um SR diferenciado da maioria dos SR atualmente desenvolvidos para ambientes MOOC.

4.7 CONSIDERAÇÕES FINAIS

Neste capítulo, dois cenários fictícios foram representando por meio de prototipação, com o objetivo de demonstrar a aplicabilidade deste trabalho. Tal aplicabilidade foi demonstrada por

meio da completa execução dos algoritmos contidos no *OERecommender*. Os exemplos demonstrados evidenciam que o *OERecommender* pode existir em um MOOC real.

Os resultados obtidos possibilitaram a conclusão de itens tais como: **(i)** é possível extrair *string* implicitamente utilizando *Slogger* como meio para coletar esses metadados e armazena-los em instâncias CAM num banco de dados Exist XML; **(ii)** é viável a utilização do algoritmo *Slope One* para recomendar REAs, por se tratar de um algoritmo que utiliza métodos simples e eficazes como já comprovado na sua implementação neste trabalho e por outros como o de Lemire e Maclachlan (2005) e Wang e Ye (2009) e; **(iii)** para os demais algoritmos, ordenação e comparação, a sua utilização no *OERecommender* foi embasada em trabalhos correlatos, portanto, sendo necessário implementação para posterior conclusões.

No próximo capítulo são apresentadas as conclusões desta dissertação, incluindo as principais lições aprendidas, bem como, os trabalhos futuros a serem desenvolvidos.

5 CONCLUSÃO

Esta dissertação apresentou o *OERecommender*, um SR de REAs para MOOCs. O uso de um projeto para este fim é imprescindível para que o mesmo seja posteriormente implementado nesse tipo de ambiente, mitigando problemas como: diversidade de material de apoio no processo de ensino aprendizagem e reduzindo a evasão que no contexto de MOOC que atualmente é em torno de 90% (CHUNG, 2013).

O *OERecommender* consiste basicamente em um processo de vários estágios, envolvendo a busca, ordenação até a recomendação desses REAs ao usuário alvo. Esse processo é embasado principalmente em filtragens colaborativas e nas informações contextuais que o usuário alvo está envolvido no momento da realização das atividades no MOOC.

Como forma de avaliação, os algoritmos que compõem o *OERecommender* foram submetidos ao contexto de um MOOCs fictícios apresentando resultados interessantes nas recomendações de REAs ao usuário alvo.

5.1 CONTRIBUIÇÕES E LIMITAÇÕES

O diferencial do *OERecommender* em relação aos demais sistemas de recomendação utilizados em AVAs atuais é justamente a forma com que o mesmo trata o processo de recomendação, utilizando informações contextuais como filtragem dos REA para o processo de predição.

Os resultados iniciais obtidos na prototipação dos cenários nos evidenciam que a predição será mais eficaz do que os demais métodos desde as primeiras recomendações, pois possivelmente irá minimizar o tempo de aprendizagem de máquina e aumentará a satisfação do usuário alvo. Essas afirmações deverão ser utilizadas como hipóteses para um estudo empírico posterior, após a implementação de todos os algoritmos do *OERecommender* em um ambiente real. Portanto, a principal contribuição deste trabalho é a viabilização de um SR que utiliza informações contextuais dos usuários no processo de recomendação de REAs em ambientes MOOC.

Entre as limitações mapeadas oriundas da não implementação e testes de implementação do *OERecommender* no ambiente real, destacam-se: **(i)** a eficácia do método de classificação dos REAs em ambiente MOOC; **(ii)** funcionamento dos métodos de

similaridade entre usuários e REAs por meio de grafos em ambiente para milhares de usuários, e; **(iii)** se os algoritmos de ordenação e comparação serão eficazes no ambiente real.

5.2 TRABALHOS FUTUROS

Esta seção apresenta os principais trabalhos que serão desenvolvidos futuramente como continuação do trabalho apresentado nesta dissertação:

- **Implementação do projeto proposto:** embora a aplicabilidade do projeto tenha sido avaliada por meio da realização de prototipação com cenários fictícios, novas análises devem ser realizadas visando ampliar essa avaliação do ponto de vista de viabilidade e efetividade. As comparações com outras abordagens similares poderão ser realizadas e também futuramente:
    - Alteração de plataforma apropriada para implementação;
    - Alteração de linguagem de programação adequando-se à plataforma;
    - Testes unitários e funcionais em cada etapa do processo de desenvolvimento e implementação do *OERecommender*.

- **Extensão do projeto do *OERecommender:*** o domínio de aplicação relacionado a sistemas de recomendação é bastante amplo, sendo que a primeira etapa que é a elaboração do projeto já foi coberta na proposta desta dissertação. Ainda assim, esse SR pode ser estendido para incorporar e/ou substituir funcionalidades ainda não cobertas por ele. Alguns exemplos são:
    - Utilização de novos algoritmos em substituição ou acréscimo aos abordados neste trabalho para fazer o processo de ordenação dos REAs;
    - Incorporar novas técnicas de coleta dos metadados contextuais dos usuários, diferentes das utilizadas por este trabalho;
    - Encontrar similaridades entre usuários, utilizando métodos alternativos, diferentes de grafos, abordado neste trabalho;
    - Sistema de predição de REA utilizando diferentes métodos matemáticos como, por exemplo: distintos métodos de aprendizagem de máquina.

REFERÊNCIAS

ABEYWARDENA I.S.; CHAN, C.S. *Review of the current oer search dilemma*. Proceedings of the 57th World Assembly of International Council on Education for Teaching (ICET 2013), 25-28 June 2013, Thailand.

ADOMAVICIUS, G.; TUZHILIN, A. *Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions*. IEEE Transactions on Knowledge and Data Engineering, p. 734-749, 2005. Disponível em: http://www.computer.org/portal/Web/csdl/doi/10.1109/TKDE.2005.99. Acesso em 13 jan. 2014

ATKINS, E.D.; BROW, S.J.; HAMMOND. A.L. *A Review of the Open Educational Resources (OER) Movement: Achievements, Challenges*, and New Opportunities. February 2007. Disponível em: <http://www.hewlett.org/uploads/files/ReviewoftheOERMovement.pdf>. Acesso em 14 nov. 2013.

BAKER M.; BOAKES R. *Slogger: A profiling and analysis system based on Semantic Web technologies*. 2008. Disponível em: <http://dl.acm.org/citation.cfm?id=1402722>. Acesso em: 7 jul 2013.

BALOIAN, N.; BREUER, H.; HOEKSEMA, K.; HOPPE, U.; MILRAD, M. *Implementing the Challenge Based Learning in Classroom Scenarios,* in: Sofoklis Sotiriou (ed.): Proceedings of the symposium on Advanced Technologies in Education, July 2004, Argostili, Greece.

BARCELLOS, C. D.; MUSA, D. L.; BRANDÃO, A. L.; WARPECHOWSKI, M. *Sistema de Recomendação Acadêmico para Apoio a Aprendizagem*. 2007. Disponível em: <http://www.cinted.ufrgs.br/ciclo10/artigos/3fDaniela.pdf>. Acesso em 7 mar. 2013.

BERKONSKY, S.; EYTANI, Y.; MANEVITZ, L. *Efficient Collaborative Filtering in Content-Addressable Spaces*. Personalization Techniques and Recommender Systems. [S.l.]:, 2008. Cap 5, p.135-164.

BERRI, J.; BENLAMRI, R.; ATIF, Y. *Ontology-based framework for context-aware mobile learning*. In: ACM (Ed.). IWCMC '06: Proceedings of the 2006 international conference on Wireless communications and mobile computing. [S.l.: s.n.], 2006.

SCHIRRU, R.; BAUMANN, S.; MEMMEL, M.; DENGEL, A. *Extraction of Contextualized User Interest Profiles in Social Sharing Platforms.* Journal of University Computer Science, 16, 2010. pg 2196-2213.

DEL BLANCO A.; SERRANO, A.; FREIRE, M.; ORTIZ, I.M.; MANJÓN, B.F. *E-Learning Standards and Learning Analytics*. Global Engineering Education Conference (EDUCON), 2013 IEEE Digital Object Identifier: 10.1109/EduCon.2013.6530268, Page(s): 1255 – 1261.

BLINKX. *Blinkx to lead in video search engine.* 2006. Disponível em: <http://phys.org/news/2006-07-blinkx-video.html>. Acesso em 12 nov. 2013.

BRAUN S.; SCHMIDT. A. *Don't annoy the informal teacher: Context-aware mediation of communication for workplace learning*. In: 6th International Conference on Knowledge Management. Graz, Austria.: [s.n.], 2006.

BREESE J.S.; HECKERMAN, D.; KADIE C. *Empirical Analysis of Predictive Algorithms for Collaborative Filtering*, October 1998, Disponível em: <http://research.microsoft.com/pubs/69656/tr-98-12.pdf>. Acesso em: 12 nov 2013.

BROISIN J. *A model driven approach for the management of learning object life cycle. Intelligent, Interactive*, Learning Object Repositories Network (I2LOR), p. 14–18, nov 2005. Vancouver, Canada.

BROWN, J.S; ADLER, R.P. *Minds on Fire: Open Education, the Long Tail, and Learning 2.0*. Educause Review, 43(1), 16-32, 2008. Educase.

BRUSILOVSKY P.; MILLAN E. *User models for adaptive hypermedia and adaptive educational systems*. Springer, v. 4321, p. 3–53, 2007. Brusilovsky, A. Kobsa, and W. Nejdl, editors - Theadaptive Web.

BURKE, R. *Hybrid Recommender Systems: Survey and Experiments. User Modeling and User-Adapted Interaction* 12(4), 331–370. 2002.

BURKE, R. *Hybrid Web recommender systems*. In: The AdaptiveWeb, pp. 377–408. 2007. Springer Berlin / Heidelberg

BURNET, M.; PREKOP, P.; RAINSFORD, C. *Intimate location modeling for context aware computing*. Workshop on location modeling for ubiquitous computing. In:. [S.l.: s.n.], 2001. p. 77–82. Disponível em: <http://www.teco.edu/locationws/13.pdf> Acesso em: 07 dez. 2013

BUTCHER, N. *A basic guide to open educational resources*. British Columbia/Paris: col-unesco. 2011. Disponível em: <http://unesdoc.unesco.org/images/0021/002158/215804e.pdf >Acesso em: 22 fev. 2014

CCK08. (2008). *Connectivism and Connected Knowledge*. 2008. Disponível em <http://wwwapps.cc.umanitoba.ca/moodle/course/view.php?id=20>. Acesso em 21 ago. 2013

CHUNG, C. (2013). *Beyond Dropouts and Dabblers: A Broader View of Auditing MOOCs*, MOOC News and Reviews, Disponível em http://moocnewsandreviews.com/beyond - dropouts-and-dabblers-taking-a-broader-view-of-auditing-moocs/>. Acesso em 21 ago. 2013

COLLIS, B.; STRIJKER, A. *Technology and human issues in reusing learning objects*. Journal of Interactive Media in Education, v. 4, 2004.

DALZIEL, J. *Lesson from LAMS for IMS learning design*. In Proceedings of the Sixth International Conference on Advanced Learning Technologies. 2006. pg.1101-1102

DESHPANDE, M., and KARYPIS, G. *Item-based top-N recommendation algorithms*. ACM Transactions on Information Systems (TOIS), 2004. 22(1), 143-177.

DING, L.; FININ, T.; JOSHI, A.; PAN, R.; COST, S.R.; PENG, Y.; REDDIVARI, P.; DOSHI, V.; SACHS, J. *Swoogle: A search and metadata engine for the semantic Web*. In: ACM Conference on Information and Knowledge Management (CIKM). Washington D.C., USA.: [s.n.], 2004. p. 8–13.

DOWNES, S. (2008). *Places to go: Connectivism & Connective Knowledge*. Innovate, 5(1). Disponível em: <http://www.innovateonline.info/index.php?view=article&id=668>. Acesso em 6 ago. 2013.

DYCKHOFF A.L.; ZIELKE, D.; BÜLTMANN, M.; CHATTI, M.A.; SCHROEDER, U. *Design and implementation of a learning analytics toolkit for teachers*. Educational Technology & Society, v. 15, n. 3, p. 58–76., 2012. Disponível em: <http://www.ifets.info/journals/15_3/5.pdf> Acesso em: 6 ago. 2013.

EDUCASE. *7 Things You Should Know About Analytics.* 2010. Disponível em <https://net.educause.edu/ir/library/pdf/ELI7059.pdf>. Acesso em 3 out. 2013.

EDX. EDX, 2013. Disponível em: <https:// http://docs.edx.org/ >. Acesso em 10 ago. 2013.

EL HELOU, S.; SALZMANN, C.; GILLET, D. *The 3a personalized, contextual and relation-based recommender system*. Journal of Universal Computer Science (J.UCS), v. 16, n. 16, p. 2179–2195, aug 2010.

EXIST. *EXist XML database.* 2006 - disponível em: <http://edutechwiki.unige.ch/en/EXist_XML_database>. Acesso em: 12 nov 2013.

FAN, W.; GORDON, M.; PATHAK, P. *A generic ranking function discovery framework by genetic programming for information retrieval*. Information Processing and Management., v. 40, p. 587–602, 2004.

FELFERNIG, A.; BURKE, R. *Constraint-based recommender systems: Technologies and research issues*. In ACM International Conference on Eletronic Commerce (ICEC08), pages 17-26, 2008.

GAEBEL M. (2013), *MOOCs Massive Open Online Courses.* Disponível em: <http://www.eua.be/eua-work-and-policy-area/building-the-european-higher-education-area/e-learning/moocs.aspx>. Acesso em 7 out. 2013.

GAROFALAKIS, J.; GIANNAKOUDI, T.; SAKKAPOULOS, E. *Semantic Web site usage analysis: The organ system*. World Wide Web 2006 Workshop Logging Traces of Web Activity: The Mechanics of Data Collection, may 2006. Edinburgh, UK.

GAZZOLA M.G.; CIFERRI, C.D.A.; GIMENES, I.M.S. *SeeOER: Uma Arquitetura para Mecanismo de Busca na Web por Recursos Educacionais Abertos*. Congresso Brasileiro de Informática na Educação, III; Simpósio Brasileiro de Informática na Educação, XXV, 2014, Dourados.

GIMENES, I.M.S.; BARROCA, L.; FELTRIM, V.D. *Tendências na Educação a Distância e Educação Aberta em Computação*. In: Souza, A. F.; Galante, R.; Cesar Junior, M. F.; Pozo, A. T. R.. (Org.). XXXI Jornadas de Atualização em Informática. 2012. 1ed.Porto Alegre: SBC, v. 1, p. 5-45.

GOOGLE. *Create Custom Modules*. (2013). Disponível em: <https://code.google.com/p/course-builder/wiki/CreateModules>. Acesso em 4 mar. 2014

GOVAERTS, S.; EL HELOU, S.; DUVAL, E.; GILLET, D. *A federated search and social recommendation widget.* 2011. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.474.8272>. Acesso em 12 mar. 2014.

HILEN, J. *Open Educational Resources: Opportunities and Challenges*. OECD's Centre for Educational Research and Innovation. Disponível em: < <http://www.oecd.org/dataoecd/5/47/37351085.pdf>. Acesso em: 25 set. 2013.

HOLZ, H.; ROSTANIN O.; DENGEL, A.; SUZUKI, T.; MAEDA, K.; KANASAKI, K. *Task-based process know-how reuse and proactive information delivery in task navigator*. In: ACM Conference on Information and Knowledge Management (CIKM). Arlington, VA, USA.: [s.n.], 2006. p. 6–11.

HSU, M. H. *A personalized English learning recommender system for ESL students*. Elsevier, Volume 24, Issue 1, January 2008, Pg 683-688.

IEEE 1484.12.2. *Draft Standard for Learning Technology*—ECMAScript Application Programming Interface for Content to Runtime Services Communication." 2002.

JANNACH, D. *Finding preferred query relaxations in content-based recommenders*. In: 3rd International IEEE Conference on Intelligent Systems, pp. 355–360. 2006

JANNACH, D.; ZANKER, M.; FELTERNIG, A.; FRIEDRICH, G. *Recommender Systems An Introduction*. Cambridge University. 2011

KITCHENHAM, B. *Procedures for performing systematic reviews*. Keele, UK, Keele University 33.2004 (2004): 1-26. Disponível em: <http://people.ucalgary.ca/~medlibr/kitchenham_2004.pdf>. Acesso em 07 ago. 2013

LEMIRE D.; MACLACHLAN. A. 2005. *Slope One Predictors for Online Rating-Based Collaborative Filtering*, Disponível em: <http://lemire.me/fr/documents/publications /lemiremaclachlan_sdm05.pdf>. Acesso em: 12 nov 2013.

LINDEN, G.; SMITH, B.; YORK, J. *Amazon.com, recommendations: Item-to-item collaborative filtering.* IEEE Internet Computing, v. 7, n. 1, p. 76–80, 2003. Disponível em: < http://www.cs.umd.edu/~samir/498/Amazon-Recommendations.pdf > Acesso em 7 dez. 2013

LINDSTAEDT. *APOSDLE: Learning Real-Time and Real-Place*, Online Educa 2006, Berlin, 30 November 2006.

LONSDALE, P.; BABER, C.; SHARPLES, M.; BYRNE, W.; ARVANITIS, T.; BRUNDELL, P.; BEALE, H. *Context awareness for MOBIlearn: creating an engaging learning experience in an art museum*. Proceedings of MLEARN 2004. Bracciano, Rome: LSDA, 2004.

LU, J. *Full-text federated search in peer-to-peer networks*. PhD thesis. Carnegie Mellon University, 2007

MAHMOOD T.; RICCI, F. *Improving recommender systems with adaptive conversational strategies*. ACM, 2009. : C. Cattuto, G. Ruffo, F. Menczer (eds.).

MARTINEZ J. M. *MPEG-7 overview*. 2003. Disponível em: <http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm>. Acesso em: 12 nov. 2014.

MCSHERRY F.; MIRONOV, I. *Differentially private recommender systems: building privacy into the net*. In: ACM (Ed.). KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. New York, NY, USA: [s.n.], 2009. p. 627–636.

MEISZNER, A. The *why and how of open education: With lessons from the openSE and openED project*. Collaborative Creativity Group. 95. 2011. Google Scholar Download: The Why and How of Open Education.pdf . 2011. Disponível em: <http://www.opense.net/>. Acesso em 26 ago. 2013.

MOTA, R. e INAMORATO, A. *MOOC: uma revolução em curso*, Jornal da Ciência. 2012. Disponível em: <http://www.jornaldaciencia.org.br/Detalhe.jsp?id=85111>. Acesso em: 9 de mar. 2014.

NAJJAR, J.; MEIRE, M.; DUVAL E. *Attention metadata management: Tracking the use of learning objects through attention.xml*. In: World Conference on Educational Multimedia, Hypermedia and Telecommunications. [S.l.: s.n.], 2005. p. 1157–1161.

NASCIMENTO, M. A., SANDER, J.; POUND, J. Analysis of SIGMOD's co-authorship graph. SIGMOD Rec., 32(3): 8-10.

NEPOMUK. 2006. *Nepomuk Semantic desktop*. Disponível em: <http://nepomuk.semanticdesktop.org/>. Acesso em: 7 ago. 2014

NGUYEN L.; DO P. *Learner model in adaptive learning*. World Academy of Science Engineering and Technology, v. 45, p. 395–400, 2008.

OCHOA X.; DUVAL. E. *Use of contextualized attention metadata for ranking and recommending learning objects*. CAMA'06, v. 11, nov 2006. Arlington, Virginia, USA.

OGATA. H. *Computer supported ubiquitous learning environment for vocabulary learning*. Int. J. 2004 Learning Technology, Vol. X, No. Y, xxxx. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.183.6571&rep=rep1&type=pdf>. Acesso em 12 nov. 2013.

OPENMOOC. 2013. Disponível em: <http://openmooc.org/>. Acesso em 5 out. 2013

PAGE L.; BRIN S. *The PageRank Citation Ranking: Bringing order to the Web.* [S.l.], 1998. Disponível em: <http://ilpubs.stanford.edu:8090/422/1/1999-66.pdf>. Acesso em: 11 fev. 2014.

PECO Pernias, P.; LUJAN-MORA, S. (2013), *Architecture of a MOOC based on CourseBuilder*, Information Technology Based Higher Education and Training (ITHET), International Conference on, vol., no., pp.1,8, 10-12 doi: 10.1109/ITHET.2013.6671045.

RADLINSKI, F.; JOACHIMS, T. *Query chains: Learning to rank from implicit feedback*. In: ACM Conference on Knowledge Discovery and Data Mining. [S.l.: s.n.], 2005.

REICH, J. *Teaching teachers to tweet (part ii).* Education Week – EdTech Research. 2012. Disponível em: <http://blogs.edweek.org/edweek/edtechresearcher/2012/08/teaching_teachers_to_tweet.html>Acesso em: 12 out. 2013

RESNICK, P.; ZECKHAUSER, R.; FRIEDMAN, E.; KUWABARA, K. *Reputation systems*. Communications of the ACM, New York, v.43, n.12, p. 45-48. 2000.

RESNICK, P., VARIAN, H.R. (1997): *Recommender systems***.** Communications of the ACM 40(3), 56–58.

RICCI, F.; ROKACH, L.; SHAPIRA, B.; KANTOR, P. *Recommender Systems Handbook*, DOI 10.1007/978-0-387-85820-3_3, © Springer Science+Business Media, LLC 2011 Semeraro, G., Basile, P., de Gemmis, M., Lops, P.: User Profiles for Personalizing Digital Libraries. In: Y.L. Theng, S. Foo, D.G.H. Lian, J.C. *Handbook of Research on Digital Libraries*: Design, Development and Impact, pp. 149–158. IGI Global (2009). ISBN 978-159904879-6.

RODA C.; THOMAS J. *Attention aware systems: Theories, application and research agenda.* Journal on Computers in Human Behaviour, v. 22, p. 557–587., 2006.

SANTANA, B.; ROSSINI, C.; PRETTO, N.D.L. *Recursos Educacionais Abertos: práticas colaborativas políticas públicas*. 2012. 1ª ed., 1 imp. – Salvador: Edufba; São Paulo: Casa da Cultura Digital. 246 p.

SARWAR, B.; KARYPIS, G.; KONSTAN, J.; RIEDL, J. *Item-based collaborative filtering recommendation algorithms*. In: 10th International World Wide Web Conference (WWW10). [S.l.: s.n.], 2001.

SAUERMANN, L. *The semantic desktop - a basis for personal knowledge management.* In: Proceedings 5th International Conference on Knowledge Management. Graz, Austria: [s.n.], 2005.

SCHAFER, J.B.; FRANKOWSKI, D.; HERLOCKER, J.; SEN, S. *Collaborative Filtering Recommender Systems.* The Adaptive Web. V. 4321/2007, p. 291-324. Disponível em <http://www.springerlink.com/content/t87386742n752843> Acesso em: 22 de out. 2013.

SCHEIN, A.I.; POPESCUL, A.; UNGAR, L.H.; PENNOCK, D.M. *Methods and Metrics for Cold-Start Recommendations*. SIGIR '02 Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval. 253-260, ACM New York, NY,USA. 2002.

SHEPHERD M.; WATTERS, C. *Adaptive user modeling for filtering electronic news*. In: 35th Annual Hawaii International Conference on System Sciences. HICSS. [S.l.: s.n.], 2002. p. 1180– 1188.

SIEMENS, G. *MOOCs are really a platform*. Elearnspace. Disponível em: < http://www.elearnspace.org/blog/2012/07/25/moocs-are-really-a-platform/> Acesso em: 12 ago. 2013

SIEMENS, J.; GASEVIC, D.; HAYTHORNTHWAITE, C.; DAWSON, S.; BUCKINGHAM, S.; FERGUSON, R.; DUVAL, E.; VERBERT, K.; BAKER, R.S.J. *Open learning analytics: an integrated and modularized platform: Proposal to design, implement and evaluate an open platform to integrate heterogeneous learning analytics techniques*. 2011. Disponível em: <http://solaresearch.org/OpenLearningAnalytics.pdf > Acesso em: 10 set. 2013

SPECHT, M.S. *Ace - adaptive courseware environment. In: Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*. London, UK: Springer-Verlag., 2000. p. 380–383.

STERN, H.; BEHAM, G.; KRAKER, P.; LINDSTAEDT, S.N. *Content recommendation in aposdle using the associative network.* Journal of Universal Computer Science, v. 16, n. 16, p. 2214– 2231, 2010.

THOMAS, P. *Server characterisation and selection for personal metasearch*. Tese (Doutorado) — Australian National University, 2008. Disponível em: <http://annotate.thoughtlets.org/pubs/thomas_thesis.pdf>. Acesso em: 22 set. 2013.

VERBERT, K.; JOVANOVIC, J.; GASEVIC, D.; DUVAL E. *Repurposing learning object components*. OTM 2005 Workshop on Ontologies, Semantics and E-learning, v. 3, nov 2005. Agia Napa, Cyprus.

VERBERT, K.; MANOUSELIS, N.; OCHOA, X.; WOLPERS, M.; DRASHSLER, H.; BOSNIC, I.; DUVAL E. *Context-aware recommender systems for learning: a survey and future challenges.* 2012. Disponível em <Https://lirias.kuleuven.be/bitstream/123456789/338644/3/survey-final.pdf>. Acesso em: 8 nov. 2013.

VÖLKEL M.; HALLER H. *Conceptual data structures (cds) – towards an ontology for semi-formal articulation of personal knowledge*. In: 14th International Conference on Conceptual

Structures. [S.l.]: 14th International Conference on Conceptual Structures, 2006. p. 16–21. Aalborg, Denmark.

WANG, P.; YE, W.H. *A Personalized Recommendation Algorithm Combining Slope One Scheme and User Based Collaborative Filtering*. ISS'09, 2009. Disponível em: <http://www.computer.org/csdl/proceedings/iis/2009/3618/00/3618a152-abs.html>. Acesso em: 22 jan. 2014

WEINREICH, H.; OBENDORF, H.; HERDER E. *Data cleaning methods for client and proxy logs.* World Wide Web 2006 Workshop Logging Traces of Web Activity: The Mechanics of Data Collection, may 2006. Edinburgh, UK.

WELSH D.; DRAGUSIN M. *The New Generation of Massive Open Online Course (MOOCS) and Entrepreneurship Education. Small Business Institute* ® Journal Vol. 9, No. 1, 51-65. 2013.

WILEY, D. *Defining the open in open content. The Open Future*, Educase Review, p. 15-20. 2010. Disponível em: <http://www.educause.edu/ero/article/openness-catalyst-educational-reformation>. Acesso em: 21 jan. 2014

WILEY, D.A. *The learning objects literature*. 2002. Disponível em: <http://www.opencontent.org/docs/wiley-lo-review-final.pdf>. Acesso em: 3 mar. 2014.

WOLPERS, M.; NAJJAR, J.; VERBERT, K.; DUVAL, E. *Tracking actual usage: the attention metadata approach*. Educational Technology & Society, v. 10, n. 3, p. 106–121., 2007.

WEIBEL, S.; KUNZE, J.; LAGOZE, C.; WOLF, M. *Dublin core metadata for resource discovery*. 1998. Disponível em: <http://dl.acm.org/citation.cfm?id=RFC2413>. Acesso em: 9 ago. 2013.

YAU, J.; JOY, M. *A context-aware and adaptive learning schedule framework for supporting learners' daily routines*. In: SOCIETY., I. C. (Ed.). Second International Conference on Systems. Washington, DC, USA: [s.n.], 2007. p. 31–37.

ZHAO, X.; ANMA, F.; NINOMIYA, T.; OKAMOTO, T. *Personalized adaptive content system for context-aware mobile learning*. In: SCIENCE, I. J. of C.; SECURITY, N. (Ed.). IJCSNS. [S.l.: s.n.], 2008. (8, v. 8)