UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

iRSS - UMA FERRAMENTA DE AGREGAÇÃO DE RSS BASEADA EM TAXONOMIA

Marcelo Canevello Ferreira

Orientadores

Prof. Dr. Astério Kiyoshi Tanaka
Prof. Dr. Sean Wolfgand Matsui Siqueira

RIO DE JANEIRO, RJ – BRASIL

SETEMBRO DE 2009

iRSS - UMA FERRAMENTA DE AGREGAÇÃO DE RSS BASEADA EM TAXONOMIA

Marcelo Canevello Ferreira

DISSERTAÇÃO APRESENTADA COMO REQUISITO PARCIAL PARA OBTENÇÃO DO TÍTULO DE MESTRE PELO PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA DA UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO (UNIRIO). APROVADA PELA COMISSÃO EXAMINADORA ABAIXO ASSINADA.

Aprovada por:

_____

Astério Kiyoshi Tanaka, Ph.D. - UNIRIO

_____

Sean Wolfgand Matsui Siqueira, D.Sc. - UNIRIO

_____

Fernanda Araujo Baião, D.SC. - UNIRIO

_____

Geraldo Bonorino Xexéo, D.Sc. - UFRJ

RIO DE JANEIRO, RJ – BRASIL

SETEMBRO DE 2009

(OBS.: Remover posteriormente esta observação... A ficha deve ser a oficial da biblioteca e a qtd de folhas deve ser atualizada conforme a versão final da dissertação)

Ferreira, Marcelo Canevello.
F383 iRSS – Uma ferramenta de agregação de RSS baseada em taxonomia / Marcelo Canevello Ferreira, 2009.
xii, 102f.

Orientador: Astério Kiyoshi Tanaka.
Co-orientador: Sean Wolfgand Matsui Siqueira.
Dissertação (Mestrado em Informática) – Universidade Federal do Estado do Rio de Janeiro, Rio de Janeiro, 2009.

1. Tecnologia da informação. 2. Sistemas de informação. 3. RSS. 4. Agregação de RSS. 5. Metadados. 6. Sistemas de recuperação da informação. 7. Filtragem da informação. 8. Taxonomia (Computação). I. Tanaka, Astério Kiyoshi. II. Siqueira, Sean Wolfgand Matsui. III. Universidade Federal do Estado do Rio de Janeiro (2003-). Centro de Ciências Exatas e Tecnologia. Curso de Mestrado em Informática. IV. Título.

CDD – 004.2

Dedicatória

Dedico este trabalho aos meus orientadores, que forneceram grande suporte. Dedico também à minha namorada e à minha família, que me ajudaram a suportar momentos difíceis e tornaram esses momentos mais suportáveis.

Agradecimentos

Agradeço ao meu orientador Astério Kiyoshi Tanaka e ao co-orientador Sean Wolfgand Matsui Siqueira pelo estímulo e colaboração na minha formação. Pela amizade e confiança no meu trabalho e pelos ensinamentos e orientação que foram fundamentais na realização desse trabalho.

Agradeço a todos os professores do PPGI/Unirio que estavam sempre dispostos a ajudar e contribuir na formação dos alunos dessa instituição e, em especial, aos professores Fernanda Araújo Baião, Márcio de Oliveira Barros, Ângelo Ernani Maia Ciarlini e Flávia Maria Santoro, pelas aulas e colaboração ao longo do caminho. Aos colegas da Unirio que contribuíram com suas dúvidas, sugestões e apoio. E agradeço principalmente à minha família, meu pai Luiz Carlos Ferreira, minha mãe Mariele Nonato Canevello Ferreira, à minha irmã Monique Canevello Ferreira e a minha avó Jandira Rosa Ferreira.

Obrigado a todos que cruzaram a minha caminhada, pois ninguém vence sozinho.

FERREIRA, Marcelo Canevello. **iRSS - Uma Ferramenta de Agregação de RSS baseada em Taxonomia**. UNIRIO, 2009. 114 páginas. Dissertação de Mestrado. Departamento de Informática Aplicada, UNIRIO.

RESUMO

A publicação de RSS normalmente é um serviço contínuo. Por isso, o resultado da agregação de RSS pode ocasionar problemas comuns na área de recuperação de informação: a abundância de informação e a falta de relevância. Para tratar o iminente crescimento de conteúdo, os agregadores de RSS procuram utilizar filtros e capturar os interesses do usuário, a fim de agregar apenas informação relevante aplicando-se algum tipo de inteligência. O processo de agregação de RSS descrito neste trabalho tem como enfoque o emprego de técnicas de recuperação de informação, apoiado numa taxonomia. Este processo utiliza termos e conceitos extraídos do RSS com o objetivo de permitir ao usuário a escolha de assuntos considerados relevantes a serem utilizados na filtragem de informação agregada. Foi desenvolvida a ferramenta iRSS segundo essa abordagem, utilizando como estudo de caso a taxonomia da ACM/IEEE para computação e os canais de RSS da biblioteca digital do IEEE.

Palavras-Chave: Sistema de Informação, RSS, Agregação de RSS, Metadados, Recuperação da Informação, Filtragem da Informação, Taxonomia.

ABSTRACT

RSS publishing uses to be a continuous service. Hence, the result of RSS aggregation may cause problems that are usual in the information retrieval area: information overload and lack of relevance. In order to address the imminent growth of content, RSS aggregators try to use filters and to capture the user's interests to aggregate only relevant information by applying some kind of intelligence. The process of RSS aggregation described in this work focuses on the use of techniques of information retrieval, supported by a taxonomy. This process uses terms and concepts extracted from RSS, aiming at allowing the user to choose topics deemed relevant to be used in the filtering of aggregated information. The iRSS tool was developed according to this approach, using as a case study the ACM / IEEE computing taxonomy and the RSS channels from the IEEE digital library.

Keywords: Information System, RSS, RSS aggregation, Metadata, Information Retrieval, Information Filtering, Taxonomy.

Índice do Texto

Índice de Figuras

Índice de Tabelas

Abreviatura e Siglas

API	Application Programming Interface
HTML	Hyper Text Markup Language
W3C	World Wide Web Consortium
RD	Recuperação de Dados
RDF	Resource Description Framework
REST	Representational State Transfer
RI	Recuperação de Informação
RSS	Really Simple Syndication
SRI	Sistema de Recuperação de Informação
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VSM	Vector Space Model
XML	Extensible Markup Language
XPATH	XML Path Language
XSL	Extensible Stylesheet Language

1. Introdução

Este capítulo apresenta o contexto, a motivação, os objetivos e a organização da dissertação. Inicialmente é apresentada uma breve visão geral sobre RSS e recuperação de informação na seção 1.1, destacando-se o contexto e a motivação. Em seguida, são expostos os objetivos do trabalho na seção 1.2, onde é abordada a proposta de agregação com base numa taxonomia. Por fim, a seção 1.3 descreve a organização da dissertação.

1.1 Contexto e Motivação

A Internet diariamente ganha mais importância devido ao desenvolvimento de novas tecnologias que dela necessitam para existir. As aplicações da Internet são mais populares a cada dia. As pessoas se relacionam, trocam experiências e informações através da grande rede. Uma das tecnologias que permite o acesso ao conteúdo da Internet é o RSS - *Really Simple Syndication*.

Na prática, RSS é largamente utilizado por agências de comunicação (ou qualquer instituição ou pessoas, citados ao longo da dissertação como “publicadores de conteúdo”) como um facilitador para o acesso a notícias. Segundo KING (2009), RSS é um vocabulário de baixa complexidade para a descrição de mensagens, notícias e

informações (metadados - dados que descrevem outros dados) disponíveis em “Web sites”, ideal para divulgação de notícias. Nesse contexto, RSS provê uma forma de distribuir conteúdo de maneira consistente, que pode ser entendida por máquinas, permitindo que “Web sites” compartilhem conteúdo com outras aplicações, seguindo um padrão (HAMMERSLEY, 2005).

A ampla utilização do RSS como meio de distribuição de conteúdo propicia a ocorrência de anomalias já observadas pela área de Recuperação de Informação – RI (apresentada no capítulo 2), tais como o excesso de informação e a baixa relevância do conteúdo recuperado por agregadores de RSS (sistemas que recuperam, exibem e permitem redistribuição de RSS). Segundo HRASTNIK (2005), o risco de excesso de informação e a conseqüente diminuição da relevância do conteúdo agregado aumentam quando o número de publicadores de RSS disponíveis aumenta devido à facilidade de subscrição a esses publicadores.

O objetivo deste trabalho é minimizar o problema de excesso de informação e o problema de falta de relevância do RSS agregado. Assim, busca-se um processo que propicie a filtragem do conteúdo entregue aos agregadores de RSS, que amenize os problemas previamente citados. Este processo é aplicado numa ferramenta localizada entre o publicador do conteúdo e o consumidor ao conteúdo proveniente dos publicadores. O processo é composto por etapas que têm como objetivo recuperar o RSS, extrair informação e classificá-la na taxonomia, fornecendo um conteúdo re-utilizável e filtrado de acordo com a classificação na taxonomia.

Dada a estreita relação entre o tradicional problema de falta de relevância de resultado existente em RI e a agregação de RSS, o processo apresentado nesta dissertação descreve o emprego de técnicas tradicionais da área de RI. Tais técnicas são empregadas para extrair termos relevantes e utilizá-los como uma coleção sobre a qual

se aplicam consultas contendo os tópicos recursivamente obtidos na taxonomia, recuperando-se dinamicamente informações (notícias ou mensagens) do RSS.

Tal processo permite que o conteúdo proveniente de inúmeros publicadores de conteúdo seja processado, filtrado e reprocessado por outros agregadores que também utilizem o processo proposto ou outros processos de agregação. Embora a abordagem proposta possa incitar discussões sobre direitos autorais, esta dissertação preocupa-se apenas com a questão do processo de agregação. Casos publicamente conhecidos como o processo movido pela agência de notícias Associated Press[1] e o buscador Google[2] mostram que a questão dos direitos autorais pode influenciar o desenvolvimento de novas tecnologias, mas isso não está no escopo desta dissertação.

1.2 Objetivos da Dissertação

O elevado número de notícias publicadas diariamente e disponibilizadas por meio de RSS torna necessário o aprimoramento das formas de agregação. Assim, o principal objetivo desse trabalho é propor um processo de agregação de RSS que se preocupe em agregar conteúdo relevante, diminuindo o excesso de informação.

Para alcançar este objetivo, os seguintes objetivos específicos são perseguidos:

- Propor um processo de agregação de RSS que realize a filtragem do conteúdo e permita a reutilização deste conteúdo;
- Aplicar o processo proposto através do desenvolvimento de um protótipo como prova de conceito para avaliação do processo como um todo.

[1] www.ap.org
[2] www.google.com

1.3 Organização da Dissertação

A dissertação está dividida em cinco capítulos. Este, capítulo 1, tratou de apresentar o contexto e a motivação do trabalho, bem como os objetivos almejados. O capítulo 2 apresenta uma revisão bibliográfica sobre RSS e Recuperação de Informação. O capítulo 3 descreve a proposta de agregação de RSS com base numa taxonomia. O capítulo 4 descreve a construção e validação do processo de agregação proposto. Este processo é validado através do protótipo iRSS, uma ferramenta que implementa a proposta de agregação apresentada no capítulo anterior. Por fim, o capítulo 5 apresenta as conclusões sobre o trabalho desenvolvido, as contribuições da pesquisa realizada, e sugere alguns possíveis trabalhos futuros.

2. Fundamentação Teórica

Este capítulo trata de RSS, recuperação de informação e conceitos fundamentais que servem de embasamento para a compreensão do trabalho proposto nesta dissertação. A seção 2.1 define o conceito de RSS, apresenta sua estrutura, descreve sua utilização, classifica os agregadores de RSS e cita a importância de RSS para os usuários da Internet. A seção 2.2 define recuperação de informação, descrevendo também índices invertidos, modelos de recuperação de informação e mecanismos de indexação e busca em documentos XML. E, por fim, a seção 2.3 apresenta algumas considerações finais sobre o conteúdo abordado neste capítulo.

2.1 RSS

RSS foi desenvolvido a partir de idéias provenientes do formato Scripting News[3] em 1999 pela Netscape[4] para o *Web site* my.netscape.com com o objetivo de permitir que as notícias publicadas no *site* pudessem ser apresentadas em outros *Web sites*. Atualmente, RSS continua sendo usado para disponibilizar notícias, mas também é

[3] http://davenet.scripting.com/1997/12/15/scriptingNewsInXML
[4] http://netscape.aol.com/

utilizado em Blogs (como o Blogger[5]) e mais recentemente em micro-blogs (como o Twitter[6]).

O significado para RSS mudou algumas vezes desde sua concepção. Cada uma das definições advém das diferentes versões do RSS. Os números das versões e suas respectivas definições são:

- 0.90, 1.0 – *RDF Site Summary*
- 0.91, 0.92, 0.93, 0.94 – *Rich Site Summary*
- 2.0 – *Really Simple Syndication*

Formalmente, RSS (neste trabalho é o acrônimo para *Really Simple Syndication*[7]) é um esquema em XML que define um dialeto simples para o encadeamento de pequenas mensagens (como trechos de notícias e artigos) de modo que possam ser divulgadas por publicadores de conteúdo e disponibilizadas a páginas Web. Segundo HAMMERSLEY (2005), tais consumidores do conteúdo disponibilizado seriam inicialmente *Web sites*, contudo o objetivo inicial foi desvirtuado e atualmente o conteúdo é recuperado por sistemas conhecidos como agregadores.

RSS é um vocabulário de baixa complexidade facilmente assimilado e largamente utilizado por agências de notícias. Sua simplicidade é destacada pela presença da palavra *Simple* em seu acrônimo, uma vez que sua estrutura contém poucas restrições, como é exemplificado na seção 2.1.1. Ele permite que notícias sejam disponibilizadas e continuamente monitoradas por agregadores de RSS, os quais fornecem uma interface onde o leitor pode ler trechos de notícias e opcionalmente clicar em *links* atribuídos às notícias que apontam para a notícia completa.

[5] http://www.blogger.com
[6] http://twitter.com
[7] Especificado em RSS (RSS Board, 2008) e RSS (W3C, 2009).

2.1.1 Estrutura

Um documento RSS 2.0 é formado por elementos (*tags* ou marcadores) organizados numa estrutura hierárquica, que possui elementos obrigatórios e opcionais. A estrutura de um documento RSS 2.0 é apresentada abaixo e contém: um elemento com a indicação da versão de XML empregada <?xml version="1.0"?>, seguida pelo elemento RSS <rss>, que delimita o conteúdo do RSS propriamente dito. Em seguida, há um sub-elemento do tipo <channel>, que indica o canal de notícias (um publicador de conteúdo pode fornecer um ou mais canais de RSS) e possui obrigatoriamente um elemento <title> para descrever o título do canal de notícias, um <link> para indicar a URI do canal e um <description> contendo a descrição deste canal, seguido opcionalmente pelo elemento <item>, que são as notícias do canal e devem conter ao menos um elemento <title>, <link> ou <description>; como apresentado no documento XML de exemplo retirado de (RSS Board, 2008) a seguir.

```
<?xml version="1.0"?>
<rss version="2.0">
   <channel>
      <title>Liftoff News</title>
      <link>http://liftoff.msfc.nasa.gov/</link>
      <description>Liftoff to Space Exploration.</description>
      <item>
         <title>Star City</title>
         <link>http://liftoff.msfc.nasa.gov/news/2003/news-starcity.asp</link>
         <description>How do Americans get ready to work with Russians aboard the
International Space Station? They take a crash course in culture, language and protocol
at Russia's &lt;a href="http://howe.iki.rssi.ru/GCTC/gctc_e.htm"&gt;Star City&lt;/a&gt;.
    </description>
         <pubDate>Tue, 03 Jun 2003 09:39:21 GMT</pubDate>
         <guid>http://liftoff.msfc.nasa.gov/2003/06/03.html#item573</guid>
      </item>
   </channel>
</rss>
```

2.1.2 Agregadores

A apresentação do RSS ao usuário é possibilitada através de um agregador (ou *reader*), que pode monitorar arquivos RSS provenientes de inúmeros publicadores de conteúdo. A arquitetura de um sistema que utiliza RSS é composta por um publicador

de conteúdo (*feeder*) que expõe, através de uma URL (endereço do canal), um arquivo no formato XML (seguindo a codificação do RSS), o qual é disponibilizado a qualquer programa agregador de RSS, que pode capturar os itens do RSS, formatá-los e apresentá-los ao usuário final. Assim, o usuário pode ler o sumário (<description>) ou ler o conteúdo na íntegra a partir do <link> de um <item> do RSS. Para utilizar um agregador e ter acesso aos itens em RSS, o usuário final subscreve (ou assina) um publicador inserindo o *link* (URL) do canal (<channel>) no agregador, a partir de onde ele recebe as atualizações do RSS.

2.1.2.1 Arquiteturas de agregadores

Assim como o navegador está para a página HTML, o agregador (*reader*) está para o arquivo RSS; e da mesma forma que o servidor Web provê página HTML, o publicador (*feeder*) provê RSS. Um agregador pode ser uma página da Web hospedada num servidor, um programa instalado no computador do usuário final ou um programa instalado no servidor que funciona como um publicador de RSS, fornecendo RSS agregado a outros agregadores.

Agregadores em páginas Web (Figura 2.1) monitoram publicadores de conteúdo, recuperam e formatam o RSS na forma de páginas HTML. A transformação do RSS para HTML pode ser realizada através da transformação (*parsing*) do XML por meio de API ou através da aplicação de XSL[8], uma família de recomendações do W3C para transformação e apresentação de XML. Assim, a partir do resultado da transformação do RSS, o usuário final pode acessar o conteúdo original.

[8] *Extensible Stylesheet Language* – W3C, 2009.

Figura 2.1 – Agregador numa página da Web, hospedada num servidor

Outra arquitetura existente (Figura 2.2) é aquela onde um programa é instalado no computador do usuário final. Nela, o publicador do conteúdo fornece RSS diretamente ao agregador do usuário final. Este agregador transforma e apresenta o RSS ao usuário e, assim como o agregador Web, permite ao usuário acessar diretamente o conteúdo original através do *link* contido no RSS. Ao clicar no *link* do RSS, o agregador executa o navegador padrão do computador do usuário final, passando a URL do *link* como parâmetro, ou abre o conteúdo dentro do próprio agregador através de um navegador embutido.

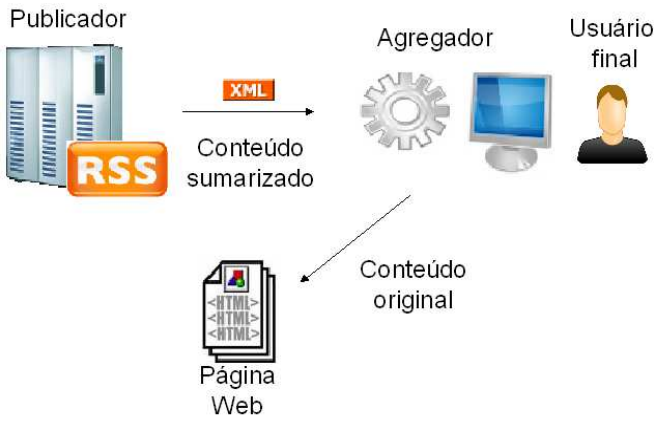

Figura 2.2 – Agregador instalado no computador do usuário final

A arquitetura inicialmente definida pelos criadores do RSS, segundo HAMMERSLEY (2005), mais tarde serviu de base para (PETROVIC *et al.*, 2005) apresentar a arquitetura de agregador intermediário, conforme a figura 2.3. Esta arquitetura permite a aplicação de filtros e classificadores ao conteúdo proveniente de vários publicadores, através de um pré-processamento das publicações realizado por um agregador intermediário. O usuário final subscreve ao agregador intermediário e recebe o conteúdo resultante do processamento. No caso deste trabalho, o capítulo 3 apresenta um processo de agregação baseado nesta arquitetura, propondo um processo de agregação que se encaixa no papel do agregador intermediário.

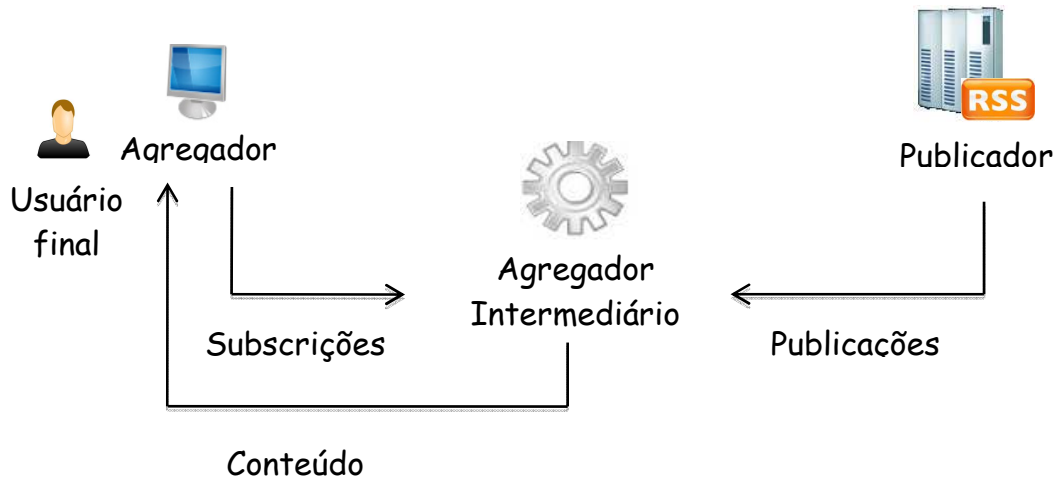

Figura 2.3 – Agregador com etapa de agregação intermediária – adaptado de (PETROVIC *et al.*, 2005)

A arquitetura criada por PETROVIC *et al.*(2005) permite a filtragem de grandes quantidades de publicações. Em seu agregador (CMS-ToPSS) é aplicado um algoritmo de busca em grafos, diferentemente de outras abordagens, como (ALTINEL e FRANKLIN, 2000) e (DIAO *et al.*, 2002), que fazem uso de XPath[9] (W3C, 1999) para recuperar conteúdo no formato XML.

2.1.2.2 Classificações de agregadores

Apresentadas as arquiteturas de agregadores, a seguir são classificados os tipos de agregadores. Na classificação de agregadores proposta neste trabalho, toma-se como base a capacidade de classificação de conteúdo para distinguir as categorias de agregadores.

A primeira geração de agregadores é a mais comum, pois não há classificação do conteúdo (normalmente notícias) recebido; o agregador apenas acumula o conteúdo, como em: RssReader[10], Blam Feed Reader[11] e ThinFeeder[12]. Este tipo de agregador, que

[9] Linguagem para endereçar partes de um documento XML.
[10] www.rssreader.com
[11] www.cmartin.tk/blam.html
[12] thinfeeder.sourceforge.net

propicia a acumulação (leia-se agregação) de conteúdo de fontes distintas ao longo do tempo sem qualquer tipo de classificação, expõe o problema de excesso de informação e o problema de falta de relevância do conteúdo agregado.

Devido aos problemas encontrados na primeira geração de agregadores, estes passaram a permitir a classificação de canais de RSS. A classificação está presente na segunda geração de agregadores, como: Google Reader[13], Akregator[14], FeedReader[15], Active Web Reader[16] e RSS Bandit[17], entre outros. Nesta categoria, os canais são separados por assunto, permitindo ao usuário final conhecer conteúdos relacionados.

Separar os canais apenas por assunto e incluí-lo numa categoria amenizou, mas logo se mostrou ineficiente para agregação de RSS. Uma vez que há canais de RSS com conteúdo referente a diferentes tópicos num dado domínio, o usuário acaba tendo acesso a conteúdo de baixa relevância. Tal dificuldade incentivou a criação de uma terceira geração de agregadores de RSS, onde se procura aplicar filtros que correspondam aos interesses do usuário.

2.1.3 Relevância do conteúdo agregado

Inicialmente, os agregadores de RSS apenas acumulavam conteúdo sem se preocupar com a qualidade. Com o crescimento exponencial da informação disponibilizada aos usuários, tornou-se necessário o emprego de técnicas mais sofisticadas, o que propiciou o desenvolvimento de inúmeros tipos de agregadores (apresentados na seção 2.1.2.2) implementados sobre diferentes arquiteturas (apresentadas na seção 2.1.2.1). Entre as técnicas utilizadas para minimizar o excesso de informação, a que mais se destaca é a filtragem de informação, porque é usada para

[13] www.google.com/reader
[14] akregator.kde.org
[15] www.feedreader.com
[16] www.deskshare.com
[17] www.rssbandit.org

fornecer conteúdo relevante. A filtragem é uma ferramenta de auxílio à recuperação de informação em grandes coleções de documentos, permitindo o atendimento às necessidades do usuário.

O termo "relevância" pode assumir diferentes interpretações. Para PINHEIRO *et al.* (2009), é considerado relevante qualquer item de RSS que contenha a palavra chave selecionada pelo usuário. No contexto deste trabalho, também é considerado relevante qualquer item de RSS que contenha o termo presente na taxonomia utilizada no processo de agregação. Apesar do conceito de termo ou palavra chave determinar a relevância de um determinado item, existem outras definições para relevância, como a data ou uma função com diferentes variáveis, cada uma com um determinado peso. Esta última definição de relevância é encontrada no AideRSS[18], onde é utilizada uma função (índice) para medir o grau de qualidade do item. Esse índice é chamado de PostRank[19], e mede a qualidade de um item de acordo com sua relevância e a reação dos usuários. A relevância é medida através da quantidade de indexações realizadas por usuários no sistema del.icio.us[20] e da quantidade de referências encontradas no *site* de busca Google[21]; enquanto a reação dos usuários é medida através do número de comentários relacionados ao item no *site* Digg[22].

2.2 Recuperação de informação

O agregador que segue o processo de agregação de RSS proposto neste trabalho é classificado como um Sistema de Recuperação de Informação – SRI. Para chegar a tal conclusão, esta seção especifica SRI e em seguida onde se evidenciam as características

[18] www.aiderss.com
[19] www.postrank.com
[20] delicious.com
[21] www.google.com
[22] digg.com

do SRI da proposta deste trabalho. Além disso, o problema de excesso de informação e o problema da falta de relevância do conteúdo recuperado são problemas da área de Recuperação de informação (RI), comumente encontrados em agregadores de RSS. A seguir são apresentados conceitos de RI utilizados na proposta apresentada no capítulo 3.

RI é a área que trata da representação, armazenamento, organização e acesso a itens de informação (MIRANDA, 2003). Segundo MANNING *et al.* (2008), RI é a busca por material (documentos) de natureza semi-estruturada ou não estruturada (normalmente texto), que satisfaça à necessidade do usuário por informação contida em grandes coleções (normalmente armazenadas em computadores).

Num SRI, os documentos são indexados e transformados em estruturas de dados que os representam; resultado de um processo de indexação, onde se busca produzir subsídios para a execução de uma consulta, que por sua vez representa o resultado do processo de especificação de consulta, onde o usuário expõe suas necessidades. Porém, a distância semântica entre o processo de especificação de consulta e a real necessidade do usuário expressa sob a forma de uma consulta, normalmente não casam com o que é realizado durante o processo de indexação, proporcionando a perda de informação. Apesar da inevitável perda de informação, o processo de recuperação de informação atua de modo a facilitar o consumo de informação, ao retornar uma lista de documentos ordenada de forma decrescente de acordo com as necessidades do usuário (expressas através da consulta). Tais componentes formam um sistema de recuperação de informação, conforme a Figura 2.4 (CARDOSO, 2000).

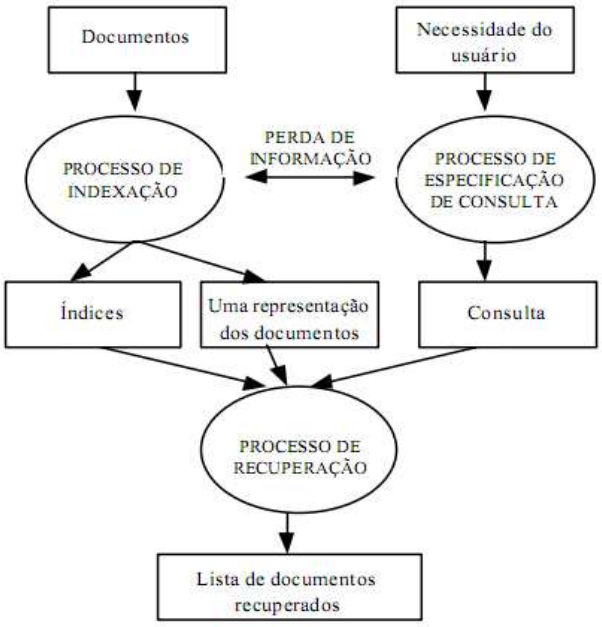

Figura 2.4 – Componentes de um sistema de recuperação de informação (CARDOSO,2000)

A proposta apresentada neste trabalho trata RSS como documentos que passam por um processo de indexação, onde são construídos índices e persistidos numa determinada representação. Após a indexação, ocorre o processo de recuperação de informação, tendo como entrada uma lista de termos recursivamente selecionados de uma taxonomia. Devido às características apresentadas, conclui-se que a afirmação no início desta seção, de que a proposta deste trabalho é um SRI está correta.

Além disso, a proposta apresentada neste trabalho se preocupa em produzir como resultado final um conjunto de documentos relevantes, diferentemente do resultado encontrado em Recuperação de Dados (RD) (MIRANDA, 2003).

RI é comumente confundida com RD, visto que a principal diferença entre as duas áreas se dá quanto ao resultado de uma consulta, pois enquanto a recuperação de dados busca encontrar a informação exata, a recuperação de informação busca encontrar a informação que melhor se adéqua aos anseios do usuário. A RD retorna todos os documentos usando consultas que satisfazem claramente a condições que estão bem

definidas em sua estrutura e em sua semântica, ao inverso da RI, que trata de informações textuais em linguagem natural, onde as consultas não são bem definidas estruturalmente e muitas vezes não estão definidas na sua semântica. Outra diferença ocorre quanto ao erro; em RI um erro em milhares de respostas é algo aceitável, enquanto em RD, um erro implica em falha total.

2.2.1 Índices invertidos

O processo de indexação num SRI tem como premissa a criação de estruturas de dados que permitam a recuperação rápida de documentos contidos em grandes coleções. Tradicionalmente, índice invertido (ou arquivo invertido) é a estrutura de dados mais utilizada no desenvolvimento de SRI. Índices invertidos (MANNING *et al.*, 2008) são estruturas de dados (também conhecidos como listas de posições ou arquivos invertidos) que facilitam a busca em textos (ou *Full Text Search* – FTS). Evitando a busca linear, eles mapeiam o conteúdo às ocorrências em textos. Por isso são largamente empregados em SRI.

Segundo MANNING *et al.* (2008), a construção de um índice invertido para um SRI passa pelos seguintes passos:

1. Coleta dos documentos a serem indexados;
2. Descarte de *stopwords*, isto é, a extração de *tokens* (termos) com elevada freqüência. Normalmente são elementos da gramática da língua, utilizados no texto, que não definem conceitos, como artigos, conjunções e advérbios que fazem parte da lista de *stopwords* ou simplesmente *stoplist* (MIRANDA, 2003);
3. Redução dos termos a uma forma única através de algoritmos de conflação. Segundo FRAKES *et al.* (1992), conflação é o ato de fundir ou combinar variantes morfológicas de termos. Segundo SPARK-

JONES, (1997) existem duas formas de conflação: o *stemming* e a redução à forma canônica. Segundo ARAMPATZIS *et al.* (2000), redução à forma canônica consiste em reduzir os verbos para o infinitivo, e os substantivos e adjetivos para a forma masculina singular. De acordo com (CHAVES, 2003) essa redução faz com que entradas que significam a mesma coisa tenham a mesma representação de significado. A outra forma de conflação, o *stemming* (forma de conflação utilizada neste trabalho), segundo SPARK-JONES (1997), consiste em reduzir todas as palavras ao mesmo *stem* (conjunto de caracteres resultante do stemming – STRZALKOWSKI, 1999), por meio da retirada dos afixos da palavra, permanecendo apenas a raiz. Para HONRADO *et al.* (2000), o propósito do *stemming* é chegar a um *stem* que captura uma palavra com generalidade suficiente para permitir o sucesso na combinação de caracteres, mas sem perder o detalhe da precisão.

4. Indexação dos documentos onde cada termo ocorre, com a criação do índice invertido, consistindo de: uma lista de termos, o número de documentos onde os termos ocorrem, o número total de ocorrências do *token* na coleção de documentos (ou *corpus*) e as respectivas listas de documentos (consistindo de um *docId* ou código identificador do documento) e as posições onde os termos ocorrem, como mostra a Tabela 2.1.

Tabela 2.1: Trecho de um índice invertido (retirado de MIRANDA, 2003)

Token	Número de documentos	Total de ocorrências	(Documento: posições)
need	2	3	(1:48)(3:6, 23)
separ	1	1	(1:7)
industri	2	5	(1:45)(3:6, 23, 51, 70)

Índices invertidos podem ser comprimidos através do emprego de tesauros (*Thesaurus* ou simplesmente lista de palavras com significados semelhantes), onde se procura encontrar sinônimos das palavras do texto, permitindo o uso de apenas um termo para os *tokens* equivalentes (sinônimos); permitindo a economia de espaço (MIRANDA, 2003).

Apesar do índice invertido permitir a busca eficiente de documentos, sua atualização torna-se lenta com o passar do tempo (RIJSBERGEN *et al.*,1979). Para minimizar lentidão durante a atualização do índice invertido, a tarefa de atualização é realizada em segundo plano nos SRI.

2.2.2 Modelos de recuperação de informação

Os modelos de RI são estratégias de busca de documentos relevantes para as consultas realizadas pelos usuários (MANNING *et al.*, 2008). Tais modelos definem como o SRI deve recuperar os documentos. Segundo KUROPKA (2003), os modelos podem ser classificados como: modelos baseados na teoria dos conjuntos, modelos algébricos e modelos probabilísticos. Segundo BAEZA-YATES *et al.* (1999), WIVES *et al.* (2001) e REZENDE (2002), os modelos clássicos de RI são: modelo booleano, modelo vetorial e modelo probabilístico. A seguir são apresentadas as definições dos três modelos.

O modelo booleano representa um documento como um conjunto de termos (SALTON, 1983). Neste modelo, uma consulta consiste de um conjunto de termos combinados com operadores de Boole (AND, OR e NOT), formando expressões. As expressões booleanas são capazes de unir conjuntos, descrever intersecções e retirar partes de um conjunto. Em uma busca, por exemplo, o usuário indica quais são as palavras (elementos) que o documento (conjunto) resultante deve ter para que seja

retornado. Assim, os documentos que possuem interseção com a consulta (mesmas palavras) são retornados. Os documentos podem ser ordenados pelo grau de interseção, onde o mais alto é aquele que contém todas as palavras especificadas na consulta do usuário, e o mais baixo o que contém somente uma.

O modelo vetorial, também conhecido como *Vector Space Model* - VSM, representa os documentos como vetores de termos seguidos por seus respectivos pesos (*weight*) (SALTON, 1989). Portanto, cada documento possui um vetor associado que é constituído de tuplas de termos na forma (termo 1, peso 1), (termo 2, peso 2)...(termo n, peso n). Nesse vetor, são representados todos os termos da coleção de documentos e não somente os termos presentes no documento. Os termos que o documento não contém recebem peso zero, enquanto os demais são calculados através de uma fórmula de cálculo de peso. Os pesos próximos de um (1) indicam termos relevantes e pesos próximos de zero (0) indicam termos irrelevantes (podendo também variar entre -1 e 1).

O peso de um termo em um documento pode ser calculado de diversas formas. Para SALTON *et al.* (1987), os métodos de cálculo de peso baseiam-se no número de ocorrências do termo no documento. Os métodos mais utilizados no cálculo de peso são a freqüência do termo (*tf - term frequency*) e a freqüência inversa do termo no documento (*tf-idf - term frequency – inverted document frequency*).

Devido a sua simplicidade, o modelo vetorial e suas variantes são muito utilizados para representar documentos textuais na mineração de textos (LOPES, 2004). Operações de vetores podem ser executadas muito rapidamente e existem algoritmos padronizados eficientes para realizar a seleção do modelo, a redução da dimensão e a visualização de espaços de vetores.

Segundo BAEZA-YATES (1999), o modelo probabilístico tenta recuperar informações usando a teoria da probabilidade. Ou seja, dada uma consulta do usuário,

há um conjunto de documentos que contém exatamente os documentos relevantes, e nenhum outro. Segundo CARDOSO (2000), o resultado de uma consulta é representado pelo conjunto R, que é obtido através do cálculo da probabilidade de todos os documentos da coleção ser relevante em relação a uma consulta. Um documento é considerado relevante quando a probabilidade dele ser relevante para uma consulta é maior que a probabilidade dele ser irrelevante.

2.2.2.1 Outros modelos de recuperação de informação

Além dos três modelos clássicos de RI, outros modelos foram criados para recuperação de informação. Entre os modelos pesquisados vale destacar: o modelo booleano estendido (SALTON, 1983), o modelo vetorial booleano baseado em tópicos estendido (KUROPKA, 2003), o modelo *fuzzy* para RI utilizando múltiplas ontologias relacionadas (LEITE, 2009). O modelo booleano estendido une o modelo booleano tradicional e o cálculo de relevância do modelo vetorial para ordenação, suplantando o problema de indecisão binária de relevância. O modelo vetorial booleano baseado em tópicos estendidos (também conhecido como *enhanced Topic-based Vector Space Model – eTVSM*) representa um avanço em relação ao *Topic-based Vector Space Model - TVSM*, pois ele permite expressar outros fenômenos lingüísticos além da sinonímia, como a metonímia e a homografia. Sua especificação leva em conta os conceitos, além dos termos para o cálculo da similaridade. Por fim, o modelo *fuzzy* utilizando múltiplas ontologias relacionadas, proposto em LEITE (2009), emprega técnicas da teoria de conjuntos *fuzzy* para tratar a imprecisão e a incerteza presentes no conhecimento e no processo de recuperação de informação. Neste modelo, a organização do conhecimento e o método de expansão de consulta foram integrados no modelo *fuzzy* para recuperação de informação utilizando múltiplas ontologias relacionadas.

2.2.3 Avaliação de sistemas de recuperação de informação

SRI são avaliados quanto à relevância dos resultados obtidos após a execução de um conjunto de consultas (RIJSBERGEN, 1979). Segundo BAEZA-YATES (1999), existem duas medidas para avaliar o desempenho de um SRI: abrangência e precisão. Para que estas medidas possam ser avaliadas, o sistema ou usuário que analisa os resultados deve saber quantos documentos relevantes à consulta existem na coleção (LEITE, 2009). Assim, dada a consulta *I* definida pelo usuário a uma coleção contendo o conjunto *R* de documentos relevantes (sendo |*R*| o número de documentos neste conjunto), a estratégia de busca a ser testada processa a consulta *I* e retorna um conjunto de documentos *A* como resposta (sendo |*A*| o número de documentos do conjunto *A*). O número de documentos relevantes é dado por $|R_a|$, que corresponde ao número de documentos da interseção dos conjuntos *R* e *A* (figura 2.5).

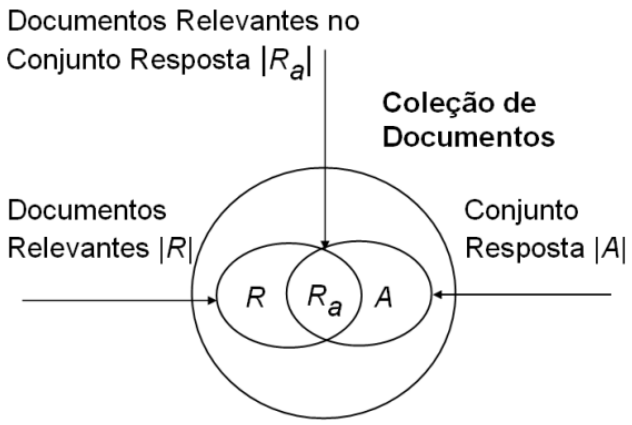

Figura 2.5 – Conjuntos *R*, *A* e R_a. Adaptado de (BAEZA-YATES, 1999)

2.2.3.1 Abrangência (*recall* ou cobertura)

Segundo BAEZA-YATES (1999), abrangência *r* é a fração do número de documentos relevantes recuperados $|R_a|$ pelo número total de documentos relevantes na coleção |*R*|, ou seja, Abrangência = $| R_a | / |R|$.

2.2.3.2 Precisão (*precision*)

Segundo BAEZA-YATES (1999), precisão *p* é a fração do número de documentos relevantes recuperados | R_a | pelo número total de documentos recuperados |*A*|, ou seja, Precisão = | R_a | / |*A*|.

2.2.3.3 Medida F (*F-measure*)

Segundo RIJSBERGEN *et al.* (1979), a medida *F* é uma média harmônica entre a abrangência (*r*) e a precisão (*p*) ou seja, $F = (2pr) / (p+r)$.

2.2.4 Mecanismos de indexação e busca em documentos XML

Segundo LUK *et al.* (2000), existem três abordagens para indexação e busca de documentos XML: a abordagem orientada a banco de dados, a abordagem orientada à RI e uma abordagem híbrida.

2.2.4.1 Abordagem orientada a banco de dados

Nesta abordagem, os metadados dos documentos XML são mapeados em tabelas e colunas, e os dados são armazenados em registros. Dessa forma, os dados posteriormente podem ser compilados para formar novos documentos XML. A abordagem orientada a banco de dados apresenta as seguintes vantagens: (i) as relações dos dados e suas restrições de integridade podem ser modeladas e checadas; (ii) as operações padrão (de definição e manipulação) de banco de dados podem ser usadas; (iii) a linguagem de consulta em geral é similar à linguagem padrão de banco de dados (SQL).

2.2.4.2 Abordagem orientada à recuperação de informação

Na abordagem orientada à RI, técnicas de RI são diretamente aplicadas para processar e recuperar documentos XML, cada qual é considerado um documento texto

com marcações (*tags*) adicionais. Vários métodos têm sido sugeridos para lidar com as marcações XML. Estes incluem simplesmente descartar todas as marcações; selecionar algumas marcações importantes e inseri-las no índice; e indexar todas as marcações como se elas fossem termos de índice (LUK *et al.*, 2000).

2.2.4.3 Abordagem híbrida

Na abordagem híbrida, um método mais preciso é usado para especificar os elementos aninhados usando expressões de caminho – através de XPath (XPath, 1999). A idéia básica é que listas invertidas de todos os termos e suas localizações (através de seus XPath associados) sejam indexadas. Assim, os resultados das buscas são mais satisfatórios, desde que a especificação dos XPath limite os casamentos possíveis. A desvantagem é a necessidade do usuário possuir algum conhecimento adicional sobre XPath para obter melhores resultados (LUK *et al.*, 2000).

2.3 Considerações

Neste capítulo foram apresentados os conceitos fundamentais para compreensão do trabalho desta Dissertação. Abordou-se a questão do RSS, que devido à sua abrangência e simplicidade, tornou-se um importante facilitador de difusão da informação disponível na Web. Em seguida, foi abordado o conceito de RI, seus modelos (neste trabalho é utilizado o modelo vetorial) e as formas de avaliação de um SRI. Por fim, foram apresentados os mecanismos de indexação e busca em documentos XML, uma vez que neste trabalho a abordagem empregada é a abordagem orientada a banco de dados.

Os conceitos citados servem como embasamento teórico para entendimento do processo proposto no capítulo 3. Este processo se aplica a agregadores que

implementam a arquitetura proposta por PETROVIC *et al.* (2005), pois esta arquitetura permite a re-utilização do conteúdo filtrado. O processo é validado na construção do protótipo iRSS, um SRI com os componentes especificados por CARDOSO (2000), mas que aplica os termos da taxonomia como sendo a necessidade do usuário. Este SRI, o iRSS, é descrito no capítulo 4.

O iRSS faz uso de índices invertidos, por permitirem a recuperação eficiente (rápida) do RSS recuperado. Associado ao índice invertido, este agregador utiliza o modelo vetorial clássico devido à sua simplicidade de integração e eficiência na recuperação de conteúdo em larga escala. Assim, a agregação e filtragem do iRSS é analisada no capítulo 4, que utiliza um conjunto de canais de RSS publicados pela biblioteca digital do IEEE.

3. Proposta de Agregação de RSS

Este capítulo apresenta uma proposta para minimizar o problema de excesso de informação e o problema de falta de relevância do RSS agregado. Esta proposta se propõe a realizar um pré-processamento do conteúdo entregue pelos publicadores de conteúdo aos agregadores de RSS dos usuários. Trata-se de um processo de agregação intermediária de RSS que emprega técnicas de processamento de linguagem natural e recuperação de informação para classificar o conteúdo com o apoio de uma taxonomia. A seção 3.1 traça uma visão geral sobre a técnica de agregação proposta, descrevendo o processo e cada uma de suas etapas. A seção 3.2 discorre sobre os motivos do emprego das técnicas das áreas supracitadas. E, por fim, a seção 3.3 cita algumas considerações finais sobre o conteúdo abordado neste capítulo.

3.1 Processo proposto para agregação de RSS

A filtragem e re-distribuição do conteúdo proveniente de vários agregadores são características da arquitetura proposta por PETROVIC *et al.* (2005). Esta é a arquitetura que melhor se encaixa o processo de agregação de RSS proposto, pois a classificação de conteúdo, segundo a abordagem proposta gera uma agregação intermediária.

Este processo de agregação de RSS tem como premissa a filtragem de itens de RSS numa estrutura de dados hierárquica, a taxonomia. Esta classificação parte da análise da descrição de cada item de RSS e seu processamento de modo a capturar termos relevantes a serem utilizados na correspondência com os termos da taxonomia. Ela fornece uma visão do mais geral para o mais específico (ou *top-down*) das categorias em que se encaixam os itens. Cada nó da taxonomia agrega todos os itens dos nós pertencentes aos níveis inferiores e assim sucessivamente até os nós folhas. Esta filtragem segundo a hierarquia de termos da taxonomia difere da filtragem de itens de RSS utilizada nos agregadores tradicionais, que permitem a aplicação de filtros através de palavras-chave.

Para se chegar à agregação de itens numa taxonomia propõe-se o emprego de técnicas consagradas em agregadores, associadas a técnicas de processamento de linguagem natural e recuperação de informação. Nesse contexto, a fim de permitir a categorização dos itens, o processo proposto de agregação constitui-se de seis etapas (conforme a figura 3.1), que fornecem como produto final o conteúdo pronto para ser consumido por outros agregadores de RSS:

1. Associação / subscrição dos canais de interesse do usuário;
2. Captura do RSS;
3. Extração dos termos relevantes dos itens do RSS;
4. Recuperação das categorias dos itens;
5. Busca dos itens com apoio da taxonomia;
6. Agregação dos itens de RSS de acordo com os termos da taxonomia.

Figura 3.1 – Processo de agregação

Inicialmente, identifica-se um conjunto de URLs de canais de RSS, associando-os (subscrevendo) a um sistema. Posteriormente, de tempos em tempos, o canal é verificado e o código XML contendo seu respectivo conteúdo é recuperado (capturado). Na etapa seguinte, os conteúdos dos itens de RSS do código XML capturado são indexados e ocorre a extração de termos relevantes, utilizados juntamente com a categoria (retirado da *tag* <category> do RSS) do item para formar o índice invertido. Dá-se então a busca dos itens com base no cruzamento dos termos recursivamente recuperados da taxonomia (da categoria mais geral para a mais específica) com os termos e categorias extraídos do RSS. Por fim, o resultado da busca gera novos canais de RSS a partir de cada termo da taxonomia. O termo raiz da taxonomia contém todos os itens agregados nos termos imediatamente abaixo.

Os itens previamente recuperados são agregados em diferentes termos da taxonomia, permitindo que o usuário escolha (subscreva) um canal e tenha acesso à informação de acordo com o grau de detalhamento desejado. Conforme descrito na

seção 2.1.3, no contexto deste trabalho também é considerado relevante qualquer item de RSS que contenha como termo ou categoria algum dos termos presentes na taxonomia.

As etapas supracitadas são executadas em cascata (*Pipeline*[23]), simplificando o processamento de informação. Estas etapas fornecem como produto final uma taxonomia de canais contendo itens de RSS organizados (classificados) segundo os termos da taxonomia, onde cada termo é visto como uma fonte de conteúdo (canal), que fornece RSS contendo itens filtrados de canais monitorados. A figura 3.2 apresentada uma visão geral sobre a agregação e a taxonomia na arquitetura proposta por PETROVIC *et al.* (2005), desta vez com a representação da taxonomia e dos itens de RSS.

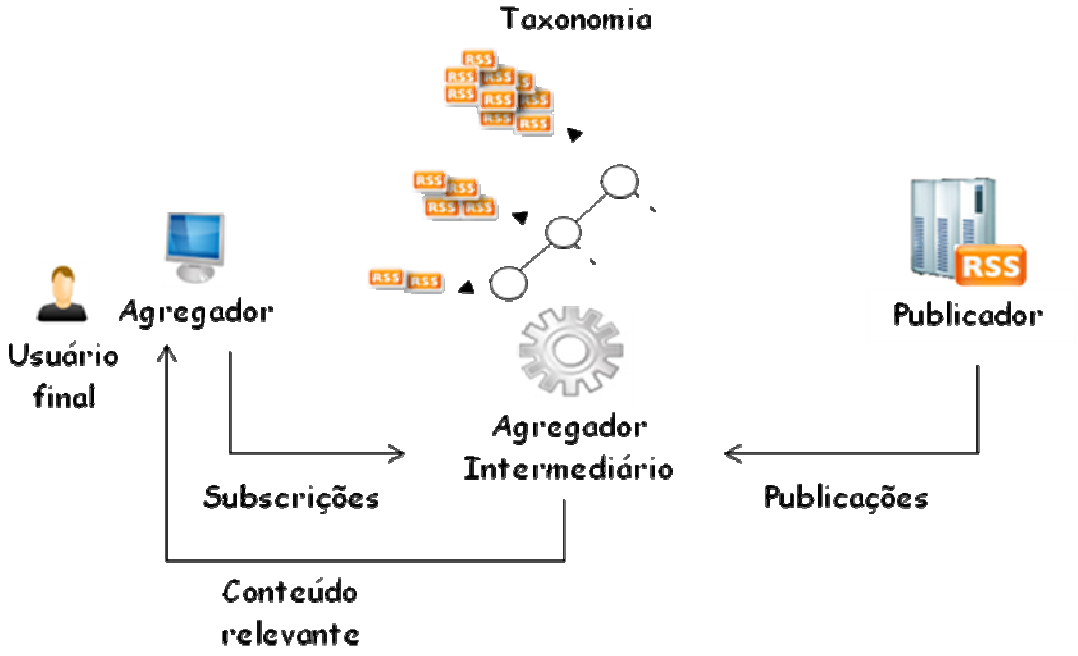

Figura 3.2 – Arquitetura proposta de disseminação de RSS adaptado de PETROVIC *et al.* (2005)

[23] Técnica de processamento de dados onde as instruções são organizadas e executadas em série, seguindo uma ordem.

3.1.1 Etapa de associação / subscrição dos canais de interesse do usuário

Esta etapa é o estágio inicial do processamento em cascata e diz respeito ao ato de associar um ou mais canais de RSS a um agregador. A associação de um RSS a um agregador é o ato de cadastrar no agregador a URL do canal que o usuário deseja acompanhar as publicações através do próprio agregador, de acordo com a definição apresentada no item 2.1.2. No entanto, esta etapa do processo não representa a associação do canal de RSS do publicador do conteúdo com o agregador do usuário final.

A associação de canais de RSS nesta etapa refere-se somente ao cadastramento da URL do canal num agregador intermediário, ou seja, para consumir o RSS contido no RSS monitorado, o usuário necessita executar outra associação. Esta segunda associação refere-se à associação das URLs fornecidas por um conjunto de canais de áreas correlatas organizadas segundo uma hierarquia (taxonomia) composta de um ou mais domínios.

A composição (união de dois conjuntos disjuntos) dessa hierarquia por mais de um domínio não afeta negativamente o funcionamento do processo como um todo, pois isso aumenta a quantidade de assuntos abrangidos pela taxonomia. Esta etapa não se preocupa com a união (ou composição) de duas ou mais taxonomias, pois este não é o foco deste trabalho.

Após o cadastramento dos canais de interesse do usuário no agregador intermediário, que monitorará constantemente qualquer nova publicação, o processo executa a captura do RSS destes canais.

3.1.2 Etapa de captura do RSS

Nesta etapa, ocorre o monitoramento contínuo dos publicadores de RSS. As URLs que indicam os caminhos dos RSS cadastrados pelo usuário final na etapa anterior (subscrição) são verificadas de tempos em tempos e quando a verificação observa alteração no documento, o agregador guarda (persiste) a informação contida no RSS para ser processada na etapa seguinte, a extração de termos relevantes.

A capacidade de guardar a informação contida no RSS para futura redistribuição pode acarretar em disputa de direitos autorais, de acordo com o que foi discutido na seção 1.1. Apesar disso, esta é a opção escolhida por agregadores que utilizam as arquiteturas das figuras 2.2 e 2.3. A persistência do RSS capturado facilita o processamento da informação (como a filtragem e agregação). Um exemplo de persistência de RSS é a realizada pelo agregador Google Reader[24] (também citado na seção 2.1.2.2), que realiza o armazenamento do RSS e permite a consulta a informações capturadas durante um longo período.

3.1.3 Etapa de extração dos termos relevantes dos itens do RSS

Nesta etapa, os itens de RSS recuperados na etapa anterior passam pelo processo de extração de termos relevantes para a criação de um índice composto por vetores de termos relevantes. Esta etapa pode ser implementada de diferentes formas, por exemplo, com base num conjunto de dicionários de termos relevantes previamente selecionados ou através da mineração de textos.

A mineração de textos permite a recuperação de termos relevantes sem o conhecimento do domínio do texto. No entanto, para esta etapa de extração considera-se mais adequada a utilização de dicionários, por conterem apenas termos relevantes, por

[24] www.google.com.br/reader/

permitir a recuperação de termos compostos por mais de uma palavra e pelo fato de poderem abranger os termos presentes na taxonomia.

A etapa de extração de termos relevantes forma um vetor de termos para cada item de RSS agregado. Por exemplo, dado o item (recuperado a partir da biblioteca digital do IEEE) com o título "*Data Mining for Software Engineering*" e a descrição "*To improve software productivity and quality, software engineers are increasingly applying data mining algorithms to various software engineering tasks. However, mining SE data poses several challenges. The authors present various algorithms to effectively mine sequences, graphs, and text from such data*.", obtém-se um vetor formado pelos termos:

- data mining algorithms;
- software productivity;
- engineering data;
- quality software;
- software engineers;
- software engineering;
- algorithm;
- sequences;
- graphs;
- databases;
- challenges.

A este vetor são adicionados novos termos na etapa de recuperação de categoria, a etapa seguinte neste processo de agregação de RSS.

3.1.4 Etapa de extração das categorias dos itens

Esta etapa busca enriquecer os vetores de termos que representam os itens de RSS recuperados na etapa de captura do RSS. Ao vetor de cada item são adicionados os termos encontrados na classificação definida pelo publicador do RSS cadastrado pelo usuário final na etapa de subscrição dos canais (abordagem similar à adotada por LEITE (2009)). Estes termos estão presentes no elemento <category> de cada item de RSS, mas, por não ser um item obrigatório na estrutura do RSS (conforme a estrutura apresentada na seção 2.1.1), este elemento muitas vezes não é fornecido pelo publicador do RSS. Quando não há categorias fornecidas pelo publicador do RSS, os termos relevantes extraídos na etapa anterior são considerados como as categorias do item. Por exemplo, ao vetor de termos apresentados na etapa anterior, são adicionados os termos:

- Data mining;
- Software engineering;
- Design and test;
- Computational intelligence.

3.1.5 Etapa de busca dos itens com apoio da taxonomia

Nesta etapa, os itens de RSS são buscados na taxonomia. A classificação utiliza o vetor de termos e categorias preenchidos pelas etapas de extração dos termos relevantes e de extração das categorias dos itens, concluindo parte da indexação do processo em questão. Considera-se esta indexação como uma preparação para a construção do índice invertido, pois os vetores que representam os itens são considerados como documentos no processo de criação do índice invertido, descrito de acordo com a seção 2.2.1.

A busca é realizada sobre os vetores dos itens no momento que o usuário deseja acessar um dos canais de RSS associados a termos da taxonomia, aplicando-se como termo de busca o conjunto de termos que formam a hierarquia a partir do termo selecionado pelo usuário. Ocorre então cruzamento dos termos de busca com os vetores aplicando algum modelo de RI.

A construção desta etapa pode ser realizadada de diferentes formas, adotando-se algum dos modelos de recuperação de informação citados na seção 2.2. De acordo com o modelos da seção citada, o modelo vetorial é o escolhido neste trabalho para a busca dos itens de RSS (utilizada na construção do protótipo iRSS), devido à sua eficácia publicamente comprovada e rapidez na recuperação de documentos, segundo BAEZA-YATES *et al.* (1999) .

3.1.6 Etapa de agregação dos itens de RSS de acordo com os termos da taxonomia

Nesta etapa, os itens de RSS previamente recuperados, indexados e buscados formam canais de RSS, que são associados aos termos da taxonomia. A agregação ocorre associada à busca dos itens com apoio da taxonomia. Os itens são agregados de acordo com a similaridade (calculada pelo modelo de RI utilizado) entre os termos da taxonomia e os vetores dos itens. Os termos menos relevantes na hierarquia (taxonomia) tendem a agregar um número menor de itens de RSS, ou seja, possuem menor grau de refinamento, conforme a figura 3.4 e os resultados obtidos no capítulo 4.

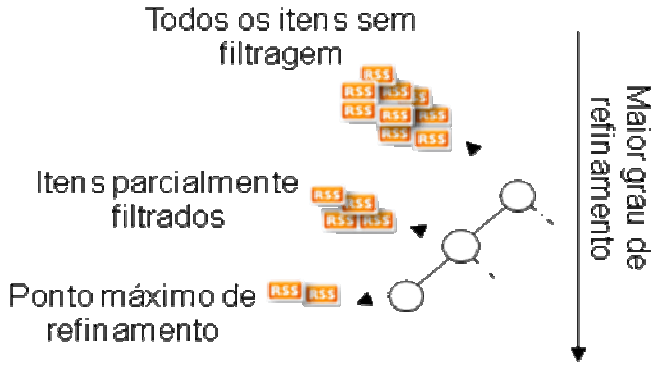

Figura 3.3 – Refinamento dos itens na taxonomia

3.2 Considerações sobre as técnicas aplicadas na agregação com o apoio de uma taxonomia

O processo de agregação descrito neste capítulo possui características especiais quanto à forma de recuperação, acesso e tratamento de itens de RSS com categoria indefinida pelo publicador e quanto à extração de termos relevantes. O acesso ao conteúdo agregado de forma hierárquica facilita o consumo de RSS por usuários comumente assolados por enxurradas de informação. Assim, quanto maior for a especialização da taxonomia, o número de itens de RSS tende a ser mais bem distribuído, observando-se termos mais relevantes da hierarquia associados a canais de RSS com um número elevado de itens e termos mais específicos associados a canais com um menor número de itens.

3.3 Considerações

Por se tratar de uma forma de agregação intermediária, que pode inclusive servir para fornecer conteúdo para outros agregadores do mesmo tipo, existem discussões relativas aos direitos autorais do conteúdo agregado. Porém, o agregador em questão não possui fins lucrativos, o conteúdo persistido consiste apenas de um sumário, um

título e um *link* para o conteúdo original, e trata-se apenas de material para validação de uma pesquisa. A proposta apresentada neste capítulo gerou um protótipo, o iRSS, descrito e validado no capítulo 4.

4. Construção e validação do processo de agregação

Este capítulo discorre sobre a construção de um protótipo que automatiza o processo de agregação proposto no capítulo 3, tratando da análise do problema e do projeto de desenvolvimento. Além disso, este capítulo descreve a avaliação do processo de agregação proposto no capítulo 3, através da aplicação em um estudo de caso.

A seção 4.1 apresenta o contexto onde se insere o protótipo construído, o iRSS. A seção 4.2 descreve brevemente o agregador, uma vez que a especificação detalhada refere-se ao processo de agregação proposto neste trabalho, descrito no capítulo 3. A seção 4.3 aborda e justifica as escolhas consideradas gerais, como a escolha do banco de dados relacional, a forma de extração dos termos relevantes contidos no RSS e a escolha do modelo de recuperação de informação utilizado. Já as escolhas específicas são descritas na seção 4.4, que descreve e justifica a infra-estrutura empregada, como a escolha do SGBD e da linguagem de programação, entre outras escolhas de projeto. A seção 4.5 descreve os componentes do protótipo. Na seção 4.6 é apresentado o cenário de avaliação, na seção 4.7 é descrita a comparação quanto ao emprego de taxonomia na metodologia de agregação em relação ao agregador RSSOWL (agregador que permite a criação de taxonomias), na seção 4.8 são realizadas comparações com outras ferramentas de RSS, por fim na seção 4.9 são apresentadas conclusões sobre os resultados obtidos.

4.1 Contexto do problema

De acordo com a contextualização apresentada na seção 1.1, RSS se mostra eficaz na distribuição de conteúdo. Sua larga utilização por agências de comunicação foi um dos motivos para o sucesso. Por outro lado, sua ampla utilização como meio de distribuição de conteúdo trouxe à tona anomalias já observadas pela área de RI (apresentada no capítulo 2). Estas anomalias são o excesso de informação e a baixa relevância do conteúdo recuperado por agregadores de RSS.

Para atacar os problemas citados, este trabalho propôs um processo que propicie a filtragem do conteúdo entregue aos agregadores de RSS (apresentado no capítulo 3). Este processo é composto por etapas que têm como objetivo recuperar o RSS, extrair informação, e fornecer conteúdo re-utilizável e filtrado de acordo com os termos de uma taxonomia.

4.2 Breve especificação do iRSS

O agregador deve seguir as etapas descritas da seção 3.1.1 à seção 3.1.6. Portanto, deve permitir a associação, captura, extração dos termos relevantes dos itens do RSS, recuperação das categorias dos itens, busca dos itens com apoio da taxonomia e agregação dos itens de RSS de acordo com os termos da taxonomia. Estas etapas devem seguir a ordem especificada pelo processo citado. Trata-se de um processo em cascata, largamente utilizado por SRIs, como o ESP[25].

[25] Sistema de recuperação de informação da Microsoft.

4.3 Escolhas para implementação do iRSS

Algumas decisões de projeto para a construção do iRSS diferenciam-no de outros SRIs. A seguir são apresentadas decisões de cunho geral. As decisões sobre a infra-estrutura são abordadas na seção seguinte.

A primeira decisão é quanto à forma de persistência dos documentos XML. Para esta tarefa utiliza-se de banco de dados relacional. A justificativa para esta escolha se refere à utilização da abordagem orientada a banco de dados para persistência dos itens de RSS recuperados. De acordo com o item 2.2.4.1, a abordagem orientada a banco de dados apresenta as seguintes vantagens: (i) as relações dos dados e suas restrições de integridade podem ser modeladas e checadas; (ii) as operações padrão (de definição e manipulação) de banco de dados podem ser usadas; (iii) a linguagem de consulta em geral é similar à linguagem padrão de banco de dados (SQL).

A segunda decisão é quanto à extração dos termos relevantes dos itens do RSS. Para facilitar a construção do agregador e devido a sua apurada eficácia, o iRSS faz uso de um serviço Web. Trata-se de um serviço de extração de termos relevantes, diferentemente da utilização de um dicionário para a extração dos termos relevantes.

A terceira escolha se refere à busca do conteúdo agregado. Devido a sua simplicidade, facilidade de integração e eficiência, o modelo vetorial e suas variantes são muito utilizados. Por isso, suas características servem de base para a escolha do modelo vetorial para a construção do iRSS.

4.4 Infra-estrutura utilizada

As ferramentas utilizadas no desenvolvimento do iRSS são as seguintes:

- Java SDK – Plataforma de desenvolvimento do iRSS. Esta plataforma foi escolhida por possuir a biblioteca ROME[26] RSS, utilizada para recuperação do RSS e geração dos canais de RSS da taxonomia.
- JDBC PostgreSQL – Biblioteca de acesso a banco de dados PostgreSQL, que permite a comunicação do sistema em Java com o banco de dados PostgreSQL 8.4.
- PostgreSQL 8.4 – Sistema de gerência de banco de dados relacional onde o documento XML (o RSS) é persistido para ser processado, ou seja, para possibilitar a classificação dos itens de RSS nos nós da taxonomia. A forma como o documento XML é persistido não influencia no resultado final, mas o PostgreSQL 8.4 foi utilizado devido ao seu suporte a FTS e por permitir consultas recursivas. Esta possibilidade de execução de consulta é útil para recuperar todos os termos subordinados a um determinado termo da taxonomia, transferindo essa responsabilidade para o banco de dados.
- PgAdmin 3 – Ferramenta de administração de banco de dados utilizada para definição do esquema do banco de dados do protótipo.
- Netbeans 6.1 – Ambiente de desenvolvimento utilizado na construção do código na linguagem Java.
- Apache Tomcat – Servidor Web com suporte a JSP utilizado na hospedagem da aplicação (protótipo).
- ArgoUML – Ferramenta de modelagem para UML utilizada para modelagem dos diagramas em UML apresentados nesta dissertação.

[26] https://rome.dev.java.net/

- Dbdesigner 4 – Ferramenta de modelagem de banco de dados utilizada para gerar o esquema do modelo físico do banco de dados.
- Extrator de termos do Yahoo[27] - Para facilitar a construção do agregador e devido a sua apurada eficácia, o iRSS faz uso de um serviço Web publicado pelo Yahoo. Trata-se de um serviço de extração de termos que o Yahoo disponibiliza aos desenvolvedores sob o protocolo REST, um protocolo baseado em XML para troca de mensagens assíncronas na Internet.

4.4.1 Modelo físico do banco de dados

Dado que a forma de persistência de documentos XML escolhida é a abordagem orientada a banco de dados, a figura 4.1 apresenta a especificação do modelo físico do banco de dados do iRSS. Neste modelo, estão representadas as tabelas: canal, taxonomia, nó (termo) da taxonomia, item, categoria e termo relevante.

A taxonomia é formada por um ou mais "nós", que por sua vez podem ser "pais" de zero ou mais "nós". Cada nó associa-se a um e apenas um canal, que contém um ou mais itens de RSS e podem possuir uma ou mais categorias e termos relevantes. Este modelo permite a persistência dos itens recuperados na etapa de captura, a persistência dos termos relevantes e categorias extraídos durante o processo, e principalmente permite a persistência e facilita a recuperação dos termos da taxonomia.

[27] http://developer.yahoo.com/search/content/V1/termExtraction.html

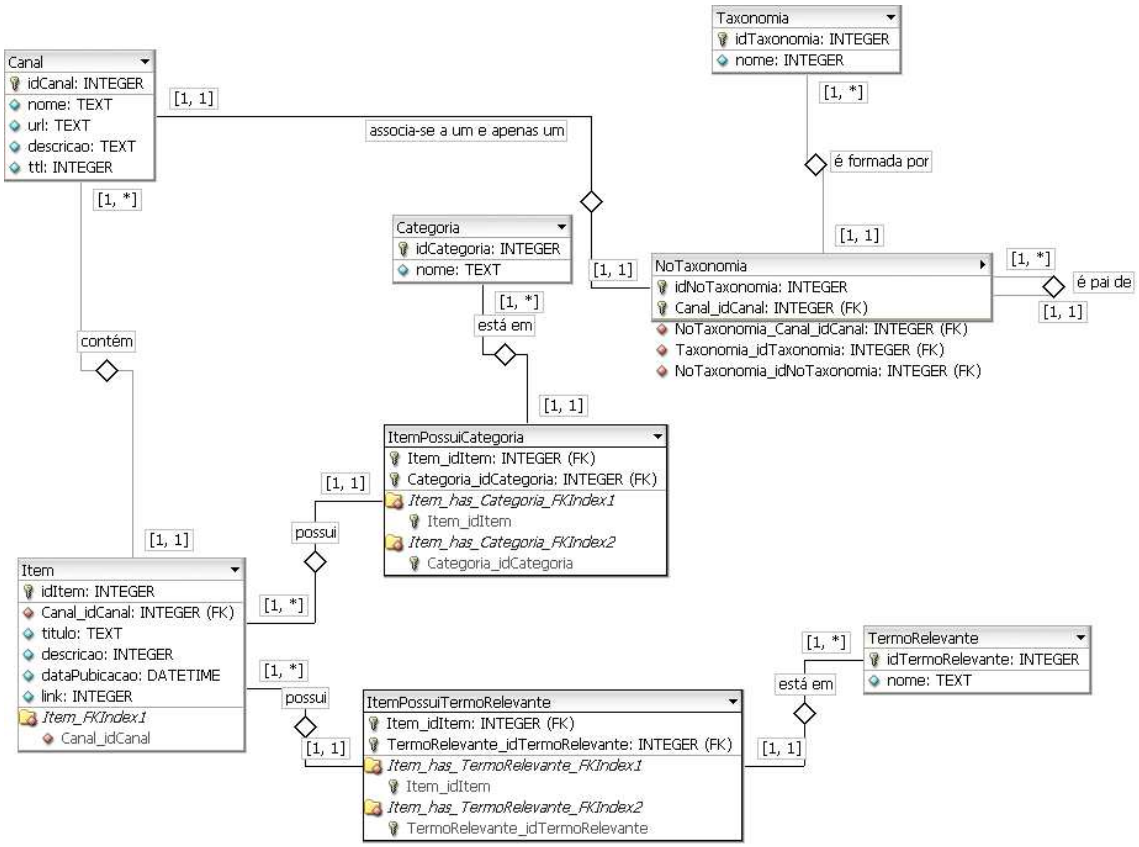

Figura 4.1 – Modelo físico do banco de dados do iRSS

4.4.2 Casos de uso

Esta seção descreve os principais casos de uso do iRSS. Estes casos de uso retratam o consumo do RSS, a indexação e a agregação do RSS persistido no agregador.

Nome do Caso de Uso: Consumir RSS	**UC #:** 1
Ator: Usuário	
Descrição Permite que os usuários naveguem na taxonomia e escolham os nós da taxonomia, permitindo o acesso ao conteúdo do RSS agregado na sub-árvore do termo escolhido. **Fluxo Principal** Este caso de uso começa quando o usuário escolhe a taxonomia desejada. 1. O sistema exibe a taxonomia de referência; 2. O usuário escolhe o tópico de seu interesse na taxonomia; 3. O sistema recupera os itens de RSS indexados que se encaixam nas categorias dos	

termos da sub-árvore do termo selecionado e exibe um documento RSS com os itens agregados.

Fluxos Alternativos

Não há.

Pontos de extensão

Não há.

Casos de uso incluídos

Não há.

Pós-condições

Não há.

Conforme apresentado no Caso de Uso 1: Consumir RSS, o sistema exibe a taxonomia de referência, tal como ilustrado na Figura 4.2, onde o iRSS exibe o nível da taxonomia abaixo do tópico Software/Software Engineering.

Figura 4.2 – Visão da taxonomia

Uma vez selecionado um determinado nó da taxonomia, por exemplo: *Programming Techniques*, o sistema recupera os itens de RSS que se referem ao nó selecionado, exibindo um documento RSS com estes itens agregados, tal como no exemplo apresentado na Figura 4.3.

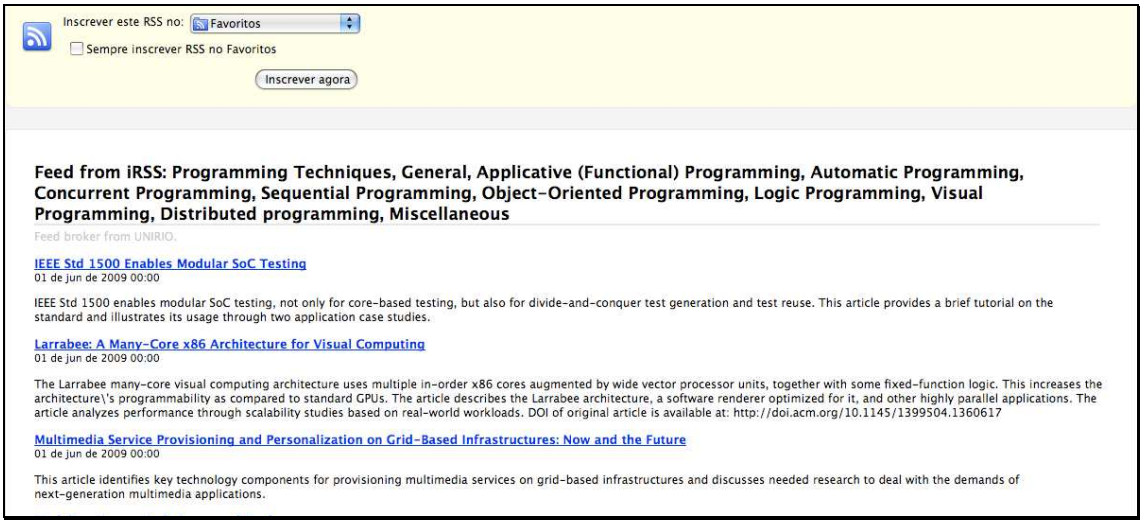

Figura 4.3 – Visão de um canal de RSS a partir do termo "Programming techniques"

Nome do Caso de Uso: Indexar RSS	**UC #:** 2
Ator: Extrator de termos	

Descrição

Este caso de uso realiza a indexação dos itens de RSS, recuperando a categoria do item e seus termos relevantes.

Fluxo Principal

Este caso de uso começa quando há algum item de RSS agregado a partir de algum canal de RSS.

1. O sistema recupera o RSS, extrai as categorias dos itens e envia o conteúdo dos itens agregados ao Extrator de termos;
2. O extrator de termos extrai os termos relevantes e os envia ao sistema;
3. O sistema associa os termos relevantes aos itens correspondentes;

Fluxos Alternativos

Não há.

Pontos de extensão

Não há.

Casos de uso incluídos

Não há.

Pós-condições

Não há.

Nome do Caso de Uso: Agregar RSS	**UC #:** 3
Ator: Canal de RSS	

Descrição

Este caso de uso realiza a agregação dos itens de RSS, recuperando-os a partir dos canais de RSS previamente cadastrados no sistema.

Fluxo Principal

Este caso de uso começa quando o usuário cadastra algum canal de RSS.

1. O usuário insere a URL o canal no sistema;
2. O sistema recupera o conteúdo dos canais de RSS cadastrados e executa o caso de uso “Indexar RSS”;

Fluxos Alternativos

Não há.

Pontos de extensão

Não há.

Casos de uso incluídos

Indexar RSS.

Pós-condições

Não há.

4.4.3 Diagrama de Atividades do processo de agregação do iRSS

O iRSS segue as etapas do processo de agregação proposto no capítulo 3. Por isso, seu diagrama de atividades inicia-se com a etapa de subscrição dos canais, seguido

pela captura do RSS, extração de termos relevantes, recuperação das categorias, busca dos itens com apoio da taxonomia e agregação do RSS.

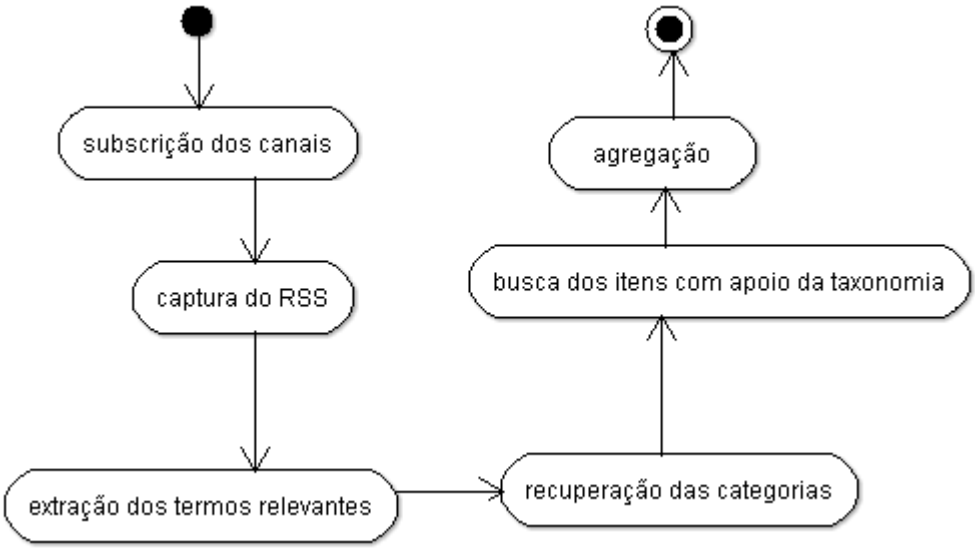

Figura 4.5 – Diagrama de Atividades

4.5 Os componentes do iRSS

Os componentes do protótipo em questão realizam a agregação com base nas seis etapas citadas no item 3.1, acrescido de um componente de visualização do conteúdo agregado. Nesse contexto, os componentes e suas associações estão representados na Figura 4.6.

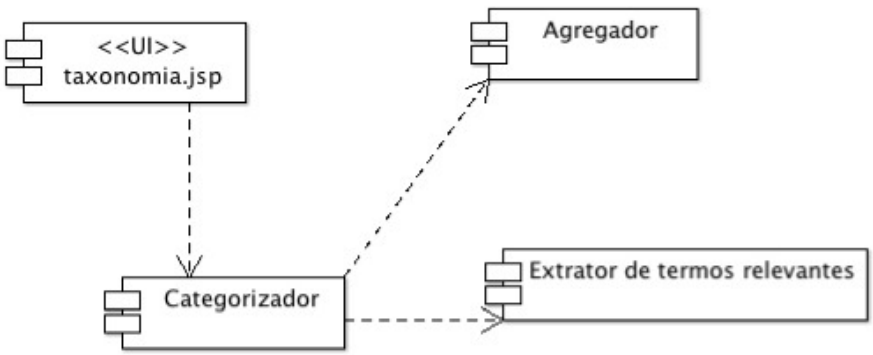

Figura 4.6 – Diagrama de componentes

O componente de visualização “taxonomia.jsp” exibe uma taxonomia e permite a navegação entre seus nós, onde cada nó fornece um canal de RSS que contém todos os itens do próprio nó e dos nós de sua sub-árvore. A agregação é realizada em tempo de execução através do componente de agregação, ou seja, no momento que um usuário escolher o RSS de um nó, o componente de agregação retorna ao componente de visualização o resultado da execução de uma consulta a base de dados todos os itens de RSS contendo as categorias da sub-árvore do nó selecionado; utilizando FTS (*Full Text Search*) no momento da consulta.

A categorização é realizada por um componente responsável pela recuperação das categorias de cada item de RSS. Este componente inicialmente persiste as categorias recuperadas através do emprego de um sistema externo de extração de termos relevantes, o sistema do Yahoo, delegando grande parte do processamento de linguagem natural a tal sistema e aumentando a escalabilidade do sistema. Em seguida o componente recupera a categoria definida no item ou página onde está publicada a notícia correspondente ao item.

4.6 O cenário utilizado na avaliação

A avaliação de um agregador de RSS carece de conteúdo proveniente de diferentes canais de RSS. Assim, este cenário de avaliação leva em consideração canais de RSS do meio acadêmico que divulgam artigos científicos, em especial da área de computação, com o objetivo de promover a divulgação da pesquisa científica.

Foram utilizados 24 canais de RSS com conteúdo proveniente da biblioteca digital do IEEE[28]. Estes canais forneceram um total de 544 itens de RSS durante o período de 20 a 24 de julho de 2009, a partir dos quais foram extraídos 1311 termos

[28] Estes canais podem ser encontrados em http://ieeexplore.ieee.org/guide/g_tools_rss.jsp

relevantes. Esses itens são notícias (artigos e comunicados) que foram classificadas na taxonomia do próprio IEEE, uma taxonomia com 2118 tópicos (ver anexo A).

4.7 Avaliação quanto ao emprego da taxonomia

No iRSS, a taxonomia visa organizar os assuntos de uma área de domínio de modo a apoiar a filtragem dos itens de RSS. Para chegar a tal filtragem, as etapas de recuperação de termos relevantes dos itens e de recuperação das categorias de RSS preenchem o vetor de termos que representam cada item (como está descrito nas seções 3.1.3 e 3.1.4). Quando esses mesmos itens são disponibilizados em agregadores comuns (descritos na seção 2.1.2), as categorias e termos relevantes são normalmente ignorados e não servem para filtragem.

A classificação hierárquica do conteúdo no iRSS nos testes realizados forneceu um grau de detalhamento não encontrado em outros agregadores, dado que o conteúdo agregado está distribuído por toda a taxonomia, conforme mostra a figura 4.7, que mostra a distribuição dos itens (notícias) na taxonomia de acordo com o identificador do nó na taxonomia, partindo do nó raiz (identificador igual a zero) e executando-se uma leitura em profundidade. Dado que um mesmo item de RSS pode ser associado a mais de um tópico, a partir deste gráfico observa-se que apesar de um determinado agregador associado a um determinado nó da taxonomia disponibilizar o conteúdo dos nós da sub-árvore a ele associado, os itens não estão diretamente associados a aquele tópico especificamente. Assim, observa-se que os tópicos mais gerais da taxonomia encontram-se na faixa acima de 50 notícias agregadas, enquanto os tópicos mais próximos às extremidades da taxonomia (folhas inclusive), possuem menos de 50 notícias agregadas.

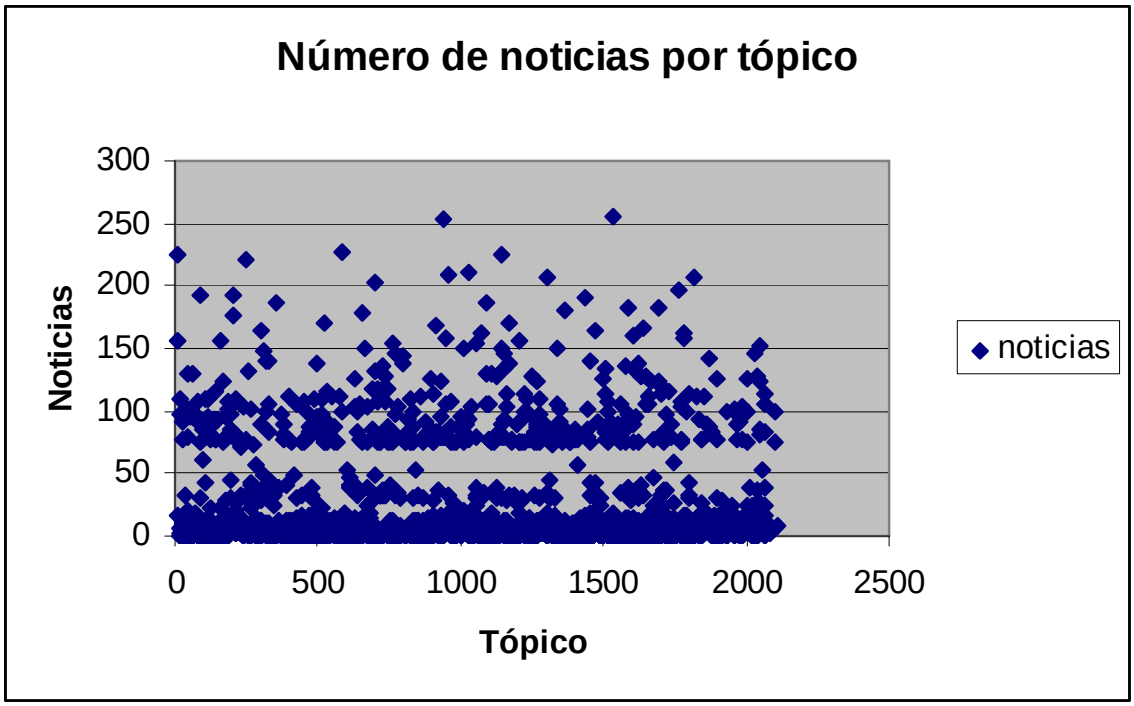

Figura 4.7 – Distribuição de notícias na taxonomia

Os filtros aplicados ao conteúdo agregado podem ser de duas formas: para recuperar documentos preferidos e para descartar documentos indesejados. O trabalho apresentado por PINHEIRO *et al.* (2009), por exemplo, busca descartar itens de RSS para agregar conteúdo relevante. Seu trabalho aplica regras lógicas fornecidas pelos usuários de forma colaborativa para filtrar itens de RSS indesejados, diferentemente da abordagem adotada neste trabalho, que busca filtrar o RSS de acordo com os termos previamente selecionados.

O agregador usado como referência para comparações foi o RSSOWL. O RSSOWL foi escolhido para comparação com o iRSS por ser um agregador de RSS que tem como principal característica a possibilidade de criação de hierarquias e aplicação de consultas. O emprego da taxonomia para recuperação de itens de RSS (conforme a abordagem proposta) propicia uma visão hierarquizada do conteúdo agregado, enquanto no RSSOWL as consultas não servem de entrada para outras consultas, forçando o

usuário a aplicar sucessivas consultas para recuperar conteúdo hierarquicamente relacionado, o que prejudica a usabilidade.

Segundo VILLORIA *et al.* (2006), o RSSOWL possui características que o tornam diferente de qualquer agregador até então desenvolvido. Para ele, os agregadores de RSS que utilizam taxonomias ou ontologias devem ter três funcionalidades básicas: visão semântica, navegação semântica e consulta semântica.

Visão semântica é a visualização de itens agregados de acordo com um esquema de metadados (dados que descrevem outros dados) definido pelo usuário. Isso permite, por exemplo, que títulos de itens sejam destacados de acordo com filtros; como um filtro de destaque de títulos de itens que tenham ou não conceitos associados aos itens exibidos pelo agregador. Outro exemplo de visão semântica é a anotação de conteúdo; nesse caso, os termos presentes no texto do item são destacados de acordo com os conceitos dos nós da taxonomia (ou ontologia). Ainda segundo VILLORIA *et al.* (2006), navegação semântica é a navegação de um publicador para outros publicadores semanticamente relacionados; o que é possibilitado através da navegação para publicadores que possuem conceitos vizinhos (na taxonomia) aos conceitos associados ao publicador selecionado. Por fim, (VILLORIA *et al.*, 2006) define consulta semântica como sendo a busca de itens através de conceitos de uma taxonomia, permitindo ao usuário uma abstração maior que a busca por palavra chave.

Tais características tornam o RSSOWL uma boa opção para a comparação dos resultados obtidos a partir da agregação, mas principalmente pelo emprego de FTS para agregar conteúdo. Por outro lado, o RSSOWL utilizar a taxonomia apenas como uma organização lógica para consultas e canais, ou seja, os termos da taxonomia não são aplicados como termos de busca. Também se observou que o RSSOWL utiliza a

ferramenta de busca Lucene[29], enquanto o iRSS utiliza uma implementação do modelo vetorial, influenciando diretamente o resultado da busca dos termos contidos na taxonomia. Como a taxonomia utilizada contém mais de dois mil tópicos, e o RSSOWL não permite o carregamento automático de uma taxonomia, foram selecionados aleatoriamente alguns tópicos da taxonomia e aplicadas as mesmas consultas sobre o mesmo conjunto de notícias na forma de RSS considerando-se apenas as categorias assinaladas por seus publicadores.

A seguir, é apresentada a lista dos termos e os resultados obtidos na agregação:

1. "Software science"
2. "Automatic Programming, Automatic analysis of algorithms, Program modification, Program synthesis, Program transformation, Program verification"
3. "Problem Solving, Control Methods, and Search, Backtracking, Constraint satisfaction, Control theory, Dynamic programming, Graph and tree search strategies, Heuristic methods, Plan execution, formation, and generation, Scheduling"
4. "Pattern Recognition, General, Models, Design Methodology, Clustering, Applications, Implementation, Miscellaneous, Deterministic, Fuzzy set, Geometric, Neural nets, Statistical, Structural, Syntactic, Classifier design and evaluation, Feature evaluation and selection, Pattern analysis, Algorithms, Similarity measures, Arts, Computer vision, Computational models of vision, Face and gesture recognition, Government, Handwriting analysis, Industry, Medicine, Military, Remote sensing, Robotics, Sciences, Signal processing, Text processing, Waveform analysis, Interactive systems, Real-time systems, Special architectures"

[29] http://lucene.apache.org/java/docs/

5. “Simulation, Modeling, and Visualization, General, Simulation Theory, Simulation Languages, Applications, Model Validation and Analysis, Model Development, Simulation Output Analysis, Simulation Support Systems, Types of Simulation, Visualization, Miscellaneous, Model classification, Systems theory, Types of simulation, Modeling methodologies, Environments, Animation, Combined, Continuous, Discrete event, Distributed, Gaming, Monte Carlo, Parallel, Visual, Applications, Flow visualization, Information visualization, Multivariate visualization, Visual programming and program visualization, Visualization systems and software, Visualization techniques and methodologies, Volume visualization”
6. “Control Structures and Microprogramming, General, Control Design Styles, Control Structure Performance Analysis and Design Aids, Control Structure Reliability, Testing, and Fault-Tolerance, Microprogram Design Aids, Microcode Applications, Miscellaneous, Hardwired control, Microprogrammed logic arrays, Writable control store, Automatic synthesis, Formal models, Simulation, Diagnostics, Error-checking, Redundant design, Test generation, Firmware engineering, Languages and compilers, Machine-independent microcode generation, Optimization, Verification, Direct data manipulation, Firmware support of operating systems/instruction sets, Instruction set interpretation, Peripheral control, Special-purpose, Emerging technologies”
7. “I/O and Data Communications, General, Data Communications Devices, Input/Output Devices, Interconnections (Subsystems), Performance Analysis and Design Aids, Reliability, Testing, and Fault-Tolerance, Miscellaneous, Processors, Receivers, Transmitters, Channels and controllers, Data terminals and printers, Image display, Voice, Asynchronous/synchronous operation, Fiber

optics, Interfaces, Parallel I/O, Physical structures, Topology, Web technologies, Wireless systems, Formal models, Simulation, Verification, Worst-case analysis, Built-in tests, Diagnostics, Error-checking, Hardware reliability, Redundant design, Test generation"

8. "Logic Design, General, Design Styles, Reliability and Testing, Design Aids, Miscellaneous, Cellular arrays and automata, Combinational logic, Logic arrays, Memory control and access, Memory used as logic, Parallel circuits, Sequential circuits, Built-in tests, Error-checking, Redundant design, Test generation, Testability, Automatic synthesis, Hardware description languages, Optimization, Simulation, Switching theory, Verification"
9. "Low-power design"
10. "Human Haptics, Touch-Based Properties, Attention, Biomechanics, Cognition, Human Factors and Ergonomics, Human Performance, Neuroscience, Perception and Psychophysics, Social Communication"
11. "Formal Languages, Algebraic language theory, Classes defined by grammars or automata, Classes defined by resource-bounded automata, Decision problems, Operations on languages"
12. "Mathematical Logic, Computability theory, Computational logic, Lambda calculus and related systems, Logic and constraint programming, Mechanical theorem proving, Modal logic, Model theory, Proof theory, Recursive function theory, Set theory, Temporal logic"
13. "Numerical Algorithms and Problems, Computation of transforms, Computations in finite fields, Computations on matrices, Computations on polynomials, Number-theoretic computations"
14. "Normal models of communication"

15. “Error control codes”
16. “Hash-table representations”
17. “Web Services, General, Composite Services, Web Services Publishing, Web Services Discovery, Web Services Modeling, Web Services Communication Protocols, Web Services Binding, Web Services Publishing, Stateful Web Services, Web Services Interoperability, Composite Web Services, Representation of Composite Services, Three-Dimensional Modeling, Public Services Registry, Private Services Registry, Distributed Services Registry, Search Discovery Language, Services Discovery Engine, Services Discovery Process and Methodology, Services Discovery Architecture, Federated Services Discovery”
18. “Business Grid, General, Logical Grid Infrastructure, Business Grid Solution Development, Service-Oriented Grid Computing, Business Grid Solution Framework, Packaged Application Grid, Business Grid Middleware, Business Process Grid, Business Grid Service Development, Business Grid Service Invocation”
19. “Solution-Level Quality of Service, Context-Aware QoS Model, Representation of QoS Model, QoS Data Management, Business Relationship Model, Solution-Level QoS Framework”
20. “Messaging”

Tabela 4.1 – Quantidade de notícias agregadas

Termos	iRSS	RSSOWL
1	17	48
2	32	212
3	98	304
4	192	479
5	167	459
6	156	478
7	190	474
8	112	377
9	8	142
10	107	356
11	8	158
12	142	329
13	117	225
14	28	100
15	3	73
16	1	22
17	136	320
18	55	268
19	39	342
20	2	0

Observou-se que o iRSS agregou apenas conteúdo relevante, enquanto o RSSOWL agregou muitos itens não considerados relevantes. Tal resultado mostra que a técnica de RI aplicada no iRSS propiciou melhores resultados em relação a técnica de RI empregada no RSSOWL, onde apenas termos únicos são considerados na consulta, diferentemente do modelo vetorial utilizado que considera termos compostos por mais de um termo e ainda conta com a extração de termos relevantes no índice.

Dada a dificuldade de reprodução de todas as consultas proporcionadas pelo iRSS no RSSOWL devido à barreira de usabilidade anteriormente citada, a tabela 4.2 mostra outro comparativo com relação ao RSSOWL, onde a média consiste na média de notícias agregadas por canal, a mediana indica a medida de tendência central do número de notícias, a moda indica o valor do número de notícias agregadas que mais se repete, o desvio padrão indica a o número de notícias mais comum na dispersão estatística e o coeficiente de variação serve para comparação com outras distribuições. A média obtida

pelo iRSS mostrou-se próxima à média do RSSOWL, mostrando que mesmo numa taxonomia com muitos tópicos, a média de notícias agregadas não ficou baixa e é composta apenas por notícias relevantes (levando-se em conta o resultado do experimento anterior). A mediana obtida pelo iRSS foi baixa pois no RSSOWL não é considerada uma taxonomia, apenas canais separados, enquanto o iRSS utiliza uma taxonomia com muitos tópicos, característica que também influencia a moda estatística. Por fim, o desvio padrão considerou-se elevado no iRSS devido a característica da forma de utilização da taxonomia, onde os tópicos mais gerais agregam o conteúdo dos tópicos de toda a sub-árvore que está abaixo.

Tabela 4.2 - Medidas de posição

	iRSS	RSSOWL
Média	26,88	22,66666667
Mediana	3	15,5
Moda	1	50
Desvio padrão	42,17456771	18,86949843
Coeficiente de variação	1,568989925	0,832477872

4.8 Comparações com outros agregadores de RSS

A utilização de taxonomia para classificação de itens de RSS não é comum. Foram realizadas buscas na Web em sites de pesquisa e a artigos, teses e dissertações que abordam o RSS como tema de pesquisa e apenas o agregador BibSonomy (baseado no CMS-TOPSS) realiza o emprego de algo similar a taxonomia para classificar itens de RSS. Trata-se de um sistema de publicação compartilhada de publicações, podendo ser visto como um agregador de RSS específico para o compartilhamento de publicações organizadas numa folksonomia[30]. Apesar de ter um enfoque específico baseado em

[30] Forma de indexação onde os usuários contribuem com termos para indexar o conteúdo.

bibliografias, o BibSonomy permite que os usuários naveguem na folksonomia e recuperem dinamicamente o conteúdo classificado.

A tabela 4.3 mostra uma lista de agregadores que utilizam FTS, comparando-os ao iRSS em termos de estrutura de apoio à agregação de RSS e extração de informação do conteúdo agregado. Embora outros agregadores de RSS possam ser encontrados, não foram listados por não considerarem algo fundamental para o contexto da abordagem proposta: a filtragem pelo texto (FTS).

Tabela 4.3 – Comparação entre Agregadores Relacionados ao Trabalho

Agregador	Utiliza hierarquias?	Extrai automaticamente informação de itens de RSS ?
BibSonomy	Não (folksonomia)	Não
RSSOWL	Sim (separar canais)	Não
RSS Bandit	Sim (separar canais)	Não
GreatNews	Sim (separar canais)	Não
iRSS	Sim (busca)	Sim

4.9 Conclusões sobre os resultados obtidos

Na amostra de 20 consultas aplicadas no iRSS e RSSOWL, em geral o iRSS agregou menos notícias que o RSSOWL, porém com maior grau de relevância que o RSSOWL, uma vez que o RSS utilizado na amostra contém a categoria corretamente assimilada pelo publicador (neste caso é o IEEE). O maior grau de relevância se deve ao fato de a busca utilizando FTS no RSSOWL não ser aplicável ao metadado que indica a categoria do RSS, enquanto o iRSS aplica FTS ao campo de categoria.

A média de notícias agregadas no iRSS ficou ligeiramente maior que a média do RSSOWL, utilizado como um agregador tradicional, sem criação de uma taxonomia. Tal fato indica que mesmo numa taxonomia com mais de dois mil canais, onde o tópico mais geral contém o conteúdo dos tópicos filhos, a média de notícias agregadas aproximou-se à média de um agregador com poucos canais, porém com relevância melhor que a observada no RSSOWL.

Quanto à comparação com outros agregadores no que tange à forma de agregação, entre os agregadores de RSS observados, o iRSS é o único que permite a aplicação de taxonomia para agregação e classificação de itens de RSS. O agregador que mais se aproxima nesse ponto é o BibSonomy, um agregador que não utiliza taxonomia, mas uma folksonomia para recuperar conteúdo, além do fato dele se restringir a bibliografias, não realizar processamento de linguagem natural e nem utilizar técnicas de RI,.

5. Conclusão

5.1 Visão geral

O trabalho descrito nesta dissertação preocupou-se em classificar itens de RSS de modo a possibilitar uma melhor filtragem de notícias aos usuários. Para isto, buscou-se desenvolver um processo para agregar automaticamente RSS segundo uma taxonomia, gerando para cada nó um novo canal de RSS e assim permitindo que o conteúdo fosse agregado mais adequadamente por outros agregadores. Para a agregação

automática segundo a taxonomia, técnicas de recuperação da informação foram utilizadas. Basicamente, o conteúdo foi filtrado por uma taxonomia, que por sua vez permite a reutilização do conteúdo. Para chegar a tal objetivo, desenvolveu-se um protótipo chamado iRSS, que implementa o processo apresentado no capítulo 3.

O processo poderia ser aplicado a qualquer taxonomia, mas para a experimentação do protótipo foi escolhida a taxonomia do IEEE. Também foram selecionados os canais de RSS com conteúdo a ser agregado na experimentação, sendo escolhidos os canais de RSS da biblioteca digital do IEEE. Feitas as escolhas dos canais e da taxonomia, aplicou-se o processo proposto.

Primeiramente, foi recuperado todo o conteúdo dos canais de RSS da biblioteca digital do IEEE durante um período de tempo específico, em seguida foram extraídos os termos relevantes dos itens de RSS, recuperadas as categorias dos itens, e por fim classificados e agregados os itens de acordo com os nós da taxonomia. Dada a classificação, os tópicos da taxonomia funcionam como canais de RSS, fornecendo acesso ao conteúdo agregado por todos os tópicos localizados imediatamente, juntamente com o conteúdo do próprio tópico.

O resultado da agregação foi avaliado quanto aos dados estatísticos de dispersão do número de itens agregados em cada tópico da taxonomia de acordo com a profundidade do tópico. Assim, observou-se que de forma geral a taxonomia funcionou de forma decisiva como meio de agregar conteúdo semanticamente relacionado.

5.2 Contribuições

O trabalho apresentado neste texto mostra que a agregação e disponibilização de RSS apoiada a uma taxonomia pode trazer benefícios aos usuários no que tange ao consumo do conteúdo proveniente de vários canais de RSS. Independentemente da

taxonomia e do conjunto de canais a serem utilizados, a reutilização de itens de RSS previamente filtrados por tópicos de uma taxonomia permite independência na forma de consumo do conteúdo agregado. O agregador produzido neste trabalho funciona como um agregador intermediário, ou seja, cada tópico da taxonomia fornece um canal de RSS.

A principal contribuição deste trabalho refere-se à especificação de um processo de agregação que atua sobre os itens de RSS e os filtra numa taxonomia. Por outro lado, cabe destacar outras contribuições, como:

- A filtragem através do próprio conteúdo de RSS e com base em taxonomia;
- O processo de agregação intermediária;
- A alternativa a listas extensas com palavras-chave cadastradas pelos usuários para a aplicação de filtros (uma tarefa trabalhosa para os usuários).

5.3 Trabalhos Futuros

O trabalho apresentado pode ser estendido em vários aspectos, visto que as formas de extração de termos relevantes, classificação, filtragem e disponibilização do conteúdo agregado podem ser aprimoradas. Entre os possíveis aprimoramentos estão: a seleção de taxonomias, a combinação de taxonomias de diferentes áreas, o uso de ontologias, o uso de outras técnicas de filtragem de mensagens (como a proposta de PINHEIRO *et al.*, (2009)), o agrupamento de notícias relacionadas em termos de relevância ou compatibilidade de conteúdo, o compartilhamento de taxonomias e o compartilhamento de notícias ou canais.

No protótipo construído, a extração de termos relevantes é realizada através de um serviço Web que analisa estatisticamente o texto recebido e, associado a um conjunto de dicionários, extrai os termos relevantes do conteúdo. Existem outras formas

de realizar tal tarefa, que vão desde o cadastramento dos termos relevantes a serem extraídos, até a inferência de termos relevantes. A extração de termos relevantes pode ser melhorada e permitir a captura de conceitos que envolvem o contexto sobre o qual o conteúdo se refere, como em (GIUSEPPE *et al.*, 2008). A inclusão apenas de conceitos na taxonomia proporcionaria a agregação com boa relevância e evitaria a existência de nós excessivamente especializados, com poucos itens de RSS associados. No entanto, o processo descrito trata de termos em geral, não apenas de conceitos.

A classificação do conteúdo agregado é realizada com base nos termos relevantes e categorias extraídas, contudo, cada tópico da taxonomia poderia estar associado a uma lista de sinônimos, o que facilitaria o cruzamento entre termos no momento da classificação. A filtragem pode incluir termos que deveriam ser excluídos do resultado final, personalizando o resultado de acordo com o perfil do usuário e o conjunto de termos especificados. Por fim, o processo proposto poderia abranger também interesses do usuário, capturando suas preferências como os publicadores de conteúdo prediletos ou mesmo sugerindo itens relacionados.

Há a possibilidade da colaboração de um grupo de usuários para detalhamento da taxonomia e conseqüente criação de novos canais de RSS, mas a colaboração não é o foco do processo descrito no capítulo 3. O processo poderia ser estendido a fim de permitir a alteração da taxonomia de forma supervisionada. Assim como uma página Wiki, um tópico poderia ser gerenciado por um supervisor, enquanto a inclusão de tópicos imediatamente abaixo poderia ser da responsabilidade de tal supervisor. Outra tarefa inerente ao supervisor poderia ser o cadastramento de sinônimos ao tópico, como foi proposto anteriormente. A alteração da taxonomia pelos usuários de forma colaborativa refletiria no resultado final da agregação, ao criar novos canais e desenvolver a taxonomia.

Referências Bibliográficas

ARAMPATZIS *et al.*, C. H. A.: Lingüistically-motivated Information Retrieval. Encyclopedia of Library and Information Science. Published by Marcel Dekker, Inc. - New York – Basel, Vol. 69, 2000

ALTINEL, M.; FRANKLIN, M. J., Effcient filtering of XML documents for selective dissemination of information. VLDB, 2000.

BAEZA-YATES, R. A.; RIBEIRO-NETO, B. A., “Modern Information Retrieval”, ACM Press / Addison-Wesley, 1999.

BibSonomy. Knowledge and Data Engineering Group, University of Kassel, Germany. Disponível em: http://www.bibsonomy.org/. Acessado em: 23 de agosto de 2009.

CARDOSO, O. N. P., “Recuperação de Informação”. In: Semana de Ciência da Computação da UFLA, 2000, Lavras. Infocomp, 2000. v. 2.

CHAVES, M. S., "Um estudo e apreciação sobre algoritmos de stemming para a língua portuguesa". IX Jornadas Iberoamericanas de Informática. Cartagena
de Indias - Colômbia, 11-15 agosto de 2003.

DIAO et al., Yfilter: Effcient and scalable filtering of XML documents. InICDE, 2002.

FRAKES, W. B., BAEZA-YATES, R.: Information retrieval: data structures algorithms. ed. Upper Saddle River, NJ : Prentice Hall, 1992.

GEY, F., “Models in Information Retrieval”. Folders of Tutorial Presented at the 19th ACM Conference on Research and Development in Information Retrieval (SIGIR), 1992.

GIUSEPPE, P.; NUNO, S., "Design, Implementation and Evaluation of a new Semantic Similarity metric combining Features and Intrinsic Information Content", in proc. of The 7th International Conference on Ontologies, DataBases, and Applications of Semantics (ODBASE). México, November, 2008.

GRAHAM., “IRA: Intelligent RSS Aggregation”. University of Pennsylvania - School of Engineering & Applied Science, 2005

HAMMERSLEY, B., “Developing Feeds with RSS and Atom”. 1 ed., Califórnia, O’Reilly Media, Abril 2005.

HONRADO et al., "A Word Stemming Algorithm for the Spanish Language". In Proceedings of the Seventh International Symposium on String Processing Information Retrieval (SPIRE'00), A Coruña, Spain, September 27 - 29, 2000.

HRASTNIK, R., “Unleash the Marketing & Publishing Power of RSS”, MarketingStudies, 2009.

IEEE, http://www2.computer.org/portal/web/publications/acmtaxonomy, visitado em junho de 2009

KING, Andrew., “The Evolution of RSS”. Disponível em: http://www.webreference.com/authoring/languages/xml/rss/1, Acessado em: 16 de agosto de 2008.

KING, A. The Evolution of RSS, disponível em: http://www.webreference.com/authoring/languages/xml/rss/1 acessado em: 16 de agosto de 2008.

KUROPKA, D., “Modelle zur Repräsentation natürlichsprachlicher Dokumente”. Logos Verlag, Berlin, 2003.

LEITE, A. A. M., “Modelo Fuzzy para Recuperação de Informação Utilizando Múltiplas Ontologias Relacionadas”, Tese de doutorado, Universidade Estadual de Campinas, UNICAMP, Campinas, SP, 2009.

LEITE, M. A. A. ; RICARTE, I. L. M. . Fuzzy Information Retrieval Model Based on Multiple Related Ontologies. In: 20th IEEE International Conference on Tools with Artificial Intelligence, 2008, Dayton. Proceedings of the 2008 20th IEEE International Conference on Tools with Artificial Intelligence. Washington, DC : IEEE Computer Society, 2008.

LOPES, M. C. S., "Mineração de Dados Textuais Utilizando Técnicas de Clustering para o Idioma Português", COPPE/UFRJ, D.SC.,Engenharia Civil – Rio de Janeiro, RJ, Brasil, 2004.

LUK et al., “A Survey of Search Engines for XML Documents”. Department of Computing, Hong Kong Polytechnic University, 2000.

MAZIERO, E. G., “Métodos simbólicos para a simplificação textual”, Relatório de Atividades, Universidade de Sao Paulo – USP, São Paulo, 2008

MANNING et al., “An introduction to information retrieval”, Cambridge, Draft of july 12, 2008

MIRANDA, Oscar Gomes de. “VIF – Uma Estrutura de índice invertido e blocos baseada em uma B+-Tree”, Dissertação de Mestrado, Universidade Federal de Pernambuco, Recife, Agosto 2003

MOSHFEGHI, Y., “Intelligent RSS Aggregator”. Dissertação de Mestrado, Department of Computing Science, University of Glasgow, Glasgow, Setembro 2007

PETROVIC et al., CMS-ToPSS: Efficient Dissemination of RSS Documents, University of Toronto, 2005.

PETROVIC, M.; LIU, H.; JACOBSEN, H., G-ToPSS - fast filtering of graph-based metadata. In the 14th International World Wide Web Conference, Chiba, Japan, May 2005.

PINHEIRO et al., "Autonomic collaborative RSS: An implementation of autonomic data using data killing patterns," cscwd, pp.492-497, 2009 13th International Conference on Computer Supported Cooperative Work in Design, 2009

PIRES, MARINA MELO, Agrupamento incremental e hierárquico de documentos, Dissertação de Mestrado, Universidade Federal do Rio de Janeiro, COPPE, Engenharia Civil, Rio de Janeiro, Junho 2008.

POLYVYANYY, A.; KUROPKA D., "A Quantitative Evaluation of the Enhanced Topic-Based Vector Space Model", Hasso Plattner Institut, Potsdam, Germany, 2007.

REZENDE, S. O., "Sistemas Inteligentes - Fundamentos e Aplicações". Editora Manole, 2002.

RIJSBERGEN et al., "Information Retrieval", Information Retrieval Group, University of Glasgow, 2 ed., London, Butterworths, 1979

RSS Board: Really Simple Syndication. Disponível em: http://www.rssboard.org/rss-specification. Acessado em: 16 de agosto de 2008.

RSS W3C: Really Simple Syndication. Disponível em http://validator.w3.org/feed/docs/rss2.html. Acessado em: 3 de setembro de 2009.

RSSOwl. SOURCEFORGE.NET. Disponível em: http://www.rssowl.org. Acessado em: 23 de agosto de 2009.

SALTON et al., "A Vector Space Model for Automatic Indexing", Communications of the ACM, vol. 18, nr. 11, pag. 613–620. 1975.

SALTON, G., "Automatic Text Processing — The Transformation, Analysis, and Retrieval of Information by Computer". Addison-Wesley, 1989

SALTON, G., "Introduction to Modern Information Retrieval" (McGraw-Hill Computer Science Series), McGraw-Hill Companies, 1983.

SALTON et al., "Term Weighting Approaches in Automatic Text Retrieval". Fachbericht, Ithaca, NY, USA, 1987.

SPARK-JONES, K., Willet, P.: Readings in Information Retrieval. San Francisco: Morgan Kaufmann, 1997.

STRZALKOWSKI, T., "Natural language information retrieval". Dordrecht: Text, Speech and Language Technology; v.7. Kluwer Academic, 1999.

XML: Extensible Markup Language. W3C. Disponível em: http://www.w3.org/XML/. Acessado em: 16 de agosto de 2008.

XPath: XML Path Language (XPath) Version 1.0, W3C, 1999 . Disponível em: http://www.w3.org/TR/xpath. Acessado em: 16 de agosto de 2008.

XSL: Extensible Stylesheet Language, W3C. Disponível em: http://www.w3.org/Style/XSL/ . Acessado em: 4 de agosto de 2009.

WIVES, L. K; DE OLIVEIRA, J. P. M., "Técnicas de descoberta de conhecimento em textos aplicadas à inteligência competitiva". Teste de Doutorado em Ciência da Computação, 2001.

Apêndice A: Taxonomia (IEEE Computer Society - Keywords)

Áreas:

1. General Literature
2. Hardware
3. Computer Systems Organization
4. Software/Software Engineering
5. Data
6. Theory of Computation
7. Mathematics of Computing
8. Information Technology and Systems
9. Computing Methodologies
10. Computer Applications
11. Computing Milieux

Estrutura detalhada da taxonomia:

- 1. General Literature
  - 0. General
    - 1. Biographies/Autobiographies
    - 2. Conference Proceedings
    - 3. General Literary Works
  - 1. Introductory and Survey
  - 2. Reference
  - 13. Miscellaneous
- 2. Hardware
  - 0. General
  - 1. Control Structures and Microprogramming
    - 0. General
    - 1. Control Design Styles
      - 1. Hardwired control
      - 2. Microprogrammed logic arrays
      - 3. Writable control store
    - 2. Control Structure Performance Analysis and Design Aids
      - 1. Automatic synthesis
      - 2. Formal models
      - 3. Simulation

3. Control Structure Reliability, Testing, and Fault-Tolerance
    1. Diagnostics
    2. Error-checking
    3. Redundant design
    4. Test generation
4. Microprogram Design Aids
    1. Firmware engineering
    2. Languages and compilers
    3. Machine-independent microcode generation
    4. Optimization
    5. Verification
5. Microcode Applications
    1. Direct data manipulation
    2. Firmware support of operating systems/instruction sets
    3. Instruction set interpretation
    4. Peripheral control
    5. Special-purpose
13. Miscellaneous
    1. Emerging technologies

2. Arithmetic and Logic Structures

3.

0. General
1. Design Styles
    1. Calculator
    2. Parallel
    3. Pipeline
    4. Multiple valued logic
2. Performance Analysis and Design Aids
    1. Simulation
    2. Verification
    3. Worst-case analysis
3. Reliability, Testing, and Fault-Tolerance
    1. Diagnostics
    2. Error-checking
    3. Redundant design
    4. Test generation
4. High-Speed Arithmetic
    1. Algorithms
    2. Cost/performance
13. Miscellaneous

4. Memory Structures

0. General
1. Semiconductor Memories
    1. DRAM
    2. ROM
    3. SRAM
2. Design Styles
    1. Associative memories
    2. Cache memories
    3. Interleaved memories
    4. Mass storage

5. Primary memory
6. Sequential-access memory
7. Shared memory
8. Virtual memory

3. Performance Analysis and Design Aids
1. Formal models
2. Simulation
3. Worst-case analysis

4. Reliability, Testing, and Fault-Tolerance
1. Diagnostics
2. Error-checking
3. Redundant design
4. Test generation

13. Miscellaneous

5. I/O and Data Communications
0. General

1. Data Communications Devices
1. Processors
2. Receivers
3. Transmitters

2. Input/Output Devices
1. Channels and controllers
2. Data terminals and printers
3. Image display
4. Voice

3. Interconnections (Subsystems)
1. Asynchronous/synchronous operation
2. Fiber optics
3. Interfaces
4. Parallel I/O
5. Physical structures
6. Topology
7. Web technologies
8. Wireless systems

4. Performance Analysis and Design Aids
1. Formal models
2. Simulation
3. Verification
4. Worst-case analysis

5. Reliability, Testing, and Fault-Tolerance
1. Built-in tests
2. Diagnostics
3. Error-checking
4. Hardware reliability
5. Redundant design
6. Test generation

13. Miscellaneous

6. Register-Transfer-Level Implementation
0. General

1. Design
1. Arithmetic and logic units

2. Control design
3. Data-path design
4. Memory design
5. Styles

2. Design Aids
1. Automatic synthesis
2. Hardware description languages
3. Optimization
4. Simulation
5. Verification

3. Reliability and Testing
1. Built-in tests
2. Error-checking
3. Redundant design
4. Test generation
5. Testability

13. Miscellaneous

7. Logic Design
0. General

1. Design Styles
1. Cellular arrays and automata
2. Combinational logic
3. Logic arrays
4. Memory control and access
5. Memory used as logic
6. Parallel circuits
7. Sequential circuits

2. Reliability and Testing
1. Built-in tests
2. Error-checking
3. Redundant design
4. Test generation
5. Testability

3. Design Aids
1. Automatic synthesis
2. Hardware description languages
3. Optimization
4. Simulation
5. Switching theory
6. Verification

13. Miscellaneous

8. Integrated Circuits
0. General

1. Types and Design Styles
1. Advanced technologies
2. Algorithms implemented in hardware
3. Gate arrays
4. Input/output circuits
5. Memory technologies
6. Microprocessors and microcomputers
7. Network connectivity chips

8. Standard cells
9. VLSI

2. Design Aids
1. Graphics
2. Layout
3. Placement and routing
4. Simulation
5. Verification

3. Reliability and Testing
1. Built-in tests
2. Error-checking
3. Fault injection
4. Redundant design
5. Test generation
6. Testability

13. Miscellaneous

9. Performance and Reliability
0. General
1. Reliability, Testing, and Fault-Tolerance
2. Performance Analysis and Design Aids
13. Miscellaneous

10. Power Management
1. Low-power design
2. Energy-aware systems

13. Miscellaneous
1. Design management

3. Computer Systems Organization
0. General
1. Emerging technologies
2. Hardware/software interfaces
3. Instruction set design
4. Modeling of computer architecture
5. System architectures, integration and modeling
6. Systems specification methodology

1. Processor Architectures
0. General
1. Single Data Stream Architectures
1. MISD processors
2. Pipeline processors
3. RISC/CISC, VLIW architectures
4. SISD processors
5. Von Neumann architectures

2. Multiple Data Stream Architectures (Multiprocessors)
1. Array and vector processors
2. Associative processors
3. Connection machines
4. Interconnection architectures
5. Load balancing and task assignment
6. MIMD processors
7. Parallel processors
8. Pipeline processors

9. TC scheduling and synchronization
10. SIMD processors

3. Other Architecture Styles
1. Adaptable architectures
2. Analog computers
3. Capability architectures
4. Cellular architecture
5. Dataflow architectures
6. Heterogeneous (hybrid) systems
7. High-level language architectures
8. Multithreaded processors
9. Neural nets
10. Neurocomputers
11. Pipeline processors
12. Stack-oriented processors

4. Parallel Architectures
1. Distributed architectures
2. Mobile processors
3. Real-time distributed
4. Scheduling and task partitioning

13. Miscellaneous
1. Analog computers
2. Hybrid systems

2. Communication/Networking and Information Technology
0. General
1. Architecture
2. Data communications
3. Emerging technologies
4. Infrastructure protection
5. Interprocessor communications
6. Network-level security and protection
7. OSI reference model
8. Standards

1. Network Architecture and Design
1. ATM
2. Centralized networks
3. Circuit-switching networks
4. Distributed networks
5. Frame relay networks
6. ISDN
7. Network communications
8. Network topology
9. Packet-switching networks
10. Store and forward networks
11. Wireless communication

2. Network Protocols
1. Applications
2. Protocol architecture
3. Protocol verification
4. Routing protocols

3. Network Operations

1. Network management
2. Network monitoring
3. Public networks

4. Distributed Systems
    1. Client/server
    2. Distributed applications
    3. Distributed databases
    4. Network operating systems
5. Local-Area Networks
    1. Access schemes
    2. Buses
    3. Ethernet
    4. High-speed
    5. Internet
    6. Token rings
6. Internetworking
    1. Bridges
    2. Gateways
    3. Multicast
    4. Protocols
    5. Routers
    6. Standards
7. Wide-area networks
    1. CATV
    2. Optical fiber
    3. Sensor networks
    4. Telephony
    5. Wireless
8. Mobile Computing
    1. Algorithm/protocol design and analysis
    2. Architectures
    3. Mobile communication systems
    4. Mobile environments
    5. Support services

13. Miscellaneous

3. Special-Purpose and Application-Based Systems
    1. Application studies resulting in better multiple-processor systems
    2. Microprocessor/microcomputer applications
    3. Process control systems
    4. Real-time and embedded systems
    5. Reconfigurable hardware
    6. Signal processing systems
    7. Smartcards
    8. Ubiquitous computing
4. Performance of Systems
    1. Design studies
    2. Fault tolerance
    3. Measurement techniques
    4. Modeling techniques
    5. Performance attributes
    6. Reliability, availability, and serviceability

- 7. Measurement, evaluation, modeling, simulation of multiple-processor systems
- 5. Computer System Implementation
  - 0. General
  - 1. Large and Medium ("Mainframe")
    - 1. Super (very large) computers
  - 2. Minicomputers
  - 3. Microcomputers
    - 1. Microprocessors
    - 2. Personal computers
    - 3. Portable devices
    - 4. Workstations
  - 4. VLSI Systems
    - 1. Impact of VLSI on system design
  - 5. Servers
    - 1. Web server
    - 2. Web browser
  - 6. Multiprocessor Systems
  - 7. Wearable Computers
  - 13. Miscellaneous
- 4. Software/Software Engineering
  - 0. General
  - 1. Programming Techniques
    - 0. General
    - 1. Applicative (Functional) Programming
    - 2. Automatic Programming
    - 3. Concurrent Programming
    - 4. Sequential Programming
    - 5. Object-Oriented Programming
    - 6. Logic Programming
    - 7. Visual Programming
    - 8. Distributed programming
    - 13. Miscellaneous
  - 2. Software Engineering
    - 0. General
      - 1. Protection mechanisms
      - 2. Software psychology
      - 3. Software engineering for Internet projects
      - 4. Standards
      - 5. Surveys of historical development of one particular area
    - 1. Requirements/Specifications
      - 1. Analysis
      - 2. Elicitation methods
      - 3. Languages
      - 4. Management
      - 5. Methodologies
      - 6. Process
      - 7. Specification
      - 8. Tools
      - 9. Validation
    - 2. Design Tools and Techniques
      - 1. CASE

2. Decision tables
3. Distributed/Internet based software engineering tools and techniques
4. Modules and interfaces
5. Programmer workbench

3. Coding Tools and Techniques
1. Object-oriented programming
2. Pretty printers
3. Program editors
4. Reentrant code
5. Standards
6. Structured programming
7. Top-down programming

4. Software/Program Verification
1. Assertion checkers, assertion languages, performance
2. Class invariants
3. Correctness proofs
4. Formal methods
5. Model checking
6. Programming by contract
7. Reliability
8. Statistical methods
9. Validation

5. Testing and Debugging
1. Code inspections and walkthroughs
2. Debugging aids
3. Diagnostics
4. Distributed debugging
5. Dumps
6. Error handling and recovery
7. Monitors
8. Reliability
9. Symbolic execution
10. Test levels
11. Testing strategies
12. Test design
13. Test coverage of code
14. Test coverage of specifications
15. Test execution
16. Test documentation
17. Test management
18. Testing tools
19. Tracing
20. Usability testing

6. Programming Environments/Construction Tools
1. Environments for multiple-processor systems
2. Graphical environments
3. Integrated environments
4. Interactive environments
5. Programmer workbench

7. Distribution, Maintenance, and Enhancement
1. Conversion from sequential to parallel forms

2. Corrections
3. Documentation
4. Enhancement
5. Evolving Internet applications
6. Extensibility
7. Maintainability
8. Maintenance management
9. Maintenance measurement
10. Maintenance planning
11. Maintenance process
12. Portability
13. Restructuring, reverse engineering, and reengineering
14. Version control

8. Metrics/Measurement
    1. Complexity measures
    2. Performance measures
    3. Process metrics
    4. Product metrics
    5. Software science
9. Management
    1. Copyrights
    2. Cost estimation
    3. Enactment
    4. Initiation and scope definition
    5. Organizational management and coordination
    6. Planning
    7. Postclosure activities
    8. Productivity
    9. Programming teams
    10. Project close out
    11. Project control & modeling
    12. Review and evaluation
    13. Risk management
    14. Schedule and organizational issues
    15. Software acquisition
    16. Time estimation
10. Design
    1. Design concepts
    2. Design notations and documentation
    3. Representation
    4. State diagrams
    5. Evolutionary prototyping
    6. Methodologies
    7. Object-oriented design methods
    8. Quality analysis and evaluation
    9. Rapid prototyping
    10. Representation
11. Software Architectures
    1. Data abstraction
    2. Domain-specific architectures
    3. Information hiding

4. Languages
5. Patterns

12. Interoperability
    1. Data mapping
    2. Distributed objects
    3. Interface definition languages
13. Reusable Software
    1. Domain engineering
    2. Reusable libraries
    3. Reuse models
14. Human Factors in Software Design
    1. User interfaces
15. Software and System Safety
16. Configuration Management
    1. Configuration auditing
    2. Configuration control
    3. Configuration identification
    4. Configuration management process
    5. Configuration status accounting
    6. Software release management and delivery
17. Software Construction
    1. Construction planning
    2. Code design
    3. Code tuning
    4. Data design and management
    5. Error processing
    6. Source code organization
    7. Code documentation
    8. Construction QA
    9. Programming paradigms
    10. System integration and implementation
18. Software Engineering Process
    1. Life cycle
    2. Process infrastructure
    3. Process measurement
    4. Process definition
    5. Software process models
    6. Qualitative process analysis
    7. Process implementation and change
19. Software Quality/SQA
    1. Quality concepts
    2. Planning for SQA and V&V
    3. Methods for SQA and V&V
    4. Measurement applied to SQA and V&V
13. Miscellaneous
    1. Software libraries
    2. System issues

3. Programming Languages
    0. General
        1. Standards
    1. Formal Definitions and Theory

1. Semantics
2. Syntax

2. Language Classifications
    1. Applicative (functional) languages
    2. Componentware
    3. Compression technologies
    4. Concurrent, distributed, and parallel languages
    5. Constraint and logic languages
    6. Dataflow languages
    7. Design languages
    8. Development tools
    9. Extensible languages
    10. Java
    11. Macro and assembly languages
    12. Microprogramming languages
    13. Multiparadigm languages
    14. Nondeterministic languages
    15. Nonprocedural languages
    16. Object-oriented languages
    17. Query languages
    18. Scripting languages
    19. Specialized application languages
    20. Very high-level languages
3. Language Constructs and Features
    1. Abstract data types
    2. Classes and objects
    3. Concurrent programming structures
    4. Constraints
    5. Control structures
    6. Coroutines
    7. Data types and structures
    8. Distributed objects, components, containers
    9. Dynamic storage management
    10. Frameworks
    11. Inheritance
    12. Input/output
    13. Modules, packages
    14. Patterns
    15. Polymorphism
    16. Procedures, functions, and subroutines
    17. Recursion
4. Processors
    1. Code generation
    2. Compilers
    3. Debuggers
    4. Incremental compilers
    5. Interpreters
    6. Memory management
    7. Optimization
    8. Parsing
    9. Preprocessors

10. Retargetable compilers
11. Runtime environments
12. Translator writing systems and compile

13. Miscellaneous

4. Operating Systems

0. General

1. Process Management

1. Concurrency
2. Deadlocks
3. Multiprocessing/multiprogramming/multitasking
4. Mutual exclusion
5. Scheduling
6. Synchronization
7. Threads

2. Storage Management

1. Allocation/deallocation strategies
2. Distributed memories
3. Garbage collection
4. Main memory
5. Secondary storage
6. Segmentation
7. Storage hierarchies
8. Swapping
9. Virtual memory

3. File Systems Management

1. Access methods
2. Directory structures
3. Distributed file systems
4. File organization
5. Maintenance

4. Communications Management

1. Buffering
2. Input/output
3. Message sending
4. Network communication
5. Terminal management

5. Reliability

1. Backup procedures
2. Checkpoint/restart
3. Disconnected operation
4. Fault-tolerance
5. High availability
6. Verification

6. Security and Privacy Protection

1. Access controls
2. Authentication
3. Cryptographic controls
4. Information flow controls
5. Invasive software
6. Security kernels
7. Verification

7. Organization and Design
    1. Batch processing systems
    2. Distributed systems
    3. Hierarchical design
    4. Interactive systems
    5. Real-time systems and embedded systems
    6. Parallel systems
7. Performance
    1. Measurements
    2. Modeling and prediction
    3. Monitors
    4. Operational analysis
    5. Queuing theory
    6. Simulation
    7. Stochastic analysis
9. Systems Programs and Utilities
    1. Command and control languages
    2. Linkers
    3. Loaders
    4. Window managers
10. Support for Adaptation
    1. Application-aware adaptation
    2. Application-transparent adaptation
    3. Fidelity, agility, and stability
    4. Low-bandwidth operation
    5. Transcoding
13. Miscellaneous

5. Data
0. General
    1. Data communications aspects
    2. Data dependencies
    3. Data encryption
    4. File organization
    5. Knowledge and data engineering tools and techniques
    6. System applications and experience
1. Data Structures
    1. Arrays
    2. Distributed data structures
    3. Distributed file systems
    4. Graphs and networks
    5. Lists, stacks, and queues
    6. Records
    7. Tables
    8. Trees
2. Data Storage Representations
    1. Composite structures
    2. Contiguous representations
    3. Hash-table representations
    4. Linked representations
    5. Object representation
    6. Primitive data items

3. Data Encryption
    1. Code breaking
    2. DES
    3. Public key cryptosystems
    4. Standards
4. Coding and Information Theory
    1. Data compaction and compression
    2. Error control codes
    3. Normal models of communication
    4. Nonsecret encoding schemes
5. Files
    1. Backup/recovery
    2. Optimization
    3. Organization/structure
    4. Sorting/searching
13. Miscellaneous

6. Theory of Computation
0. General
1. Computation by Abstract Devices
    0. General
    1. Models of Computation
        1. Automata
        2. Bounded-action devices
        3. Computability theory
        4. Relations between models
        5. Self-modifying machines
        6. Unbounded-action devices
    2. Modes of Computation
        1. Alternation and nondeterminism
        2. Interactive and reactive computation
        3. Online computation
        4. Parallelism and concurrency
        5. Probabilistic computation
        6. Relations among modes
        7. Relativized computation
    3. Complexity Measures and Classes
        1. Complexity hierarchies
        2. Machine-independent complexity
        3. Reducibility and completeness
        4. Relations among complexity classes
        5. Relations among complexity measures
    13. Miscellaneous
2. Analysis of Algorithms and Problem Complexity
    0. General
    1. Numerical Algorithms and Problems
        1. Computation of transforms
        2. Computations in finite fields
        3. Computations on matrices
        4. Computations on polynomials
        5. Number-theoretic computations
    2. Nonnumerical Algorithms and Problems

1. Complexity of proof procedures
2. Computations on discrete structures
3. Geometrical problems and computations
4. Pattern matching
5. Routing and layout
6. Sequencing and scheduling

3. Sorting and searching

13. Miscellaneous

3. Logics and Meanings of Programs

0. General

1. Specifying and Verifying and Reasoning about Programs

1. Assertions
2. Invariants
3. Logics of programs
4. Mechanical verification
5. Pre- and post-conditions
6. Specification techniques

2. Semantics of Programming Languages

1. Algebraic approaches to semantics
2. Denotational semantics
3. Operational semantics
4. Partial evaluation
5. Process models
6. Program analysis

3. Studies of Program Constructs

1. Control primitives
2. Functional constructs
3. Object-oriented constructs
4. Program and recursion schemes
5. Type structure

13. Miscellaneous

4. Mathematical Logic and Formal Languages

0. General

1. Mathematical Logic

1. Computability theory
2. Computational logic
3. Lambda calculus and related systems
4. Logic and constraint programming
5. Mechanical theorem proving
6. Modal logic
7. Model theory
8. Proof theory
9. Recursive function theory
10. Set theory
11. Temporal logic

2. Grammars and Other Rewriting Systems

1. Decision problems
2. Grammar types
3. Parallel rewriting systems
4. Parsing
5. Thue systems

3. Formal Languages
    1. Algebraic language theory
    2. Classes defined by grammars or automata
    3. Classes defined by resource-bounded automata
    4. Decision problems
    5. Operations on languages
  13. Miscellaneous
 13. Miscellaneous
7. Mathematics of Computing
 0. General
 1. Numerical Analysis
  0. General
    1. Computer arithmetic
    2. Conditioning and ill-conditioning
    3. Error analysis
    4. Interval arithmetic
    5. Multiple precision arithmetic
    6. Numerical algorithms
    7. Parallel algorithms
    8. Stability and instability
  1. Interpolation
    1. Difference formulas
    2. Extrapolation
    3. Interpolation formulas
    4. Smoothing
    5. Spline and piecewise polynomial interpolation
  2. Approximation
    1. Approximation of surfaces and contours
    2. Chebyshev approximation and theory
    3. Elementary function approximation
    4. Fast Fourier transforms
    5. Least squares approximation
    6. Linear approximation
    7. Minimax approximation and algorithms
    8. Nonlinear approximation
    9. Rational approximation
    10. Special function approximations
    11. Spline and piecewise polynomial approximation
    12. Wavelets and fractals
  3. Numerical Linear Algebra
    1. Conditioning
    2. Determinants
    3. Eigenvalues and eigenvectors
    4. Error analysis
    5. Linear systems
    6. Matrix inversion
    7. Pseudoinverses
    8. Singular value decomposition
    9. Sparse, structured, and very large systems
  4. Quadrature and Numerical Differentiation
    1. Adaptive and iterative quadrature

2. Automatic differentiation
3. Equal interval integration
4. Error analysis
5. Finite difference methods
6. Gaussian quadrature
7. Iterative methods
8. Multidimensional (multiple) quadrature

5. Roots of Nonlinear Equations
1. Continuation (homotopy) methods
2. Convergence
3. Error analysis
4. Iterative methods
5. Polynomials, methods for
6. Systems of equations

6. Optimization
1. Constrained optimization
2. Convex programming
3. Global optimization
4. Gradient methods
5. Inter programming
6. Least squares methods
7. Linear programming
8. Nonlinear programming
9. Quadratic programming methods
10. Simulated annealing
11. Stochastic programming
12. Unconstrained optimization

7. Ordinary Differential Equations
1. Boundary value problems
2. Chaotic systems
3. Convergence and stability
4. Differential-algebraic equations
5. Error analysis
6. Finite difference methods
7. Initial value problems
8. Multistep and multivalue methods
9. One-step (single step) methods
10. Stiff equations

8. Partial Differential Equations
1. Domain decomposition methods
2. Elliptic equations
3. Finite difference methods
4. Finite element methods
5. Finite volume methods
6. Hyperbolic equations
7. Inverse problems
8. Iterative solution techniques
9. Method of lines
10. Multigrid and multilevel methods
11. Parabolic equations
12. Spectral methods

9. Integral Equations
    1. Delay equations
    2. Fredholm equation
    3. Intro-differential equations
    4. Volterra equations
10. Applications
13. Miscellaneous

2. Discrete Mathematics
0. General
1. Combinatorics
    1. Combinatorial algorithms
    2. Counting problems
    3. Generating functions
    4. Permutations and combinations
    5. Recurrences and difference equations
2. Graph Theory
    1. Graph algorithms
    2. Graph labeling
    3. Hypergraphs
    4. Network problems
    5. Path and circuit problems
    6. Trees
3. Applications
13. Miscellaneous

3. Probability and Statistics
1. Contingency table analysis
2. Correlation and regression analysis
3. Distribution functions
4. Experimental design
5. Markov processes
6. Multivariate statistics
7. Nonparametric statistics
8. Probabilistic algorithms
9. Queuing theory
10. Random number generation
11. Reliability and life testing
12. Renewal theory
13. Robust regression
14. Statistical computing
15. Statistical software
16. Stochastic processes
17. Survival analysis
18. Time series analysis

4. Mathematical Software
1. Algorithm design and analysis
2. Certification and testing
3. Documentation
4. Efficiency
5. Parallel and vector implementations
6. Portability
7. Reliability and robustness

8. User interfaces
9. Verification

13. Miscellaneous

1. Queuing theory

8. Information Technology and Systems

0. General

1. Infrastructure Protection

1. Models and Principles

0. General

1. Systems and Information Theory

1. General systems theory
2. Information theory
3. Value of information

2. User/Machine Systems

1. Human factors
2. Human-centered computing
3. Human information processing
4. Software psychology

13. Miscellaneous

2. Database Management

0. General

1. Security, integrity, and protection
2. Database design, modeling and management
3. Query design and implementation languages

1. Logical Design

1. Data models
2. Database architectures
3. Database integration
4. Database models
5. Normal forms
6. Schema and subschema

2. Physical Design

1. Access methods
2. Deadlock avoidance
3. Indexing methods
4. Physical database design prototypes
5. Recovery and restart

3. Languages

1. Data description languages
2. Data manipulation languages
3. Database (persistent) programming languages
4. Database semantics
5. Query languages
6. Report writers

4. Systems

1. Active databases
2. Buffer management
3. Concurrency
4. Distributed databases
5. Multimedia databases
6. Object-oriented databases

7. Parallel databases
8. Query processing
9. Relational database
10. Rule-based databases
11. Spatial databases
12. Statistical databases
13. Temporal databases
14. Textual databases
15. Transaction processing
16. Workflow management

5. Heterogeneous Databases
    1. Data translation
    2. Program translation
6. Database Machines
7. Database Administration
    1. Data dictionary/directory
    2. Data warehouse and repository
    3. Logging and recovery
    4. Security, integrity, and protection
8. Database Applications
    1. Bioinformatics (genome or protein) databases
    2. Clustering, classification, and association rules
    3. Data and knowledge visualization
    4. Data mining
    5. Feature extraction or construction
    6. Knowledge management applications
    7. Image databases
    8. Interactive data exploration and discovery
    9. Mining methods and algorithms
    10. Modeling structured, textual and multimedia data
    11. Personalization
    12. Text mining
    13. Web mining
    14. Scientific databases
    15. Spatial databases and GIS
    16. Statistical databases
13. Miscellaneous

3. Information Storage and Retrieval
    0. General
        1. Web Search
    1. Context Analysis and Indexing
        1. Abstracting methods
        2. Dictionaries
        3. Indexing methods
        4. Linguistic processing
        5. Thesauruses
    2. Information Storage
        1. Document/file management
        2. File organization
        3. Record classification
        4. Storage/repositories

3. Information Search and Retrieval
    1. Clustering
    2. Information filtering
    3. Internet search
    4. Metadata
    5. Query formulation
    6. Relevance feedback
    7. Retrieval models
    8. Search process
    9. Selection process
4. Systems and Software
    1. Current awareness systems
    2. Distributed systems
    3. Information networks
    4. Performance evaluation
    5. Question-answering systems
    6. User profiles and alert services
5. Online Information Services
    1. Commercial services
    2. Data sharing
    3. DOM
    4. HTML/DHTML CSS
    5. Web-based services
    6. XML/XSL/RDF
6. Library Automation
    1. Large text archives
7. Digital Libraries
    1. Collection
    2. Dissemination
    3. Standards
    4. Systems issues
    5. User issues

13. Miscellaneous

4. Information Technology and Systems Applications
    0. General
    1. Office Automation
        1. Desktop publishing
        2. Equipment
        3. Groupware
        4. Spreadsheets
        5. Time management
        6. Word processing
        7. Workflow management
    2. Types of Systems
        1. Decision support
        2. Logistics
    3. Communications Applications
        1. Bulletin boards
        2. Computer conferencing, teleconferencing, and videoconferencing
        3. Electronic mail
        4. Information browsers

5. Videotex
13. Miscellaneous

5. Information Interfaces and Representation (HCI)
0. General
1. Multimedia Information Systems
1. Animations
2. Artificial, augmented, and virtual realities
3. Audio input/output
4. Evaluation/methodology
5. Hypertext navigation and maps
6. Image/video retrieval
7. Video
2. User Interfaces
1. Auditory (non-speech) feedback
2. Benchmarking
3. Design for wearability
4. Ergonomics
5. Evaluation/methodology
6. Graphical user interfaces
7. Haptic I/O
8. Input devices and strategies
9. Interaction styles
10. Natural language
11. Prototyping
12. Screen design
13. Standardization
14. Style guides
15. Theory and methods
16. Training, help, and documentation
17. User-centered design
18. User interface management systems
19. Vision I/O
20. Voice I/O
21. Windowing systems
3. Group and Organization Interfaces
1. Asynchronous interaction
2. Collaborative computing
3. Computer-supported cooperative work
4. Evaluation/methodology
5. Organizational design
6. Synchronous interaction
7. Theory and models
8. Web-based interaction
4. Hypertext/Hypermedia
1. Architectures
2. Navigation
3. Theory
4. User issues
5. Sound and Music Computing
1. Methodologies and techniques
2. Modeling

3. Signal analysis, synthesis, and processing
4. Systems

13. Miscellaneous

13. Miscellaneous

9. Computing Methodologies

0. General

1. Symbolic and algebraic manipulation

0. General

1. Expressions and Their Representation

1. Representations
2. Simplification of expressions

2. Algorithms

1. Algebraic algorithms
2. Algorithms for data and knowledge management
3. Analysis of algorithms
4. Nonalgebraic algorithms
5. Performance evaluation of algorithms and systems

3. Languages and Systems

1. Evaluation strategies
2. Nonprocedural languages
3. Special-purpose algebraic systems
4. Special-purpose hardware
5. Substitution mechanisms

4. Applications

13. Miscellaneous

2. Artificial Intelligence

0. General

1. Cognitive simulation
2. Philosophical foundations

1. Applications and Expert Knowledge-Intensive Systems

1. Cartography
2. Computer vision
3. Decision support
4. Education
5. Environment
6. Games and infotainment
7. Industrial automation
8. Law
9. Mathematics
10. Medicine and science
11. Military
12. Natural language interfaces
13. Office automation
14. Space
15. Transportation

2. Automatic Programming

1. Automatic analysis of algorithms
2. Program modification
3. Program synthesis
4. Program transformation
5. Program verification

3. Deduction and Theorem Proving and Knowledge Processing
    1. Answer/reason extraction
    2. Constraint-based processing
    3. Deduction
    4. Inference engines
    5. Logic processing
    6. Logic programming
    7. Mathematical induction
    8. Metatheory
    9. Nonmonotonic reasoning and belief revision
    10. Resolution
    11. Rule-based processing
    12. Uncertainty, "fuzzy," and probabilistic reasoning
4. Knowledge Representation Formalisms and Methods
    1. Agent communication languages
    2. Distributed representations
    3. Frames and scripts
    4. Knowledge base management
    5. Knowledge base verification
    6. Modal logic
    7. Predicate logic
    8. Relation systems
    9. Representation languages
    10. Representations (procedural and rule-based)
    11. Semantic networks
    12. Storage mechanisms
    13. Temporal logic
5. Programming Languages and Software
    1. Expert and knowledge-intensive system tools and techniques
6. Learning
    1. Analogies
    2. Concept learning
    3. Connectionism and neural nets
    4. Heuristics design
    5. Induction
    6. Knowledge acquisition
    7. Machine learning
    8. Language acquisition
    9. Parameter learning
7. Natural Language Processing
    1. Discourse
    2. Language generation
    3. Language models
    4. Language parsing and understanding
    5. Language summarization
    6. Machine translation
    7. Speech recognition and synthesis
    8. Text analysis
    9. Web text analysis
8. Problem Solving, Control Methods, and Search
    1. Backtracking

2. Constraint satisfaction
3. Control theory
4. Dynamic programming
5. Graph and tree search strategies
6. Heuristic methods
7. Plan execution, formation, and generation
8. Scheduling

9. Robotics
    1. Autonomous vehicles
    2. Biorobotics
    3. Commercial robots and applications
    4. Kinematics and dynamics
    5. Manipulators
    6. Nanorobots
    7. Neuromorphic computing
    8. Operator interfaces
    9. Propelling mechanisms
    10. Sensors
    11. Workcell organization and planning
    12. Vision
10. Vision and Scene Understanding
    1. 3D/stereo scene analysis
    2. Architecture and control structures
    3. Intensity, color, photometry, and thresholding
    4. Modeling and recovery of physical attributes
    5. Motion
    6. Perceptual reasoning
    7. Representations, data structures, and transforms
    8. Shape
    9. Texture
    10. Video analysis
11. Distributed Artificial Intelligence
    1. Coherence and coordination
    2. Intelligent agents
    3. Languages and structures
    4. Multiagent systems
12. Intelligent Web Services and Semantic Web
    1. Intelligent Web service languages
    2. Internet reasoning services
    3. Ontology design
    4. Ontology languages
13. Knowledge Management
    1. Knowledge acquisition
    2. Knowledge engineering methodologies
    3. Knowledge life cycles
    4. Knowledge maintenance
    5. Knowledge modeling
    6. Knowledge personalization and customization
    7. Knowledge publishing
    8. Knowledge retrieval
    9. Knowledge reuse

10. Knowledge valuation

13. Miscellaneous

1. Adaptive hypermedia
2. Computational neuroscience
3. Evolutionary computing and genetic algorithms
4. Wearable AI

3. Computer Graphics

0. General

1. Hardware Architecture

1. Graphics processors
2. Hardcopy devices
3. Input devices
4. Parallel processing
5. Raster display devices
6. Storage devices
7. Three-dimensional displays
8. Vector display devices

2. Graphics Systems

1. Distributed/network graphics
2. Remote systems
3. Stand-alone systems

3. Picture/Image Generation

1. Antialiasing
2. Bitmap and frame buffer operations
3. Digitizing and scanning
4. Display algorithms
5. Image-based rendering
6. Line and curve generation
7. Viewing algorithms

4. Graphics Utilities

1. Application packages
2. Device drivers
3. Graphics editors
4. Graphics packages
5. Meta files
6. Paint systems
7. Picture description languages
8. Software support
9. Virtual device interfaces

5. Computational Geometry and Object Modeling

1. Boundary representations
2. Constructive solid geometry
3. Curve, surface, solid, and object representations
4. Geometric algorithms, languages, and systems
5. Hierarchy and geometric transformations
6. Modeling packages
7. Modeling from video
8. Object hierarchies
9. Physically based modeling
10. Splines

6. Methodology and Techniques

1. Device independence
2. Ergonomics
3. Graphics data structures and data types
4. Interaction techniques
5. Languages
6. Standards

7. Three-Dimensional Graphics and Realism
1. Animation
2. Color, shading, shadowing, and texture
3. Fractals
4. Hidden line/surface removal
5. Radiosity
6. Raytracing
7. Virtual reality
8. Visible line/surface algorithms

8. Applications

13. Miscellaneous

4. Image Processing and Computer Vision

0. General
1. Image displays
2. Image processing software

1. Digitization and Image Capture
1. Camera calibration
2. Imaging geometry
3. Quantization
4. Radiometry
5. Reflectance
6. Sampling
7. Scanning

2. Compression (Coding)
1. Approximate methods
2. Exact coding
3. Model-based coding
4. MP-4 and MP-7
5. Video coding

3. Enhancement
1. Filtering
2. Geometric correction
3. Grayscale manipulation
4. Registration
5. Sharpening and deblurring
6. Smoothing

4. Restoration
1. Inverse filtering
2. Kalman filtering
3. Pseudoinverse restoration
4. Wiener filtering

5. Reconstruction
1. Series expansion methods
2. Summation methods
3. Transform methods

6. Segmentation
    1. Edge and feature detection
    2. Graph-theoretic methods
    3. Markov random fields
    4. Pixel classification
    5. Region growing, partitioning
    6. Relaxation
    7. Stochastic methods
7. Feature Measurement
    1. Feature representation
    2. Invariants
    3. Moments
    4. Projections
    5. Size and shape
    6. Texture
8. Scene Analysis
    1. Color
    2. Depth cues
    3. Image models
    4. Motion
    5. Object recognition
    6. Photometry
    7. Range data
    8. Sensor fusion
    9. Shading
    10. Shape
    11. Stereo
    12. Surface fitting
    13. Time-varying imagery
    14. Tracking
9. Applications
10. Image Representation
    1. Hierarchical
    2. Morphological
    3. Multidimensional
    4. Statistical
    5. Volumetric
13. Miscellaneous

5. Pattern Recognition
    0. General
    1. Models
        1. Deterministic
        2. Fuzzy set
        3. Geometric
        4. Neural nets
        5. Statistical
        6. Structural
        7. Syntactic
    2. Design Methodology
        1. Classifier design and evaluation
        2. Feature evaluation and selection

3. Pattern analysis
3. Clustering
1. Algorithms
2. Similarity measures
4. Applications
1. Arts
2. Computer vision
3. Computational models of vision
4. Face and gesture recognition
5. Government
6. Handwriting analysis
7. Industry
8. Medicine
9. Military
10. Remote sensing
11. Robotics
12. Sciences
13. Signal processing
14. Text processing
15. Waveform analysis
5. Implementation
1. Interactive systems
2. Real-time systems
3. Special architectures
13. Miscellaneous

6. Simulation, Modeling, and Visualization
0. General
1. Simulation Theory
1. Model classification
2. Systems theory
3. Types of simulation
2. Simulation Languages
3. Applications
4. Model Validation and Analysis
5. Model Development
1. Modeling methodologies
6. Simulation Output Analysis
7. Simulation Support Systems
1. Environments
8. Types of Simulation
1. Animation
2. Combined
3. Continuous
4. Discrete event
5. Distributed
6. Gaming
7. Monte Carlo
8. Parallel
9. Visual
9. Visualization
1. Applications

2. Flow visualization
3. Information visualization
4. Multivariate visualization
5. Visual programming and program visualization
6. Visualization systems and software
7. Visualization techniques and methodologies
8. Volume visualization

13. Miscellaneous

7. Document and Text Processing

0. General
1. Document and Text Editing
1. Document management
2. Languages
3. Spelling
4. Version control
2. Document Preparation
1. Desktop publishing
2. Format and notation
3. Hypertext/hypermedia
4. Index generation
5. Languages and systems
6. Markup languages
7. Multi/mixed media
8. Photocomposition/typesetting
9. Scripting languages
10. Standards
3. Index Generation
4. Electronic Publishing
5. Document Capture
1. Document analysis
2. Document indexing
3. Graphics recognition and interpretation
4. Optical character recognition
5. Scanning

13. Miscellaneous

13. Miscellaneous

10. Computer Applications

0. General
1. Administrative Data Processing
1. Business
2. Education
3. Financial
4. Government
5. Law
6. Manufacturing
7. Marketing
8. Military
2. Physical Sciences and Engineering
1. Aerospace
2. Archaeology
3. Astronomy

    4. Chemistry
    5. Earth and atmospheric sciences
    6. Electronics
    7. Engineering
    8. Mathematics and statistics
    9. Physics
3. Life and Medical Sciences
    1. Biology and genetics
    2. Health
    3. Medical information systems
4. Social and Behavioral Sciences
    1. Economics
    2. Psychology
    3. Sociology
5. Arts and Humanities
    1. Architecture
    2. Arts, fine and performing
    3. Fine arts
    4. Language translation
    5. Linguistics
    6. Literature
    7. Music
    8. Performing arts
6. Computer-Aided Engineering
    1. Computer-aided design
    2. Computer-aided manufacturing
7. Computers in Other Systems
    1. Command and control
    2. Consumer products
    3. Industrial control
    4. Military
    5. Process control
    6. Publishing
    7. Real time
8. Internet Applications
    1. Client/server and multitier systems
    2. Databases
    3. Database connectivity
    4. Distributed file systems
    5. Electronic commerce
    6. Engineering design
    7. Games
    8. Health care
    9. Intranet/extranet/VPNs
    10. Libraries/information repositories/publishing
    11. Manufacturing
    12. Middleware/business logic
    13. Network repositories/data mining/backup
    14. Software engineering
    15. Traffic analysis
    16. Transaction software

17. Web browsers
18. Web servers
19. Web site management/development tools

9. Mobile Applications
1. Location-dependent and sensitive
2. Nomadic computing
3. Multimedia applications and multimedia signal processing
4. Pervasive computing
5. Wearable computers and body area networks
6. Wireless sensor networks

13. Miscellaneous

11. Computing Milieux
0. General
1. The Computer Industry
1. Markets
2. Standards
3. Statistics
4. Suppliers

2. History of Computing
1. Hardware
2. People
3. Software
4. Systems
5. Theory

3. Computers and Education
0. General
1. Computer Uses in Education
1. Collaborative learning
2. Computer-assisted instruction
3. Computer-managed instruction
4. Distance learning

2. Computer and Information Science Education
1. Accreditation
2. Computer science education
3. Curriculum
4. Information systems education
5. Literacy
6. Self-assessment

13. Miscellaneous
1. Accreditation
2. Computer literacy

4. Computers and Society
0. General
1. Public Policy Issues
1. Abuse and crime involving computers
2. Computer-related health issues
3. Ethics
4. Human safety
5. Intellectual property rights
6. Privacy
7. Regulation

8. Transborder data flow
9. Use/abuse of power

2. Social Issues
1. Abuse and crime involving computers
2. Assistive technologies for persons with disabilities
3. Employment
4. Handicapped persons/special needs

3. Organizational Impacts
1. Automation
2. Computer-supported collaborative work
3. Deployment, usage experience
4. Employment
5. Reengineering
6. Scalability, maintainability

4. Electronic Commerce
1. Cybercash, digital cash
2. Distributed commercial transactions
3. Electronic data interchange
4. Intellectual property
5. Payment schemes
6. Security
7. Internet security policies
8. Mobile code security
9. Economic and other policies

13. Miscellaneous

5. Legal Aspects of Computing
0. General

1. Hardware/Software Protection
1. Copyrights
2. Licensing
3. Patents
4. Proprietary rights
5. Trade secrets

2. Governmental Issues
1. Censorship
2. Regulation
3. Taxation

13. Miscellaneous
1. Contracts
2. Hardware patents

6. Management of Computing and Information Systems
0. General
1. Economics
2. Information resource management

1. Project and People Management
1. Life cycle
2. Management techniques
3. Staffing
4. Strategic information systems planning
5. Systems analysis and design
6. Systems development

7. Training
2. Installation Management
1. Benchmarks
2. Computer selection
3. Computing equipment management
4. Performance and usage measurement
5. Pricing and resource allocation
3. Software Management
1. Software development
2. Software maintenance
3. Software process
4. Software selection
4. System Management
1. Centralization/decentralization
2. Management audit
3. Quality assurance
5. Security and Protection
1. Authentication
2. Insurance
3. Invasive software (viruses, worms, Trojan horses)
4. Physical security
5. Unauthorized access (hacking, phreaking)
13. Miscellaneous
1. Insurance
2. Security

7. The Computing Profession
0. General
1. Career Management
1. Occupations
2. Organizations
3. Testing, Certification, and Licensing
4. Professional Ethics
1. Codes of ethics
2. Codes of good practice
3. Ethical dilemmas
13. Miscellaneous
1. Codes of good practice
2. Ethics

8. Personal Computing
0. General
1. Games
1. Application Packages
1. Data communications
2. Database processing
3. Freeware/shareware
4. Graphics
5. Spreadsheets
6. Word processing
2. Hardware
3. Management/Maintenance
13. Miscellaneous

13. Miscellaneous
    1. Business
    2. Education
    3. Financial
    4. Healthcare
    5. Industrial
    6. IT Applications
    7. Legal
    8. Library
    9. Military
    10. Publishing
    11. Sports