Laporan Penelitian

PEMBANGUNAN PERANGKAT LUNAK PENUNJANG PERKULIAHAN TEMU BALIK INFORMASI DI JURUSAN TEKNIK INFORMATIKA

disusun oleh:
Hendra Bunyamin, S.Si., M.T.
Laurentius Risal, S.T.
Daniel Jahja Surjawan, S.Kom.

Januari 2010
Fakultas Teknologi Informasi
Universitas Kristen Maranatha

LEMBAR IDENTITAS

1. Judul Penelitian: Pembangunan Perangkat Lunak Penunjang Perkuliahan Temu Balik Informasi di Jurusan Teknik Informatika.

2. Ketua/Penanggung Jawab Pelaksana Kegiatan Penelitian:

   Nama (lengkap dengan gelar) : Hendra Bunyamin, S.Si., M.T.

   NIK : 720001

   Jabatan Akademik / Golongan : Asisten Ahli / III B

   Fakultas / Jurusan : Universitas Kristen Maranatha

3. Jumlah Tim Peneliti : 3 orang

4. Lokasi Pelaksana Penelitian : Fakultas Teknologi Informasi
   Universitas Kristen Maranatha

5. Lama Pelaksanaan : 6 bulan

6. Sumber Dana Penelitian : Universitas Kristen Maranatha

7. Biaya Penelitian : Rp. 8.000.000,-

Bandung, 31 Januari 2010

Ketua/ Penanggung Jawab Pelaksana

Hendra Bunyamin, S.Si., M.T.

LEMBAR PENGESAHAN

Judul Penelitian : Pembangunan Perangkat Lunak Penunjang Perkuliahan Temu Balik Informasi di Jurusan Teknik Informatika

Peneliti : 1. Hendra Bunyamin, S.Si., MT.
2. Laurentius Risal, S.T.
3. Daniel Jahja Surjawan, S.Kom.

Lokasi Pelaksana Penelitian : Fakultas Teknologi Informasi
Universitas Kristen Maranatha
Jl. Surya Sumantri no. 65
Bandung

Penelitian ini telah diselesaikan pada tanggal 31 Januari 2010 sebagai salah satu perwujudan Tridharma Perguruan Tinggi Universitas Kristen Maranatha

Bandung, 31 Januari 2010

Ketua Peneliti

Hendra Bunyamin, S.Si., M.T.

Dekan Fakultas Teknologi Informasi

Radiant Victor Imbar, S.Kom., M.T.

Ketua LPPM

Ir. Yusak Gunadi Santoso, MM.

ABSTRAK

Information retrieval (IR) *system* adalah sebuah sistem yang digunakan untuk menemukan balik (*retrieve*) informasi-informasi yang relevan terhadap kebutuhan pengguna dari suatu kumpulan informasi secara otomatis. Konsep yang dipelajari di dalam IR system ini akan lebih dapat dipahami oleh mahasiswa apabila terdapat kakas (*tools*) yang dapat membantu mahasiswa di dalam mempelajarinya. Oleh karena itulah, penelitian ini diadakan. Tujuan dari penelitian ini adalah membangun sebuah komponen perangkat lunak penunjang yang dapat membantu pemahaman mahasiswa dalam mempelajari konsep bagaimana IR *system* bekerja. Perangkat lunak penunjang ini diharapkan dapat membantu para dosen dalam mencapai tujuan perkuliahan IR *system* yang akan mulai diadakan di jurusan Teknik Informatika pada semester ganjil 2009/2010. Adapun konsep yang diterapkan dalam komponen IR system ini adalah konsep metode vektor.

Kata kunci : IR *system*, komponen perangkat lunak, metode vektor

ABSTRACT

IR system is a system used to retrieve information which is relevant to users' needs from document collections automatically. The concepts in IR system are more understandable if there are tools which help students in the process of learning them. This research focuses on making the tools. The objective of this research is to develop software components which assist students in understanding the concepts of how an IR system works. Hopefully, this software components also help lecturers to achieve the goals of Introduction to IR system which begins to be taught in semester ganjil 2009/2010. Lastly, the method which is applied in the software components is vector method.
.

Keywords : IR system, software components, vector method

Daftar Isi

Daftar Gambar

I. PENDAHULUAN

Dokumen ini adalah laporan pertanggungjawaban pelaksanaan penelitian Pembangunan Perangkat Lunak Penunjang Konsep Metode Vektor di dalam Perkuliahan Temu Balik Informasi
di Fakultas Teknologi Informasi Universitas Kristen Maranatha yang dilaksanakan pada bulan September 2009 – November 2009.

I.1 Latar Belakang Penelitian

Pada semester reguler ganjil 2009/2010, jurusan Teknik Informatika telah menggunakan kurikulum baru, yaitu kurikulum 2009. Di dalam kurikulum tersebut, terdapat mata kuliah-mata kuliah baru yang sebelumnya tidak ada di dalam kurikulum 2005. Salah satu mata kuliah baru dalam kurikulum 2009 adalah Pengantar Temu Balik Informasi (*Introduction to Information Retrieval*).

Mata kuliah ini bertujuan agar mahasiswa mampu mempelajari konsep dan teori dari sebuah sistem temu balik informasi (*Information Retrieval System*). Mahasiswa akan mempelajari konsep *boolean retrieval*, *term-term vocabulary* dan *posting lists*. Mahasiswa juga akan mempelajari konsep *dictionaries*, dan *tolerant retrieval*. Pembangunan indeks dan kompresi juga akan dipelajari. Dan pada tahap akhir, mahasiswa akan belajar bagaimana mengevaluasi sebuah sistem *information retrieval* (*IR system*).

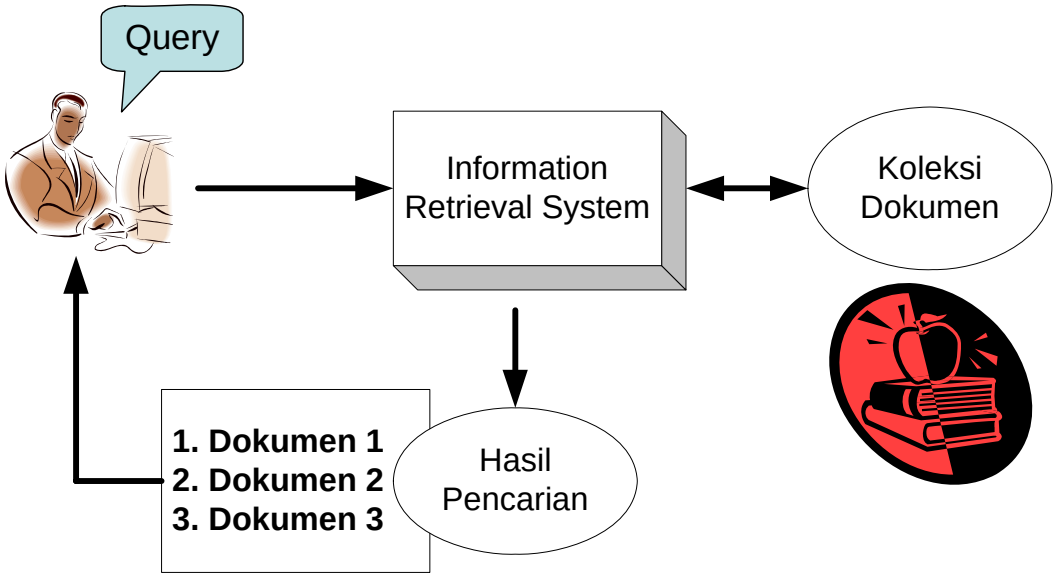
Namun ada kesulitan yang akan dihadapi oleh para mahasiswa dalam perkuliahan ini. Kesulitannya adalah bagaimana menerapkan konsep *IR system* ke dalam bentuk perangkat lunak. Apabila perangkat lunak *IR system* dikembangkan dari nol, mahasiswa akan mengalami kesulitan dan menghabiskan banyak waktu karena mahasiswa mesti mengembangkan banyak komponen pembentuk *IR system*. Ibaratnya, mahasiswa hendak mempelajari bagaimana mobil bekerja. Dan mahasiswa mesti membuat dahulu ban mobil, pelek mobil, kap mesin, kap bagasi, jendela, pintu dan lain sebagainya. Mahasiswa akan menghabiskan banyak waktu untuk membuat komponen-komponen tersebut, padahal konsep yang hendak dipelajari adalah bagaimana mobil bekerja bukan membuat komponen-komponen pembangun mobil. Begitu juga ketika mahasiswa hendak mempelajari *IR system*. Mahasiswa disarankan untuk tidak menghabiskan waktu membuat komponen pembangun *IR system*. Namun mahasiswa diarahkan untuk mempelajari konsep bagaimana *IR system* bekerja dengan menggunakan komponen-komponen yang sudah ada.

Oleh karena itu, untuk membantu pemahaman mahasiswa dalam mempelajari konsep bagaimana *IR system* bekerja diperlukan sebuah perangkat lunak penunjang. Perangkat lunak penunjang ini diharapkan dapat membantu para dosen dalam mencapai tujuan perkuliahan. Dan selanjutnya, perangkat lunak penunjang ini dapat menjadi tugas diskusi diantara mahasiswa untuk dapat mengembangkan perangkat lunak ini lebih lanjut lagi sehingga *software* ini dapat memberikan pemahaman tentang konsep *IR system* yang mudah dicerna bagi para mahasiswa atau pengguna *IR system* di Indonesia.

I.2 Information Retrieval

Information retrieval (IR) *system* adalah sebuah sistem yang digunakan untuk menemukan balik (*retrieve*) informasi-informasi yang relevan terhadap kebutuhan pengguna dari suatu kumpulan informasi secara otomatis.
Gambar 1 menggambarkan sebuah IR *system*. Secara sederhana, langkah-langkah dalam IR *system* adalah

1) Pengguna memasukkan kata kunci (*query*).
2) IR *system* mencari dokumen-dokumen yang memuat *query* tersebut. Dokumen-dokumen yang memuat query tersebut dapat dikatakan sebagai dokumen yang relevan dengan *query* pengguna.
3) IR *system* menampilkan hasil pencarian dalam bentuk *ranking*.

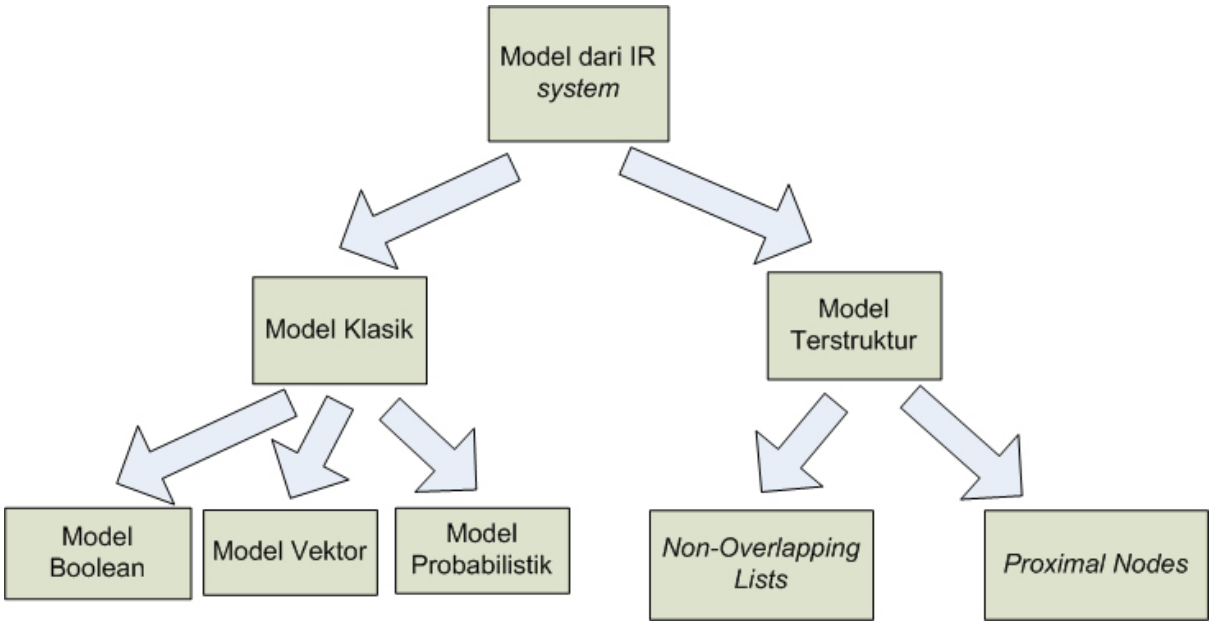


Gambar 1 Ilustrasi dari sebuah IR *system*

Salah satu aplikasi umum dari IR *system* adalah *search engine* atau mesin pencarian yang terdapat pada jaringan Internet. Pengguna dapat mencari halaman-halaman web yang dibutuhkannya melalui *search engine*. Contoh lain dari IR *system* adalah sistem informasi perpustakaan.

Salah satu topik yang menarik mengenai *Information Retrieval* adalah topik mengenai bagaimana memprediksi apakah sebuah dokumen relevan atau tidak relevan. Keputusan untuk menentukan dokumen relevan atau tidak biasanya bergantung pada algoritma *ranking* yang digunakan. Algoritma *ranking* menentukan urutan dari dokumen-dokumen yang relevan. Contohnya, *ranking* pertama menunjukkan dokumen yang paling relevan dengan *query* pengguna. Oleh sebab itu, topik mengenai algoritma *ranking* dapat disimpulkan sebagai inti (*core*) dari sebuah IR *system*.

Terdapat beberapa algoritma *ranking* yang dapat digunakan. Algoritma *ranking* ini bergantung pada model dari IR *system*. Selanjutnya, suatu dokumen relevan atau tidak ditentukan oleh model dari IR *system* yang digunakan.



Gambar 2 Model-model dari IR *system*

Taksonomi dari model sebuah IR *system* adalah seperti pada Gambar 2. Dalam penelitian ini, model yang hendak dikembangkan adalah model vektor. Penelitian ini berfokus untuk membangun perangkat lunak yang menunjang materi metode vektor di dalam perkuliahan pengantar temu balik informasi.

Harapannya, mahasiswa dengan menggunakan perangkat lunak ini dapat lebih mudah di dalam memahami materi metode vektor.

I.3 Tujuan dan Manfaat Penelitian

Adapun manfaat dan tujuan yang dapat dirasakan dengan adanya ”Pembangunan Perangkat Lunak Penunjang Perkuliahan Temu Balik Informasi di Jurusan Teknik Informatika” ini adalah:

A. Untuk Mahasiswa

Memungkinkan untuk mendiskusikan bagaimana cara pembuatan sebuah perangkat lunak *information retrieval* dengan menggunakan bahasa pemrograman JAVA serta mengembangkannya dengan menambah fitur-fitur yang diperlukan.

B. Untuk Dosen

Memungkinkan untuk membantu mahasiswa supaya mahasiswa dapat lebih mengerti tentang *information retrieval* yaitu cara kerja sebuah perangkat lunak temu balik informasi, untuk kemudian didiskusikan. Kemudian dari hasil diskusi dapat diperoleh ide untuk mengembangkan perangkat lunak lain secara bersama-sama.

Dengan melihat banyaknya manfaat yang dapat dirasakan oleh mahasiswa dan dosen maka penelitian untuk “Pembangunan Perangkat Lunak Penunjang Perkuliahan Temu Balik Informasi di Jurusan Teknik Informatika” perlu dilaksanakan.

I.3 Batasan Penelitian

Adapun batasan – batasan yang dipakai untuk penelitian ini adalah sebagai berikut:

- Koleksi dokumen yang digunakan oleh perangkat lunak ini adalah koleksi dokumen ADI (dokumen-dokumen mengenai *information science*) [Baeza-Yates and Ribeiro-Neto, 1999].
- Bahasa yang digunakan dalam koleksi dokumen adalah bahasa Inggris.

I.4 Metoda Penelitian

Metode pembangunan perangkat lunak ini dilakukan dengan menggunakan *traditional waterfall model* [Bennet, McRobb, and Farmer, 2002]. *Traditional waterfall model* membagi pekerjaan pembangunan perangkat lunak menjadi lima tahapan, yaitu

1) *System Engineering*
2) *Requirement Analysis*
3) *Implementation*
4) *Testing*
5) *Installation*
6) *Maintenance*

Tahapan-tahapan ini akan dijelaskan di dalam bab subbab-subbab berikutnya.

I.5 Perangkat Keras dan Perangkat Lunak yang dipakai

Untuk menunjang penelitian ini diperlukan perangkat keras dan perangkat lunak penunjang yaitu:

1. Perangkat keras: Komputer Desktop / Notebook
2. Perangkat lunak:
    - **Sistem Operasi: Windows XP**. Versi Windows XP yang digunakan dalam penelitian ini adalah Windows XP dengan Service Pack 2.
    - **Java 2 Runtime Environment**. Versi J2RE yang digunakan dalam penelitian ini adalah J2RE versi 1.6.0.
    - **Netbeans IDE**: perangkat lunak yang digunakan untuk membangun komponen IR System metode vektor. Versi Netbeans IDE yang digunakan dalam penelitian ini adalah Netbeans versi 6.7.
    - **Microsoft Visio**: perangkat lunak yang digunakan untuk membuat desain perangkat lunak dari software EuroBudget.

II. LANDASAN TEORI

Di dalam mempelajari Information Retrieval, terdapat beberapa konsep yang perlu dipahami. Beberapa konsep tersebut adalah konsep model vektor (model yang dikembangkan di dalam penelitian ini) dan bagaimana kita mengevaluasi sebuah IR system.

II.1 Model Vektor

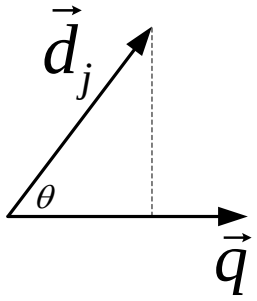
Model vektor menggunakan konsep vektor untuk menghitung relevansi antara query dengan koleksi dokumen. Prinsip utamanya adalah query diubah menjadi vektor query dan dokumen-dokumen di dalam koleksi dokumen diubah menjadi vektor-vektor dokumen ($\vec{d_j}$, $1 \leq j \leq n$) [Salton et al., 1975].

Misalkan banyaknya dokumen di dalam koleksi dokumen adalah n. Kemudian vektor query adalah $\vec{q}$ dan masing-masing vektor dokumen adalah $\vec{d_j}$ dengan $j = 1, 2, ..., n$. Nilai relevansi (*similarity*) antara query dengan dokumen ke-j [Baeza-Yates and Ribeiro-Neto, 1999] adalah

$$sim(\vec{d_j}, \vec{q}) = \frac{\vec{d_j} \cdot \vec{q}}{|\vec{d_j}| \times |\vec{q}|}$$

…...……(1)

Menurut konsep aljabar linier, nilai $sim(\vec{d_j}, \vec{q})$ adalah $\cos\theta$, seperti pada Gambar 3.



Gambar 3 Besar sudut antara vektor query dan vektor dokumen

Nilai relevansi terbesar antara dokumen ke-j dan query adalah ketika nilai $\cos\theta$ sama dengan 1 atau nilai $\theta = 0°$.

Contoh:

Terdapat tiga buah dokumen, yaitu D_1, D_2, D_3 dan sebuah *query* Q. Misalkan tiga buah dokumen tersebut dan *query* dikonversi menjadi vektor-vektor sebagai berikut,

D_1 = <1, 1>
D_2 = <1, 0>
D_3 = <0, 1>
Q = <1, 1>

Kemudian marilah kita menghitung nilai-nilai *similarity* antara Q dan D_1, Q dan D_2, kemudian Q dan D_3.

$sim(D_1 1, Q) = (< 1,1 > \cdot < 1,1 >)/(| < 1,1 > | \times | < 1,1 > |) = 2/(\sqrt{2} \times \sqrt{2}) = 1 \square (\Leftrightarrow -)\ \cos \square \theta = 1 \square (\Leftarrow$

$sim(D_1 2, Q) = (< 1,0 > \cdot < 1,1 >)/(| < 1,0 > | \times | < 1,1 > |) = 1/\sqrt{2} = 1/2\ \sqrt{2} \square (\Leftrightarrow -)\ \cos \square \theta = 1/2\ \sqrt{}$

$sim(D_1 3, Q) = (< 0,1 > \cdot < 1,1 >)/(| < 0,1 > | \times | < 1,1 > |) = 1/\sqrt{2} = 1/2\ \sqrt{2} \square (\Leftrightarrow -)\ \cos \square \theta = 1/2\ \sqrt{}$

Dari ketiga perhitungan *similarity* di atas, diperoleh bahwa D_1 merupakan dokumen yang paling relevan dengan Q.

Kemudian hal penting yang perlu kita pelajari adalah konsep pembobotan untuk membangun komponen-komponen dari vektor-vektor tersebut. Untuk membangun sebuah vektor yang merupakan representasi dari sebuah dokumen, pandanglah beberapa definisi berikut:

tf_{ij} = banyaknya kemunculan term t_j di dalam dokumen D_i.

tf_{ij} disebut sebagai *term frequency*……..(2)

df_j = banyaknya dokumen yang memuat term t_j.

df_j disebut sebagai *document frequency*……...(3)

$idf_j = \log \frac{d}{df_j}$ dengan d adalah banyaknya dokumen keseluruhan…….…….(4)

idf_j disebut sebagai *inverse document frequency.*

Faktor pembobotan untuk sebuah *term* di dalam sebuah dokumen adalah kombinasi dari *term frequency* dan *inverse document frequency*. Contohnya, marilah kita menghitung nilai ke-j di dalam sebuah vektor yang merupakan representasi dari dokumen ke-i . Rumus berikut ini yang akan digunakan

$$d_{ij} = tf_{ij} \times idf_j \quad \text{.....(5)}$$

Marilah kita aplikasikan konsep dan rumus di atas dengan menyelesaikan soal berikut ini.

Diketahui sebuah query dan tiga dokumen sebagai berikut:

Q : ”gold silver truck”

D_1: ”Shipment of gold damaged in a fire”

D_2: ”Delivery of silver arrived in a silver truck”

D_3: ”Shipment of gold arrived in a truck”

Dalam koleksi ini, terdapat tiga dokumen sehingga diperoleh d = 3.
Inverse document frequency dapat dihitung dan hasilnya seperti pada Tabel 1.

No.	Frekuensi kemunculan term	Nilai *idf*

1.	1	$\log\frac{d}{df_j} = \log\frac{3}{1} = 0.477$
2.	2	$\log\frac{d}{df_j} = \log\frac{3}{2} = 0.176$
3.	3	$\log\frac{d}{df_j} = \log\frac{3}{3} = 0$

Tabel 1 Nilai *idf* untuk setiap frekuensi kemunculan

Kemudian Table 2 menunjukkan menunjukkan nilai *idf* untuk masing-masing term

idf_a = 0
$idf_{arrived}$ = 0.176
$idf_{damaged}$ = 0.477
$idf_{delivery}$ = 0.477
idf_{fire} = 0.477
idf_{gold} = 0.176
idf_{in} = 0
idf_{of} = 0
idf_{silver} = 0.477
$idf_{shipment}$ = 0.176
idf_{truck} = 0.176

Setelah semua nilai *idf* diketahui, kita dapat membangun vektor dokumen-vektor dokumen. Bobot untuk *term i* pada vektor *j* dihitung sebagai $idf_i \times tf_{ij}$. Tabel 2 menunjukkan vektor dokumen-vektor dokumen dari ketiga dokumen di atas. Vektor dokumen-vektor dokumen dari dokumen ke-1, ke-2 dan ke-3 membentuk sebuah matriks yang disebut matriks *terms-documents*. Matriks ini memiliki ukuran 3 (banyak dokumen) x 11 (banyak term).

docid	**a**	**arrived**	**damaged**	**delivery**	**fire**	**gold**	**In**	**of**	**shipment**	**silver**	**truck**
D_1	0	0	0.477	0	0.477	0.176	0	0	0.176	0	0
D_2	0	0.176	0	0.477	0	0	0	0	0	0.954	0.176
D_3	0	0.176	0	0	0	0.176	0	0	0.176	0	0.176
Q	0	0	0	0	0	0.176	0	0	0	0.477	0.176

Tabel 2 Vektor Dokumen-Vektor Dokumen dari Tiga Dokumen

Setelah kita mempunyai vektor-vektor di atas, kita dapat menghitung *similarity* antara Q dan D1, Q dan D2, dan Q dan D3.

$$sim(Q, D_1) = (0)(0) + (0)(0) + (0)(0.477) + (0)(0) + (0)(0.477) + (0.176)(0.176) + (0)(0) + (0)(0) + (0)(0.176) + (0.477)(0) + (0.176)(0) = (0.176)^2 \approx 0.031$$

Dengan cara yang sama, diperoleh

$$sim(Q, D_2) = (0.954)(0.477) + (0.176)^2 \approx 0.486$$

$$sim(Q, D_3) = (0.176)^2 + (0.176)^2 \approx 0.062$$

Dari ketiga hasil perhitungan di atas, urutan dokumen dimulai dari dokumen yang paling relevan adalah D_2, D_3, D_1.

II.2 Koleksi Dokumen

Koleksi dokumen yang digunakan sebagai uji coba dalam penelitian ini adalah koleksi dokumen ADI.

Koleksi dokumen ADI terdiri 82 (delapan puluh dua) dokumen. Sebagai contoh, cuplikan dokumen ke-9 dari koleksi dokumen ADI adalah

```
.I 9
.T
analysis of the role of the computer in the reproduction
and distribution of scientific papers
.A
J. H. KUNEY
.W
the american chemical society has begun
 an analysis of the role of the computer in related aspects
 of the reproduction, distribution, and retrieval of scientific
 information .  initial work will attempt to solve problems
 of photocomposition via computer .
```

Keterangan:

- .I menunjukkan nomor dokumen.
- .T menunjukkan judul dokumen.
- .A menunjukkan pengarang dokumen.
- .W menunjukkan isi dokumen.

III. ANALISIS DAN DESAIN PERANGKAT LUNAK

Analisis dan desain perangkat lunak IR *System* dengan metode vektor dikembangkan dengan menggunakan kakas bantu yaitu bahasa pemodelan UML (*Unified Modeling Language*). Diagram-diagram yang digunakan dalam pemodelan IR *System* adalah

1. *Use case diagram*
2. *Class diagram*

Dalam subbab-subbab berikutnya kita akan membahas setiap diagram yang digunakan untuk mendesain IR *System*.

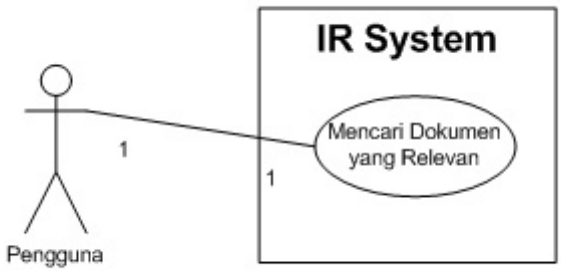
III.1 Diagram Use Case

Requirement yang diperoleh dari IR system [Bennet, McRobb, and Farmer, 2002] adalah

Requirement A

IR System mesti mencari dokumen-dokumen yang relevan dengan *query*.

Requirement A di atas dapat digambarkan menjadi sebuah use case diagram seperti pada Gambar 4.



Gambar 4 Diagram Use Case mengenai IR System

Diagram *use case* di atas menggambarkan hanya satu *use case*, yaitu Mencari Dokumen yang Relevan namun terdapat beberapa tahapan di dalam satu *use case* tersebut yang akan dijelaskan di dalam *use case specification* [Chonoles and Schardt, 2003] di bawah ini.

Use-case name: Mencari Dokumen yang Relevan

Description: pengguna mengetikkan *query* untuk memperoleh dokumen-dokumen yang relevan dengan *query* tersebut.

Main course of event: Dokumen-dokumen yang relevan berhasil ditampilkan.

Precondition: Pengguna sudah berada di tampilan *command prompt*.

Successful postcondition: Pengguna dapat melihat dokumen-dokumen yang relevan

Pengguna	**IR System**
1. *Use case* dimulai ketika pengguna memasukkan *query*	2. Sistem membaca file properties dan menginisialisasi semua variables.
	3. Sistem membangun objek koleksi dokumen.
	4. Sistem membangun objek *query*.
	5. Sistem membangun objek *DocumentRanker*.
	6. Objek *DocumentRanker* mengolah *query* dan koleksi dokumen dan hasil pengolah disimpan di objek *RankedDocuments*.
	7. Sistem menampilkan dokumen-dokumen relevan yang tersimpan di objek *RankedDocuments*.

Dari *use case specification* di atas, ada beberapa objek yang membangun IR System. Beberapa objek tersebut berperan di dalam proses pencarian dokumen-dokumen yang relevan dengan query, yaitu

1. **File Properties** (nomor 2 dari use case specification) adalah File yang berisi setting dari IR System.
2. **Koleksi dokumen** (nomor 3 dari use case specification) adalah kumpulan dokumen-dokumen yang digunakan sebagai ujicoba.
3. **Query** (nomor 4 dari use case specification) adalah query dari pengguna yang digunakan sebagai input di dalam proses pencarian dokumen.
4. **DocumentRanker** (nomor 5 dari use case specification) adalah objek yang berperan untuk mengurutkan dokumen-dokumen yang relevan.
5. **RankedDocuments** (nomor 6 dari use case specification) adalah objek yang berperan untuk menyimpan hasil ranking dari proses yang dilakukan oleh DocumentRanker.

III.2 Diagram Class

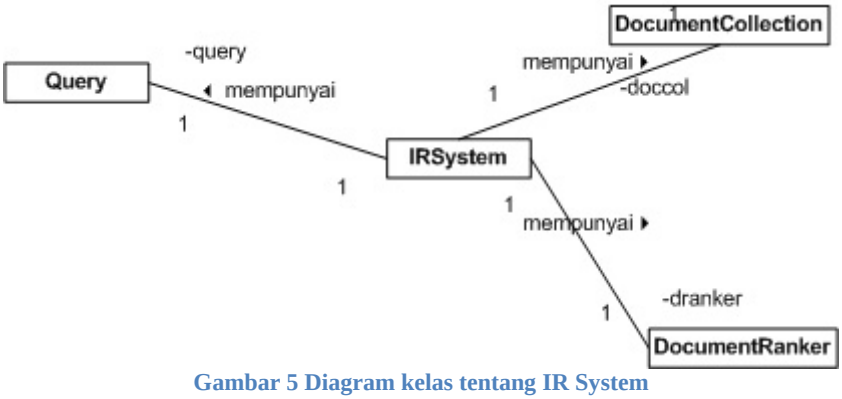
Marilah kita tinjau kembali proses yang terjadi di dalam IR System.

”Pengguna memasukkan query ke dalam IR System. Lalu IR System membangun objek dari koleksi dokumen.
Koleksi dokumen yang berbentuk teks di-*parse* oleh sebuah parser. Selanjutnya, hasil dari parser dikirim ke sebuah stemmer. Stemmer berfungsi untuk membuang imbuhan dari sebuah kata sehingga kita memperoleh sebuah kata dasar. Prosesnya ini disebut *stemming*. Kemudian, hasil dari proses stemming disimpan di koleksi dokumen.
Selanjutnya, query dan koleksi dokumen menjadi input untuk proses mencari dokumen-dokumen yang relevan. Hasil dokumen-dokumen yang relevan tersebut diurutkan oleh sebuah pengurut dokumen. Terakhir, hasil pengurutan tersebut disimpan di dalam sebuah vektor.”

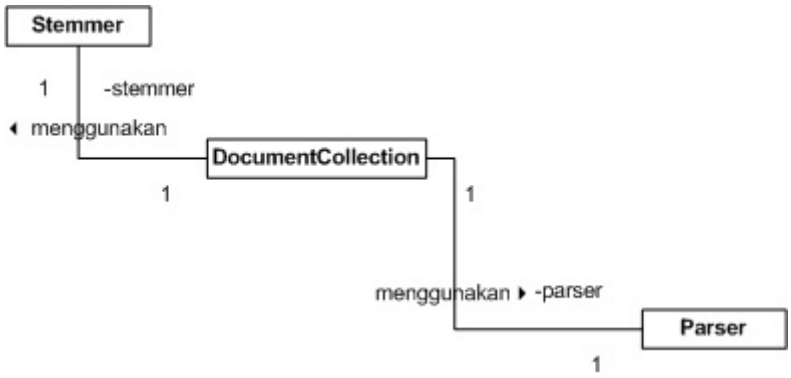
Berikut adalah kelas-kelas yang diperoleh dari proses yang terjadi di atas:

1. IRSystem dari IR System.
2. Query dari query.
3. DocumentCollection dari koleksi dokumen.
4. Parser dari parser.
5. Stemmer dari stemmer.
6. DocumentRanker dari pengurut documen

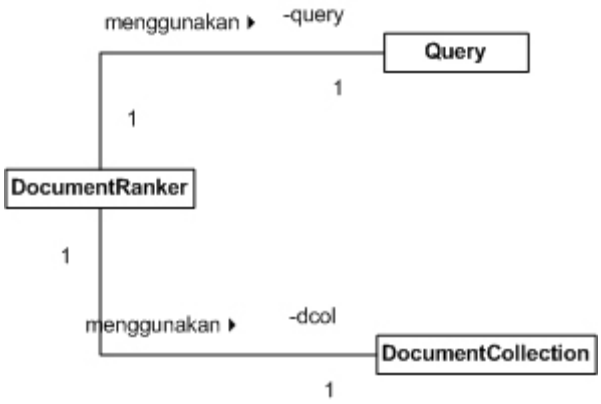
Selanjutnya adalah pembuatan asosiasi dari kelas-kelas tersebut. Hasilnya adalah diagram kelas



Gambar 5 Diagram kelas tentang IR System



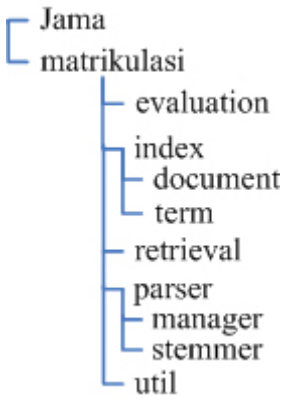
Gambar 6 Diagram kelas tentang Document Collection



Gambar 7 Diagram kelas tentang DocumentRanker

IV. IMPLEMENTASI DARI IR SYSTEM

Berdasarkan diagram kelas-diagram kelas pada bab sebelumnya, IR *system* dibagi menjadi beberapa modul atau *package*.
Package utama dari IR system adalah matrikulasi [Bunyamin, 2005].



Gambar 8 Struktur Package dari komponen IR System

Jama merupakan singkatan dari *Java Matrix*®, berisi kelas-kelas yang berfungsi untuk melakukan operasi matriks seperti membuat struktur data matriks dan mendekomposisi suatu matriks (*singular value decomposition*) [Bunyamin, 2005].

Matrikulasi merupakan *core* dari program Matrikulasi [Bunyamin, 2005]. *Package* matrikulasi dibagi menjadi 5 (lima) buah *sub-package*, yaitu evaluation, index, parser, retrieval, dan util. Penjelasan mengenai masing-masing *sub-package* adalah

- evaluation berisi kelas-kelas yang berfungsi menghitung performansi hasil ranking;
- index berisi kelas-kelas yang berfungsi melakukan proses indexing terhadap koleksi dokumen;
- parser berisi kelas-kelas yang berfungsi mengerjakan stemming terhadap hasil indexing;

- retrieval berisi kelas-kelas yang berfungsi melakukan komputasi untuk memperoleh nilai relevansi query dengan dokumen.
- util berisi kelas-kelas yang berfungsi sebagai utility dalam mengerjakan proses-proses.

Dalam penelitian ini, komponen perangkat lunak IR System metode vektor dikembangkan pada package index dan retrieval.

IV.1 Diagram Class

Implementasi program dari diagram kelas pada Gambar 5, 6 dan 7 adalah sebagai berikut:

1. Kelas-kelas pada Gambar 5.

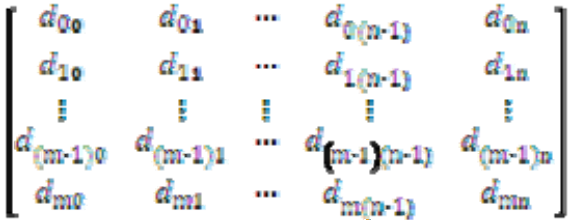
1. Kelas Query tetap menjadi kelas Query (Query.java). Kelas ini berada di dalam package matrikulasi.retrieval [Bunyamin, 2005]. Kelas Query memiliki method utama, yaitu:

   `setString(String, DocumentCollection, DocumentRanker)`. Method ini berfungsi untuk menerima query dalam bentuk String dan mengubah bentuk String tersebut menjadi bentuk matriks. Matriks ini mempunyai ukuran n × 1, dengan n adalah banyaknya term di dalam koleksi dokumen.

2. Kelas IRSystem berubah namanya menjadi kelas VectorMethodTestDrive (VectorMethodTestDrive.java). Hal ini terjadi karena tujuan dari penelitian ini adalah pembuatan komponen IR system dan bukan IR system yang lengkap. Kelas VectorMethodTestDrive hanya memiliki sebuah method, `main(String[])`. Method ini berfungsi untuk menjalankan langkah-langkah di dalam use case Mencari Dokumen Relevan.

3. Kelas DocumentCollection tetap menjadi kelas DocumentCollection (DocumentCollection.java). Kelas ini berada di dalam package matrikulasi.index [Bunyamin, 2005]. Kelas DocumentCollection merupakan representasi dari koleksi dokumen di real world. Method-method utama yang dimilikinya adalah
   a. `getAllDocs(int)`. Method ini berfungsi untuk mengembalikan semua dokumen yang memiliki identitas term tertentu (bertipe integer).
   b. `getAllTerms(int)`. Method ini berfungsi untuk mengembalikan semua term yang memiliki identitas dokumen tertentu (bertipe integer).
   c. `getCollectionCount()`. Method ini berfungsi untuk mengembalikan banyaknya dokumen yang berada di dalam koleksi dokumen.
   d. `getDocFreq(String)`. Method ini berfungsi untuk mengembalikan banyaknya dokumen yang memiliki term tertentu (bertipe String).
   e. `getTermFreq(String, int)`. Method ini berfungsi untuk mengembalikan frekuensi kemunculan term tertentu (bertipe String) di dalam dokumen tertentu (bertipe integer).
4. Kelas DocumentRanker tetap menjadi kelas DocumentRanker (DocumentRanker.java). Kelas ini berada di dalam package matrikulasi.retrieval [Bunyamin, 2005]. Kelas DocumentRanker adalah kelas yang berperan dalam membangun matriks *terms-documents*. Matriks *terms-documents* ini memiliki ukuran

banyak dokumen (m) × banyak *terms* (n). Elemen-elemen dari matriks *terms-documents* adalah hasil perhitungan dari rumus (5). Ilustrasi di bawah ini menjelaskan elemen-elemen dari matriks terms-documents.



2. Kelas-kelas pada Gambar 6.

1. Kelas Stemmer tetap menjadi kelas PorterStemmer (PorterStemmer.java). Kelas ini berada di dalam package matrikulasi.parser.stemmer [Bunyamin, 2005]. Kelas PorterStemmer berfungsi untuk membuang awalan dan akhiran dari sebuah kata. Contohnya, kata ’technical’ dimasukkan ke dalam Stemmer dan outputnya adalah ’techic’. Kemudian, kata ’tradition’ menjadi ’tradit’. Method utama yang dimiliki olehnya adalah

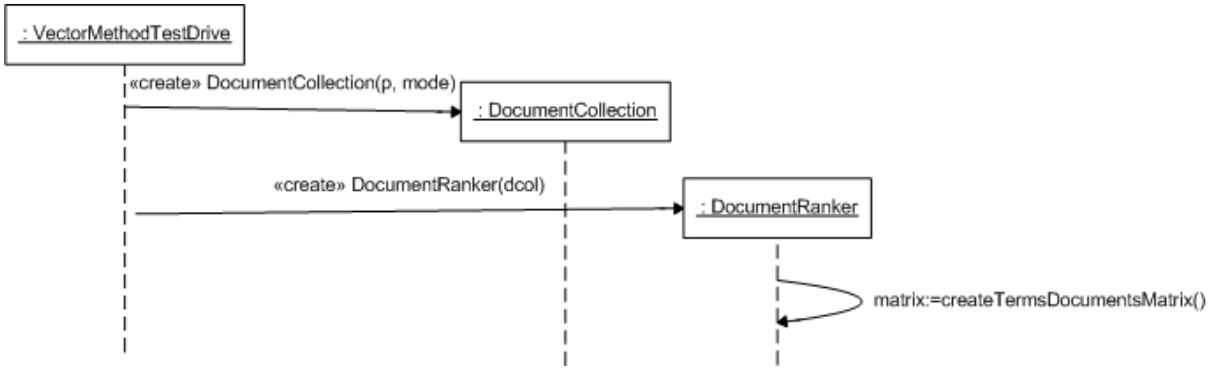
   `stem(String)`. Method ini berfungsi untuk menerima sebuah kata (bertipe String) dan memprosesnya menjadi kata dasar (bertipe String).

2. Kelas Parser tetap menjadi kelas Parser (Parser.java). Kelas ini berada di dalam package matrikulasi.parser [Bunyamin, 2005]. Kelas Parser berfungsi untuk mem-parse koleksi dokumen menjadi term-term. Selanjutnya, term-term tersebut disimpan di dalam objek BufferedRandomAccessFile. Method utama yang dimiliki kelas Parser adalah

   `parseIt()`. Method ini berfungsi untuk memanggil method parse dari kelas ParseManager.

IV.2 Diagram Sequence

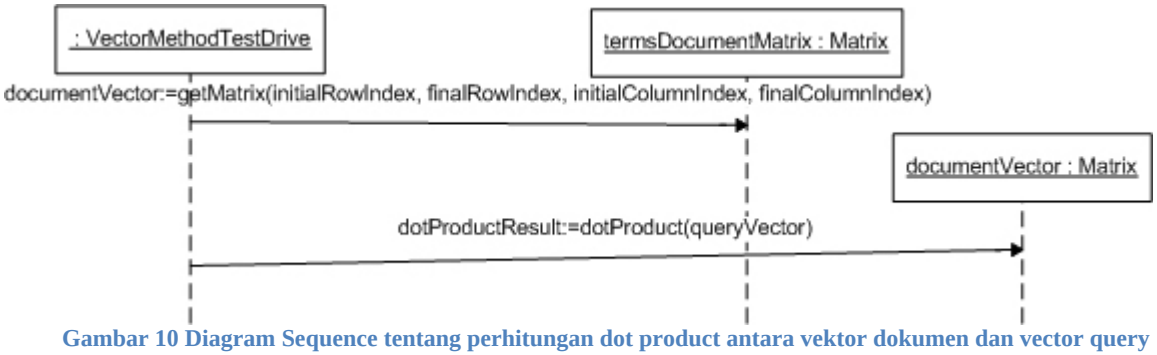
Gambar 9 menjelaskan pembuatan matriks *terms-documents*. Matriks *terms-documents* adalah matriks yang elemen-elemennya merupakan vektor-vektor dokumen dari koleksi dokumen.



Gambar 9 Diagram Sequence tentang Pembuatan Matriks Terms-Documents

Method `createTermsDocumentsMatrix` dari Gambar 9 menghasilkan sebuah matriks. Apabila matriks tersebut dikenakan operasi transpose, matriks ini akan menjadi matriks *terms-documents* [Bunyamin, 2005].

Langkah selanjutnya adalah membuat submatriks-submatriks dari matriks *terms-documents*, yaitu baris-baris dari matriks *terms-documents*. Contohnya, baris pertama dari matriks terms-documents menjadi submatriks yang disebut vektor dokumen pertama. Vektor dokumen pertama adalah vektor yang merupakan representasi dari dokumen pertama. Demikian juga dengan baris kedua yang merupakan vektor dokumen kedua, dan seterusnya.



Gambar 10 Diagram Sequence tentang perhitungan dot product antara vektor dokumen dan vector query

Gambar 10 menjelaskan kelas-kelas dan method-method yang terlibat di dalam perhitungan hasil kali titik (*dot product*) antara vektor dokumen dan vektor *query*. Contohnya, untuk menghitung nilai *dot product* antara vektor dokumen pertama dan vektor *query*, *initialRowIndex* dan *finalRowIndex* diberi nilai 1, kemudian *initialColumnIndex* diberi nilai 1 dan *finalColumnIndex* diberi nilai 82. Cara menghitung hasil kali titik antara dua vektor menggunakan rumus (1).

V. PENGUJIAN KOMPONEN IR SYSTEM

Komponen IR System diujikan untuk koleksi dokumen ADI yang memuat 82 dokumen. Pengujian dilakukan dengan membandingkan hasil perhitungan komputer dengan hasil perhitungan manual.

Contoh query yang diinputkan ke dalam IR System adalah
"*What problems and concerns are there in making up descriptive titles?*

What difficulties are involved in automatically retrieving articles from

approximate titles?

What is the usual relevance of the content of articles to their titles?"

Berikut adalah dua contoh perhitungan manual adalah

1. Term *retriev* adalah term nomor 11 dengan frekuensi kemunculan 1 di *query* ($tf_{q(11)}$). Frekuensi dokumen yang memuat term retriev (df_{11}) adalah 29. Hasil perhitungan $d_{q(11)}$ dengan IR System adalah 0.3010299956639812. Sedangkan, perhitungan manual adalah

$$d_{q(11)} = tf_{q(11)} \times idf_{11} = tf_{q(11)} \times \log\frac{d}{df_{11}} = 1 \times \log\frac{82}{29} = 0.3010299956639812$$

   Hasil perhitungan IR System dan manual menunjukkan angka yang sama.

2. Term *technic* adalah term nomor 2 dengan frekuensi kemunculan 3 di dokumen 1 (tf_{12}). Frekuensi dokumen yang memuat term *technic* (df_2) adalah 9. Hasil perhitungan d_{12} dengan IR System adalah 2.8627275283179747. Sedangkan, perhitungan manual adalah

$$d_{12} = tf_{12} \times idf_2 = tf_{12} \times \log\frac{d}{df_2} = 3 \times \log\frac{82}{29} = 2.8627275283179747$$

   Hasil perhitungan IR System dan manual menunjukkan angka yang sama.

Proses *parsing* dan *stemming* dilakukan terhadap query dan koleksi dokumen ADI. Hasilnya adalah

1. Matriks terms-documents terbentuk.
   Contoh outputnya adalah tampilan semua term dari koleksi dokumen ADI.

```
0-->process  1-->technic  2-->total  3-->catalog  4-->featur  5-->output
6-->format  7-->integr  8-->drawn
                          ⋮
891-->proposit  892-->calculu  893-->linguist  894-->predic
```

2. Query di-*parsing* dan di-*stemming*.
   Contoh outputnya adalah

```
Data ke-10 --> TERM = retriev FREQUENCY = 1.0
Data ke-73 --> TERM = automat FREQUENCY = 1.0
Data ke-129 --> TERM = make FREQUENCY = 1.0
Data ke-196 --> TERM = problem FREQUENCY = 1.0
Data ke-269 --> TERM = relev FREQUENCY = 1.0
Data ke-327 --> TERM = titl FREQUENCY = 3.0
Data ke-366 --> TERM = content FREQUENCY = 1.0
Data ke-423 --> TERM = articl FREQUENCY = 2.0
Data ke-466 --> TERM = involv FREQUENCY = 1.0
Data ke-690 --> TERM = descript FREQUENCY = 1.0
Data ke-695 --> TERM = approxim FREQUENCY = 1.0
```

3. Hasil kali produk antara vektor dokumen dan vektor *query* dihitung untuk dokumen 1 sampai dengan dokumen 82. Hasilnya adalah

```
 No Document   Nilai Hasil Kali Titik
      69              0.237431
      46              0.125754
      17              0.093781
      25              0.072638
      47              0.072096
      75              0.069567
      27              0.067214
      71              0.066713
      56              0.066137
      57               0.05667
```

30	0.045032
19	0.042905
32	0.040956
4	0.039697
61	0.038179
50	0.035419
70	0.035277
14	0.03416
79	0.033105
2	0.030562
23	0.029277
41	0.021117
9	0.021094
58	0.020815
51	0.020295
7	0.019242
66	0.016255
18	0.013114
16	0.011321
15	0.010795
64	0.005708
39	0.004414
22	0.003691
12	0.003691
55	0.00307
62	0.00265
21	0.001666
43	0.001629
6	0.00153
26	0.00149
38	0.001461
73	0.001334
78	0.001274
11	0.001179
36	9.81E-04
1	8.61E-04

Dari hasil diperoleh bahwa dokumen yang relevan dengan *query* dengan menggunakan metode vektor adalah dokumen nomor 69.

VI. KESIMPULAN DAN SARAN

VI.1 Kesimpulan

Kesimpulan yang dapat diperoleh dari penelitian Pembangunan Perangkat Lunak Penunjang Perkuliahan Temu Balik Informasi di Jurusan Teknik Informatika adalah:

1. Komponen IR *System* ini dapat digunakan sebagai demonstrasi konsep metode vektor untuk mata kuliah Pengantar Temu Balik Informasi.
2. Komponen IR *System* ini dapat digunakan untuk membantu mahasiswa memahami konsep sebuah IR *System* bekerja. Mencari *query* yang sesuai di dalam koleksi dokumen langkah demi langkahnya dapat dipelajari sehingga memudahkan mahasiswa untuk mengerti.
3. Komponen IR *System* ini dikembangkan dengan menggunakan bahasa pemrograman JAVA sehingga mahasiswa dapat mempelajari bahwa bahasa pemrograman JAVA dapat diaplikasikan dalam sebuah konsep IR *System*.

VI.2 Saran

Saran yang dapat diberikan dari penelitian Pembangunan Perangkat Lunak Penunjang Perkuliahan Temu Balik Informasi di Jurusan Teknik Informatika adalah:

1. Koleksi dokumen yang digunakan di dalam demonstrasi IR *System* adalah koleksi dokumen ADI. Demonstrasi IR *System* akan lebih menarik apabila koleksi dokumennya dapat ditambah.
2. Bahasa yang digunakan di dalam demonstrasi IR *System* adalah bahasa Inggris. IR System akan lebih bermanfaat bagi masyarakat Indonesia apabila *query* dan koleksi dokumennya adalah bahasa Indonesia.
3. Demonstrasi IR *System* ini lebih menfokuskan kepada komponen yang dapat digunakan. IR *System* akan lebih membantu pengguna apabila IR *System* juga memiliki tampilan *user interface design*.

Daftar Pustaka

[Baeza-Yates and Ribeiro-Neto, 1999] Baeza-Yates, R. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval.* Addison Wesley.

[Bennet, McRobb, and Farmer, 2002] Bennet, S., McRobb, S., and Farmer, R. (2002). *Object-Oriented Systems Analysis and Design using UML*. McGraw-Hill.

[Bunyamin, 2005] Bunyamin, H. (2005). *Information Retrieval System dengan Metode Latent Semantic Indexing*. Tesis Magister Teknik ITB.

[Chonoles and Schardt, 2003] Chonoles, M.J., and Schardt, J.A. (2003). *UML 2 for Dummies*. Wiley Publishing, Inc.

[Grossman and Frieder, 2004] Grossman, D. and Frieder, O. (2004). *Information Retrieval: Algorithms and Heuristics*. Springer.

[Lethbridge and Laganiere, 2002] Lethbridge, T.C., and Laganiere, R. (2002). *Object-Oriented Software Engineering*. McGraw-Hill.

[Rijsbergen, 1979] Rijsbergen, C.J. van (1979). *Information Retrieval*, Butterworths, London.

[Salton et al., 1975] Salton, G., Yang, C.S., and Wong, A. (1975). *A vector-space model for automatic indexing*. Communications of the ACM.