Studia Informatika: Jurnal Sistem Informasi, 2(1), 2009, 15-25

IMPLEMENTASI *EXTENDED BOOLEAN* DAN PEMANFAATAN TESAURUS PADA TEMU KEMBALI INFORMASI

Nia Kumaladewi[a] dan Victor Amrizal[b]

[a] *Staf Pengajar Fakultas Sains dan Teknologi*
Universitas Islam Negeri Syarif Hidayatullah Jakarta
Tel : (021) 7493547 Fax : (021) 7493315
e-mail : nia_april12@yahoo.com

[b]*Staf Pengajar Fakultas Sains dan Teknologi*
Universitas Islam Negeri Syarif Hidayatullah Jakarta
Tel : (021) 7493547 Fax : (021) 7493315
e-mail : ersebros@yahoo.com

ABSTRACT

Method that is used in meeting system is back information it is methodic Extended Boolean who merge query Boolean's formula and vector room model. If Boolean's method utilize AND'S notation, OR, NOT but that Abbreviate can't resulting document, and vector room method that Abbreviate can resulting document but not utilize Boolean's notation, therefore method extended Boolean constitutes affiliate both. Utilizing thesaurus in do query's extension so one document that relation can with other document. To know operator performance and AND on extended Boolean, doing that test-driving more operator a lot of get relevant document to be compared with AND'S operator, but with operator also irrelevant document amount most prodigious takings. Meanwhile meeting performance is back information on thesaurus which is more get relevant document on query that specific than query what does common. Query that more effective being utilized on extended Boolean's method which is query long (more than 2 syllables) to get document that aptly relevant. Meanwhile on thesaurus, query is short (less than 3 syllables) more effective to be utilized.

Keywords**:** *Extended Boolean, Information, Precision, Recall Tesaurus and Cluster*.

1. PENDAHULUAN

Sudah sejak lama, orang-orang melakukan komunikasi dengan menggunakan teks. Semakin lama semakin banyak teks yang ditulis maka jumlah data yang dihasilkan pun semakin meningkat. Untuk memilih atau menemukan kembali informasi diperlukan suatu pencarian dokumen yang cepat dan akurat, dimana hal ini sesuai dengan keinginan user. Salah satu bidang yang menerapkan hal tersebut adalah sistem temu kembali informasi. Di dalam sistem temu kembali informasi ini, tidak hanya informasi yang cepat dan akurat yang dibutuhkan tetapi juga informasi yang relevan bagi penggunanya. Salah satu penerapan sistem temu kembali informasi adalah mesin pencari atau *search engine*. Metode yang dipakai dalam sistem temu kembali informasi ini adalah metode *Extended Boolean* yang menggabungkan formula *query Boolean* dan model ruang vektor. Bila metode *Booelan* menggunakan notasi AND, OR, NOT tetapi tidak dapat memperingkat dokumen yang dihasilkan, dan metode ruang vektor bisa memperingkat dokumen yang dihasilkan tetapi tidak menggunakan notasi *Boolean*, maka metode *extended Boolean* merupakan gabungan keduanya. Selain itu juga memanfaatkan tesaurus dalam melakukan perluasan *query* sehingga satu dokumen bisa berelasi dengan dokumen lain. Di dalam melakukan pengembangan sistem, memakai metode sekuensial linier yang terdiri dari tahap analisa, desain, kode, pengujian, implementasi, dan pemeliharaan.

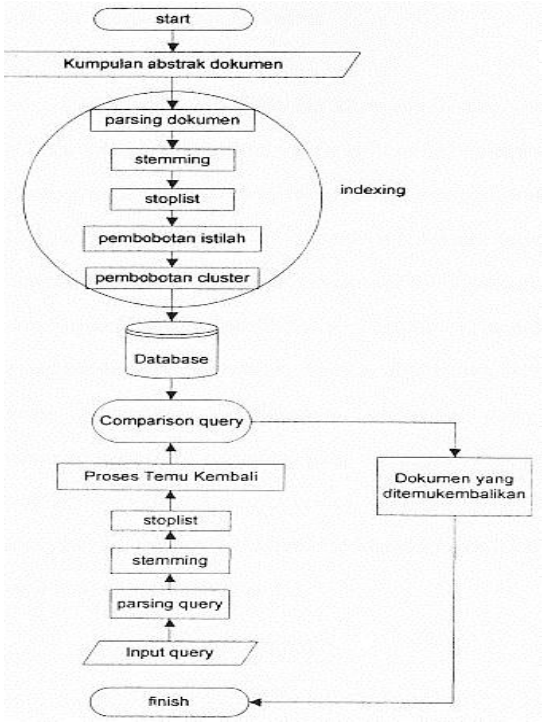
Untuk mengetahui kinerja operator OR dan AND pada *extended Boolean*, melakukan uji coba bahwa operator OR lebih banyak mendapatkan dokumen relevan dibandingkan dengan operator AND, tetapi dengan operator OR juga jumlah dokumen yang tidak relevan terambil sangat banyak. Sedangkan kinerja temu kembali informasi pada tesaurus yaitu lebih mendapatkan dokumen yang relevan pada *query* yang spesifik daripada *query* yang umum. *Query* yang lebih efektif digunakan pada metode *extended Boolean* yaitu *query* panjang (lebih dari 2 suku kata) untuk mendapatkan dokumen yang benar-benar relevan. Sedangkan pada tesaurus, *query* pendek

Copyright © 2009, Studia Informatika: Jurnal Sistem Informasi, p-ISSN 1979-0767

(kurang dari 3 suku kata) lebih efektif untuk digunakan.

2. ANALISIS

Dalam melakukan pencarian dokumen dalam suatu sistem, terkadang pengguna ingin mendapatkan dokumen yang sangat relevan atau sesuai dengan yang diinginkannya. Di lain sisi, sistem hanya sebuah alat yang tidak bisa selalu memberikan hasil yang sangat relevan dan kadang memberikan hasil yang relevan saja atau sama sekali tidak relevan. Hal itu dapat diatasi dengan menggunakan metode *extended Boolean* operator AND untuk menghasilkan dokumen yang benar-benar relevan berdasarkan istilah query yang dicarinya, atau bisa memanfaatkan extended boolean operator OR untuk menghasilkan dokumen yang relevan berdasarkan minimal satu istilah *query* yang cocok. Selain metode *extended Boolean*, bisa juga dilakukan dengan memanfaatkan tesauras untuk memperluas *query* agar didapatkan dokumen relevan berdasarkan hubungan di antara istilah *query* dengan istilah yang ada pada basis data. Dalam memahami tahap analisa ini, merancang sistem temu kembali informasi yang dapat dilihat pada Gambar 1.



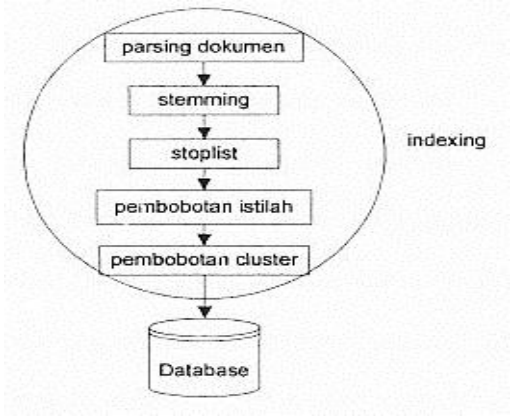
Gambar 1. Rancangan Sistem Temu Kembali Informasi

Berdasarkan rancangan Gambar 1, terdapat banyak proses yang terjadi di dalamnya. mengidentifikasikan kebutuhan sistem yang terdapat dalam tahap analisa ini, yaitu:

2.1 Analisis Perancangan Proses Sistem *Indexing*

Dalam pengindeksan ini akan dilakukan pemilihan istilah. Proses pengindeksan ini dimulai dari menyiapkan korpus atau kumpulan dokumen yang merupakan sebagai domain informasi. Korpus yang dipakai di sistem ini adalah kumpulan abstrak skripsi Fakultas Sains dan Teknologi UIN sebanyak 120 abstrak dan abstrak Fakultas Ilmu Komputer sebanyak 131.

Dalam pengindeksan selanjutnya, korpus di-*parsing* menjadi kata-kata yang kemudian dihilangkan imbuhan yang tidak perlu (*stemming*), lalu dibuang kata buangan (*stoplisf*), dilakukan pembobotan istilah, pembobotan *cosin* untuk *cluster* dan terakhir menyimpan istilah ke dalam *database*. Proses pengindeksan ini bisa dilihat pada Gambar 2.



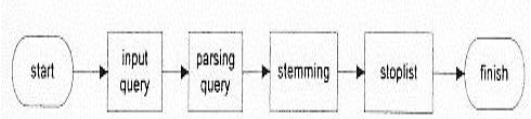
Gambar 2. Analisa *Indexing*

2.2 Pemrosesan *Query*

Untuk mendapatkan hasil yang relevan, *query* yang dimasukkan berupa teks dengan tanda petik, atau tanpa tanda petik dan pengguna bisa juga memasukkan parameter atau nilai ambang. Contoh:

Copyright © 2009, Studia Informatika: Jurnal Sistem Informasi, p-ISSN 1979-0767

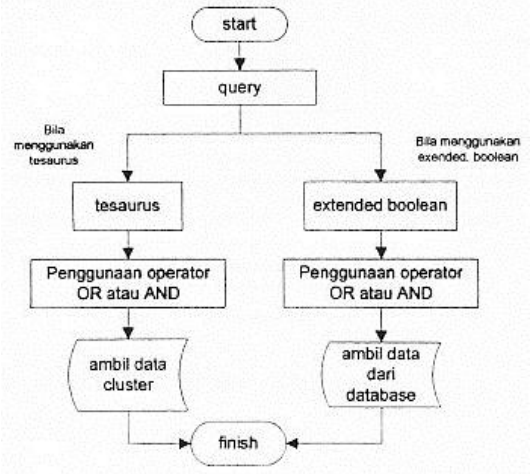
sistem pakar dengan nilai ambang 1
" sistem pakar" dengan nilai ambang 100
Query yang diproses ke dalam sistem, akan melalui tahap parsing, stemming, stoplist. Untuk lebih jelasnya bisa dilihat pada Gambar 3.



Gambar 3. Analisa *Query*

2.3 Proses Temu Kembali

Di dalam proses temu kembali ini, terdapat dua metode yang digunakan yaitu metode *extended Boolean* dan metode tesaurus. Dalam penggunaan metode *extended Boolean* diperlukan karakter tanda petik ("") untuk mendapatkan hasil *query* dengan menggunakan operator AND. Sedangkan jika tidak menggunakan karakter tanda petik, hasil *query* menggunakan operator OR. Untuk metode tesaurus, hasil *query* didapat dari operator OR/AND dan diperluas dengan pengambilan data *cluster*. Proses temu kembali dapat dilihat pada Gambar 4.



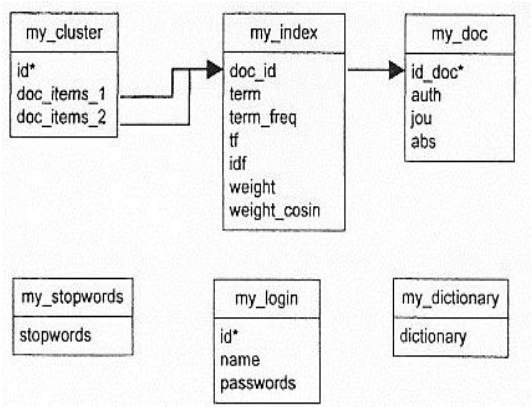
Gambar 4. Proses Temu Kembali

2.4 Analisa *Database*

Database yang digunakan untuk kebutuhan sistem terdiri dari 1 *database* dengan 6 tabel. *Database myindex* yang dipakai pada sistem terdiri dari enam tabel, yaitu:

- mydoc
- myindex
- mycluster
- my_stopswords
- my_dictionary
- mylogin

Gambar 5 menampilkan relasi antar tabel dalam *database*.



Gam bar 5. *Database Myindex*

Tahap ini selanjutnya adalah:

2.5 Analisa Perancangan Antarmuka

Untuk memudahkan pemakai (*user*) dalam melakukan pencarian temu kembali informasi, membuat antarmuka dengan menggunakan bahasa php, html serta ess dan agar lebih interaktif dipakai pula bahasa *javascript*. Perancangan antar muka yang akan dibuat terdiri dari tujuh halaman, yaitu:

- halaman login
- halaman pengindeksan dan peng- *cluster*-an
- halaman *term frequency* (tf)
- halaman *inverse document frequency* (idf)
- halaman bobot istilah
- halaman pencarian tesaurus
- halaman efektifitas dan efisiensi STKI

2.6 Desain

Berdasarkan analisa yang lakukan, maka pada tahap ini dilakukan desain perancangan proses sistem, basis data, dan antarmuka (*interface*).

Perancangan Proses Sistem

Berdasarkan hasil analisa, sistem temu kembali informasi yang dikembangkan dalam penelitian ini terdiri dari proses-proses berikut:

- *Indexing*
- Pemrosesan *Query*
- Proses Temu kembali dokumen

Rancangan proses sistem temu kembali untuk berbahasa Indonesia dapat dilihat Gambar 4.

- *Indexing*
  Korpus

Tahap pertama yang dilakukan pada tahap *indexing* ini adalah persiapan korpus. Korpus yang dipakai yaitu kumpulan abstrak dengan ekstensi txt. Adapun format an untuk setiap

Copyright © 2009, Studia Informatika: Jurnal Sistem Informasi, p-ISSN 1979-0767

abstrak adalah sebagai berikut:
DOC : {/d abstrak dokumen}
TIT : {judul}
AUT : {penulis}
ABS : {isi abstrak}
Contoh secara lengkap dilampirkan pada Lampiran V.

- *Parsing* dokumen

a. Tujuan
Menghasilkan pasangan istilah tanpa tanda baca.
b. Algoritma
//fungsi untuk memparsing karakter tanda baca
larik a_doc mempunyai data id_doc, abstrak
iterasi abstrak
data abstrak direplace tanda bacanya dengan karakter""

Prosedural algoritma di atas dapat dilihat pada Gambar 6.

Gambar 6. *Parsing* Dokumen

c. Masukan
Korpus yang terdiri dari kumpulan abstrak dokumen.
d. Keluaran
Keluaran dari proses ini adalah abstrak yang tidak mengandung karakter tanda baca.

- *Stemming* kata

a. Tujuan
Menghasilkan kata yang tidak berimbuhan.
b. Algoritma
varajdoc mempunyai data idjdoc, abs iterasi a_doc
buat larik a_abs yang memisahkan sebab kata dim abs dengan parameter spasi
iterasi a_abs sebagai *word*
Penggal kata imbuhan menjadi kata dasar
Pemanggilan fungsi *stoplist*

Prosedural algoritma di atas, dapat dilihat pada Gambar 7.

Gambar 7. *Flowchart Stemming*

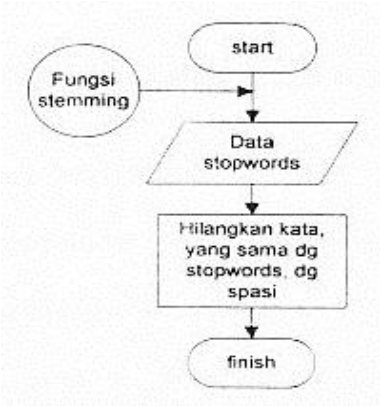
c. Masukan
Data abstrak yang mengandung kata tanpa karakter tanda baca.
d. Keluaran
Keluaran dari proses ini adalah kata-kata tak berimbuhan.

- Buang kata buangan (*stoplist*)

a. Tujuan
Menghasilkan kata yang bersih dari kata buangan.
b. Algontma
inisialisasi item_stoplist
item_stoplist = berkas_stoplist
jika stemj != item_stoplist
maka ambil stem_i dan id_dok dan simpan ke berkas
index_data

Prosedural algontma di atas, dapat dilihat pada Gambar 8.

Copyright © 2009, Studia Informatika: Jurnal Sistem Informasi, p-ISSN 1979-0767



Gambar 8 Flowchart Stoplist

c. Masukan
Data abstrak yang mengandung kata tanpa imbuhan.

d. Keluaran
Keluaran dari proses ini adalah kata-kata yang tidak mengandung kata buangan.

2.7 Pembobotan Istilah

Pembobotan istilah ini terdiri dari 3 macam yaitu: pembobotan tf, pembobotan idf, dan pembobotan tf.idf.

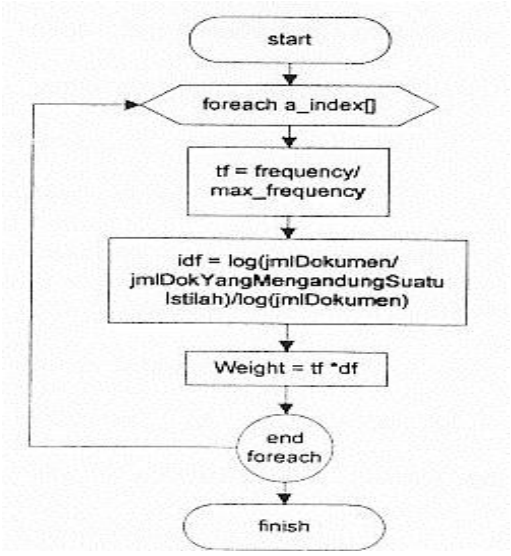
a. Tujuan

- Pembobotan Tf
  Mendapatkan nilai T/untuk pasangan nilai istilah dan id dokumen dengan menggunakan perhitungan. Frekuensi pasangan istilah – iddokumen Max(frekuensi _ i, j)
- Pembobotan Idf
  Mendapatkan nilai Idfxmtvk pasangan nilai istilah dan id dokumen dengan perhitungan Idfij -
- Pembobotan tf.idf
  Mendapatkan nilai Tf.idf sebagai bobot untuk masing-masing pasangan istilah dan id dokumennya. Nilai Tf.idf didapat dari rumus sebagai berikut: Tf.idf = Tf* Idf

b. Algoritma

Algoritma:
var N berisi jumlah dokumen keseluruhan ajndex mempunyai data id_doc, term, frequency iterasi ajndex sebagai term=>frequency
inisialisasi ajndex tf = frequency/maxima!_frequency
inisialisasi ajndex idf = log(N/jmlDokYangMengandungSuatulstilah)/log(N)
inisialisasi ajndex weight = ajndex tf * ajndex idf

Prosedural algoritma di atas dapat dilihat pada *flowchart* Gambar 9.



Gambar 9. *Flowchart* Pembobotan Istilah

c. Masukan

- Pembobotan Tf
  Masukan dalam pembobotan Tf adalah nilai frekuensi pasangan istilah dan id dokumen serta nilai maximum untuk pasangan istilah dan id dokumen.
- Pembobotan Idf
  Masukan dalam pembobotan Idf adalah jumlah dokumen («) dan jumiah dokumen yang memiliki istilah (dfk).
- Pembobotan Tf.Idf
  Masukan dalam pembobotan Tf.Idf adalah nilai Tf yang dihasilkan pada pembobotan 7y dan nilai Idf yang dihasilkan pada pembobotan Idf.

d. Keluaran

- Pembobotan Tf
  Keluaran dari pembobotan Tf ini adalah nilai Tf untuk pasangan istilah dan id dokumen.
- Pembobotan Idf
  Keluaran dari pembobotan Idf ini adalah matrik nilai idf untuk istilah ke i untuk semua id dokumen.
- Pembobotan Tf.Idf
  Keluaran dari pembobotan Tf.Idf mi adalah matrik nilai Tf.Idf untuk istilah sebagai bobot untuk pasangan istilah dan id dokumennya.

- Pembobotan untuk *cluster*

a. Tujuan

- Perkalian vektor dokumen
  Membentuk matriks pasangan dokumen *cluster*.
- Pembentukan cluster dengan metode *complete link cluster*
  Membuat pasangan *cluster*.

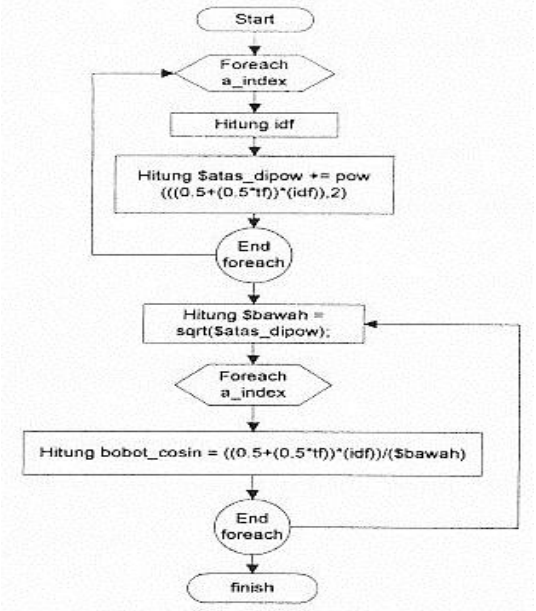
b. Algoritma

$N = jumlah larik $this->a_doc inisialisasi var

Copyright © 2009, Studia Informatika: Jurnal Sistem Informasi, p-ISSN 1979-0767

$atas_dipow = 0 iterasi larik $this->a_doc
hitung $idf
$atas_dipow += pow(((0.5+(0.5*tf))*($idf)),2); $bawah = sqrt($atas_dipow); iterasi larik $this->a_index
hitung bobotcosin = ((0.5+(0.5*tf))*($idf))/($bawah);

Prosedural algoritma di atas dapat dilihat pada *flowchart* Gambar 10.

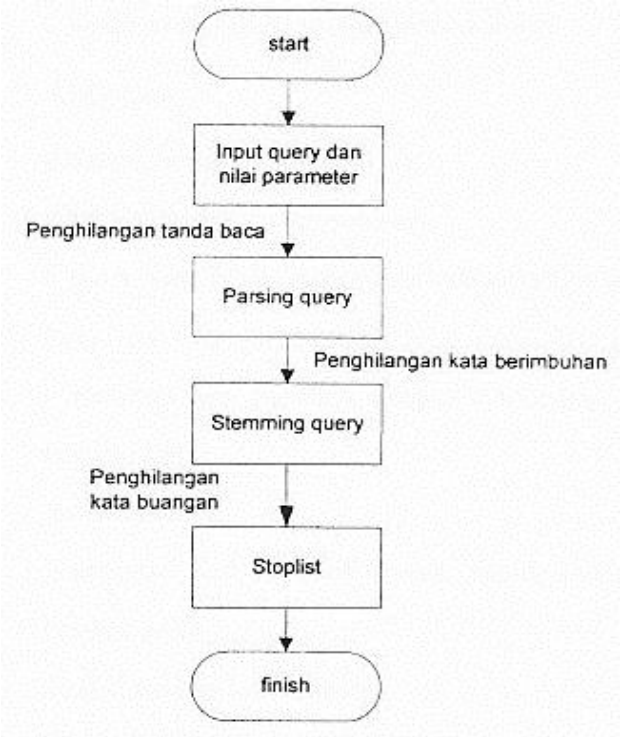


Gambar 10. *Flowchart* Perkalian Vektor Dokumen

c. Masukan
- Perkaiian vektor dokumen
  Istilah yang mempunyai term frekuensi.
- Pembentukan *cluster* dengan metode *complete link cluster*
  Dokumen kluster yang mengandung istilah-istilah yang mirip.

d. Keluaran
- Perkalian vektor dokumen
  Keluaran dari proses ini adalah bobot hasil perkalian vektor dokumen dengan menggunakan metode *cosine*.
- Pembentukan *cluster* dengan metode *complete link duster*.
  Keluaran dari penyusunan tesaurus ini adalah istilah-istilah yang mirip dalam satu dokumen kluster.

- Pemrosesan *Query*
  a. Tujuan
  Memproses *query* pemakai dalam rangka temu kembali informasi.
  b. Algoritma
  Adapun algoritma modul proses *query* ini adalah sebagai berikut:
  input query
  penghilangan tanda baca pada query //parsing query penghilangan imbuhan pada query //stemming query penghilangan kata buangan pada query //stoplist query
  Proses ini dapat dilihat pada Gambar 11.



Gambar 11. *Flowchart* Pemrosesan *Query*

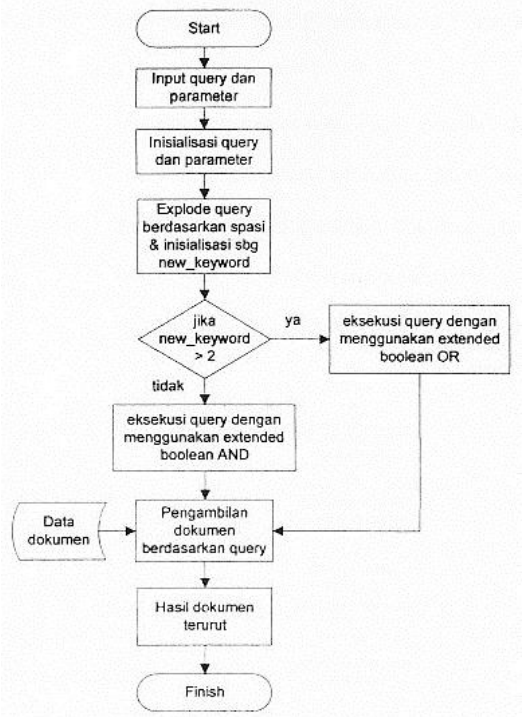
c. Masukan
Masukan di dalam pemrosesan *query* ini adalah *query* dan parameter dari *user*.

d. Keluaran
Keluaran dari pemrosesan *query* ini adalah kata dasar tanpa tanda baca.

- Proses Temu Kembali
  Menemukan kembali dokumen dengan *Extended Boolean*
  a. Tujuan
  Menghasilkan dokumen yang sesuai dengan *query* dengan menggunakan operator *Boolean*.
  b. Algoritma
  Input *query* dan parameter inisialisasi *query* dan parameter $new_keyword = explode query berdasarkan spasi jika jumlah new _keyword > 2
  eksekusi new_keyword dengan menggunakan *extended Boolean* OR jika tidak eksekusi new_keyword dengan menggunakan *extended Boolean* AND,
  ambil dokumen dari *database* Hasil dokumen diurut berdasarkan peringkat tertinggi.

Copyright © 2009, Studia Informatika: Jurnal Sistem Informasi, p-ISSN 1979-0767

Proses ini bisa dilihat pada ilustrasi Gambar 12.

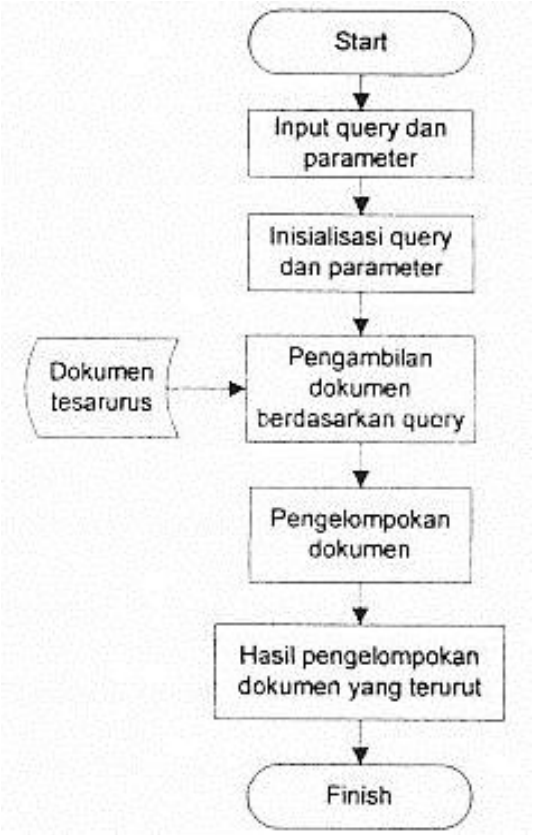


Gambar 12. Menemukan kembali dokumen dengan *Extended Boolean*

c. Masukan
Masukan di dalam proses ini adalah *query* dari *user* dan parameter untuk p-*value*.

d. Keluaran
Keluaran dari proses *query* ini adalah terambilnya dokumen terurut dengan menggunakan operator *Boolean*.

- Menemukan kembali dokumen dengan Tesaurus
  a. Tujuan
  Menampilkan dokumen yang terambil berdasarkan *query* dari pengguna kemudian dikelompokkan dan diurut berdasarkan peringkat dokumen yang tertinggi sampai yang terendah. Hasil proses ini bisa dilihat pada ilustrasi Gambar 13.



Gambar 13. Menemukan kembali dokumen dengan Tesaurus

b. Algoritma
Input *query* dan parameter
Inisialisasi *query* dan parameter
pemrosesan *query*
ambil Dokumen tesaurus dari *database*
Pengambilan dokumen tesarurus berdasarkan *query*
Pengelompokan dokumen
Hasil pengelompokan dokumen yang terurut

c. Masukan
Masukan di dalam proses ini adalah p-*value* dari *user* sebagai parameter untuk membobot dokumen.

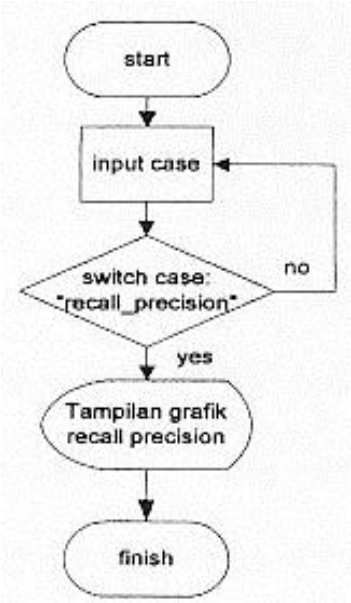
d. Keluaran
Keluaran dari proses *query* ini adalah dokumen yang dikelompokkan berdasarkan *query* yang diinginkan.

- Menghitung efektifitas dan efisiensi sistem temu kembali informasi
  a. Tujuan
  Mengukur efektifitas dari sebuah sistem temu kembali informasi yang telah dibuat.
  b. Algoritma
  jika $input = "recall_precisionn tampilkan grafik

Copyright © 2009, Studia Informatika: Jurnal Sistem Informasi, p-ISSN 1979-0767

recall precision

Prosedural algoritma di atas dapat dilihat pada *flowchart* Gambar 14.



Gambar 14. *Flowchart Recall Precision*

c. Masukan
Masukan di dalam proses ini adalah jumlah dokumen yang terambil, jumlah dokumen yang relevan bagi pengguna, dan jumlah dokumen relevan keseluruhan.

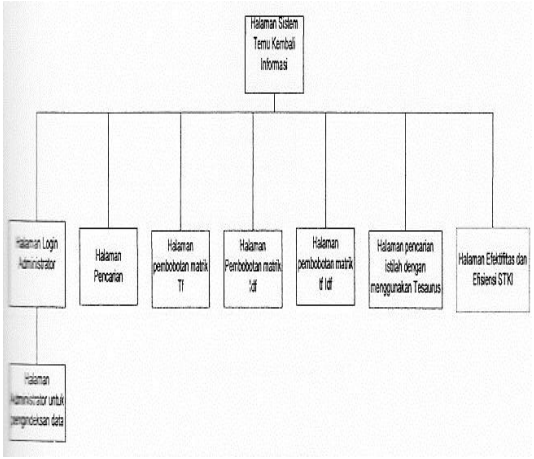
d. Keluaran
Keluaran dari proses ini adalah grafik recall dan precision untuk suatu *query* yang sudah ditentukan sebelumnya.

2.8 Proses Manual

Berikut ini proses manual untuk sistem yang akan dijelaskan seperti di bawah ini.

a. Mempersiapkan kumpulan abstrak skripsi yang tersusun menjadi satu *file*.
b. Mem-*parsing* dokumen *file* tersebut, dan keluarannya adalah kata tanpa karakater tanda baca. Contoh:
Teks asli: "Bahwa membangun keluarga sejahtera hendaknya dimulai dengan mengetahui faktor-faktor dominan".
Setelah di-*parsing*: Bahwa membangun keluarga sejahtera hendaknya dimulai dengan mengetahui faktor faktor dominan BKKBN 2001
c. Melakukan *stemming* pada kata berimbuhan, dan hasil keluarannya adalah kata tanpa imbuhan. Contohnya yaitu hasil kalimat yang di-parsing di atas, kemudian di-*stemming* menjadi:
Bahwa bangun keluarga sejahtera hendak mulai dengan mengetahui faktor faktor dominan BKKBN 2001
d. Membuang kata-kata yang tidak perlu (*stoplist*) dan hasil keluarannya adalah kata yang bersih dari kata buangan. Contohnya yaitu menggunakan hasil kalimat yang da-*stemming* di atas, kemudian di-*stoplist* menjadi:
Bangun keluarga sejahtera mulai mengetahui faktor faktor dominan BKKBN 2001
e. Membobot kata (istilah). Contoh bobot untuk istilah "bkkbn":
Term frekuensi (tf): frekuensi istilah/max frekuensi =
5/8 = 0.625
Inverse Document Frequency (idf): log (251/1) / log
251 = 1
Bobot: tfidf = 0.625/1 = 0.625
f. Membobot kata untuk peng-*cluster*-an. Contoh bobot *cluster* untuk istilah "bkkbn":
Bagian-bawah = V((0.5 + (0.5*0.625)) * (log (251 /I)))2 Bobot cluster = ((0.5 + (0.5*0.625))* 1) / Bagian-bawah = 0.00783326028529
g. Memasukan kata, dan hasil pembobotan ke dalam *database*.
h. *User* melakukan pencarian dokumen ke dalam sistem
i. Sistem akan melakukan *parsing query* agar tidak ada karakter tanda baca.
Contoh *query*: sistem?? pakar,
Contoh *query* setelah di-*parsing*: sistem pakar
j. Sistem akan melakukan *stemming* pada *query* dan hasil keluarannya adalah tidak ada kata imbuhan.
Contoh query: pendukung keputusan
Contob *query* setelah di-*stemming*: pendukung putus
k. Sistem akan membuang kata buangan (*stoplist*) Contoh *query*: informasi akademik dengan menggunakan sms.
Contoh *query* setelah di-*stoplist*: informasi akademik sms
1. Sistem akan memproses *query* dengan operator OR atau AND jika menggunakan metode *extended Boolean.*
Contoh *query* OR: sistem informasi sekolah
Contoh *query* AND: "sistem informasi sekolah"
m. Sistem akan memproses *query* dengan tesaurus jika menggunakan metode tesaurus.
n. Sistem akan mencocokan *query* dengan yang ada di *database*,
o. Sistem akan mengambil dokumen yang terurut dan akan dikembalikan ke penunjuk.

Copyright © 2009, Studia Informatika: Jurnal Sistem Informasi, p-ISSN 1979-0767



Gambar 15. Struktur Program STKI

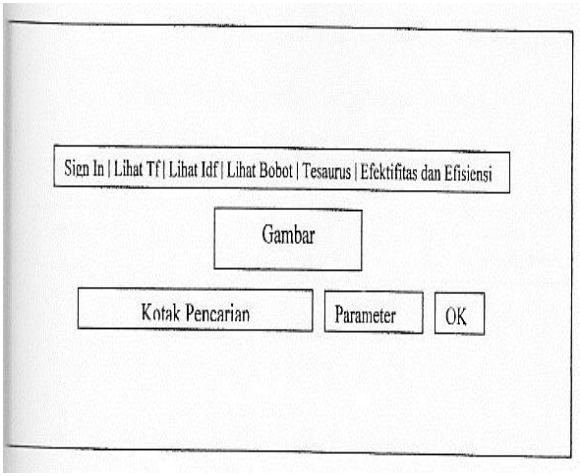
2.7 Rancangan Halaman Utama Sistem Temu Kembali Informasi

Pada halaman STKI terdapat tujuh halaman modul yaitu: Halaman pencarian istilah dengan *extended Boolean*, halaman pembobotan matrik tf, halaman pembobotan matrik idf, halaman pembobotan matrik tf, idf, dan halaman pencarian istilah dengan tesaurus. Halaman utama STKI ini dapat dilihat pada Gambar 15.

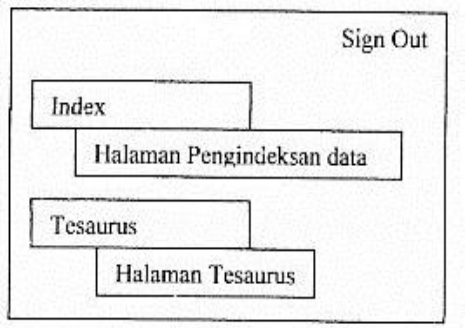
Adapun *State Transistion Diagram* masing-masing halaman adalah sebagai berikut:

a. Halaman *Administrator*

Untuk halaman *administrator*, ada halaman pengindeksan data dan tesaurus yang digabung menjadi satu halaman (Gambar 23).

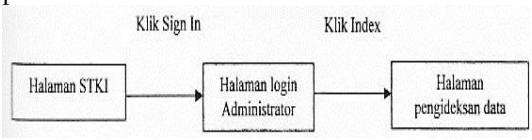


Gambar 22. Rancangan Halaman Utama STKI



Gambar 23. Rancangan Halaman *Administrator*

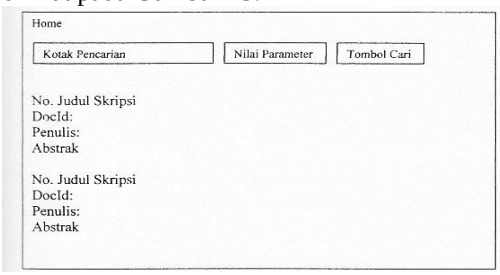
Berikut ini rancangan halaman administrator digambarkan dalam STD (*State Transition Diagram*) pada Gambar 24.



Gambar 24. STD Halaman *Administrator*

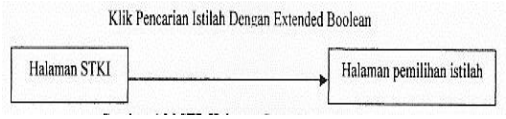
b. Halaman Pencarian Istilah dengan *Extended Boolean*

Halaman pencarian dengan *extended Boolean* dapat dilihat pada Gambar 25.



Gambar 25. Rancangan Halaman Pencarian

STD dari rancangan di atas dilihat pada Gambar 26.

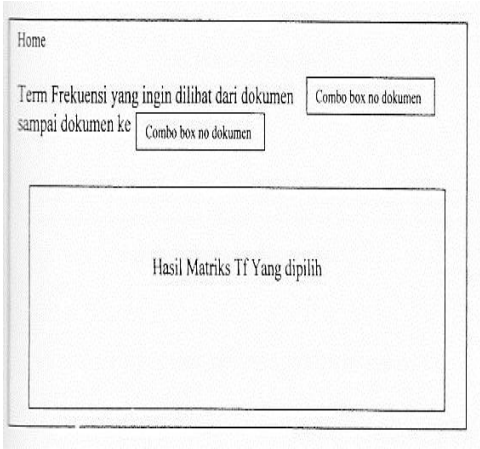


Gambar 26. STD Halaman Pemilihan Istilah

c. Halaman Pembobotan Matrik tf

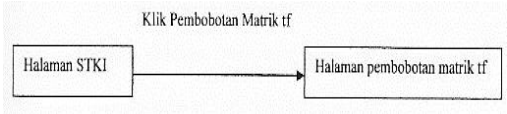
Halaman ini hanya untuk menampilkan frekuensi pada *term* di masing dokumen yang sebelumnya di tentukan dahulu dokumen mana saja yang hendak ditampilkan.

Copyright © 2009, Studia Informatika: Jurnal Sistem Informasi, p-ISSN 1979-0767



Gambar 27. Halaman Pembobotan Matrik tf

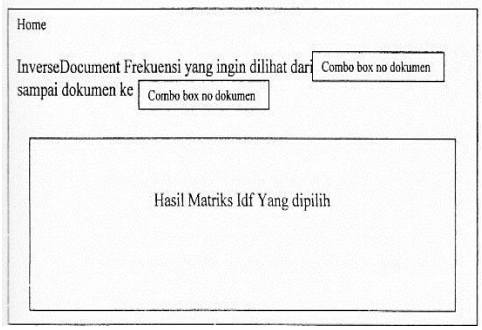
STD pada rancangan di atas dapat dilihat pada Gambar 28.



Gambar 28. STD Halaman Pembobotan Matrik tf

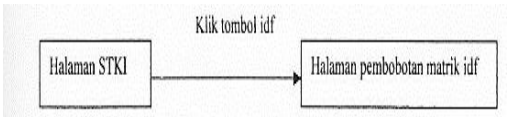
d. Halaman pembobotan matrik idf

Halaman ini hanya untuk menampilkan Idf pada term di masing dokumen yang sebelumnya di tentukan dahulu dokumen mana saja yang hendak ditampilkan.



Gambar 29. Halaman Pembobotan Matrik idf

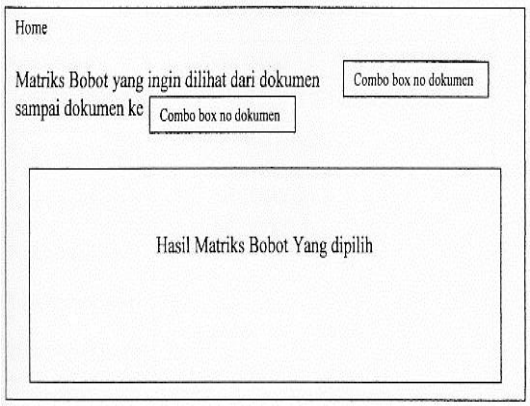
STD pada rancangan di atas dapat dilihat pada Gambar 30.



Gambar 30. STD Halaman Pembobotan Matrik idf

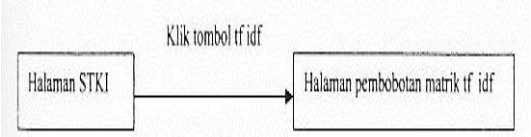
e. Halaman pembobotan matrik tf idf (bobot)

Halaman ini hanya untuk menampilkan bobot pada term di masing dokumen yang sebelumnya ditentukan dahulu dokumen mana saja yang hendak ditampilkan.



Gambar 31. Halaman Pembobotan Matrik tf idf

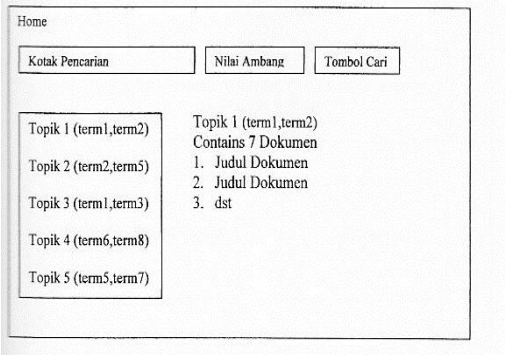
STD pada rancangan di atas dapat dilihat pada Gambar 32.



Gambar 32. STD Halaman Pembobotan Matrik tfidf

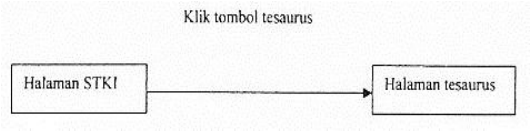
f. Halaman tesaurus

Halaman ini untuk menampilkan hasil pencarian dengan menggunakan metode tesaurus. Untuk lebih jelasnya, dapat dilihat pada Gambar 33.



Gambar 33. Halaman Tesaurus

STD pada rancangan di atas dapat dilihat pada Gambar 34.

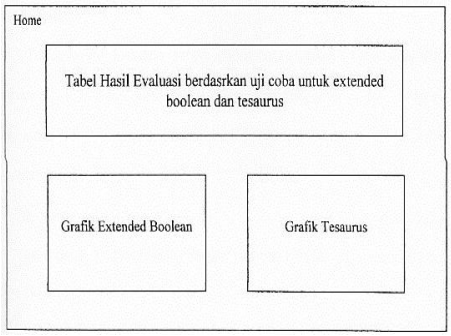


Gambar 34. STD Halaman Tesaurus

g. Halaman Efektifitas dan Efesiensi STKI

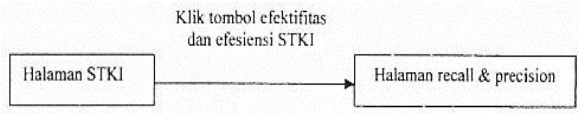
Halaman ini untuk menampilkan Efektifitas dan Efesiensi STKI dengan menggunakan rumus *Recall Precision* yang dapat dilihat pada Gambar 35.

Copyright © 2009, Studia Informatika: Jurnal Sistem Informasi, p-ISSN 1979-0767



Gambar 35. Halaman Efektifilas dan Efisiensi STKI

STD pada rancangan di atas dapat dilihat pada Gambar 36.



Gambar 36. STD Halaman Efektifitas dan Efisiensi STKI

3. PENGUJIAN

Pengujian ini dilakukan dengan menggunakan pengujian efektifitas dan efisiensi sistem temu kembali informasi serta pengujian sistem. Adapun komputer untuk pengujian beberapa *query* mempunyai spesifikasi sebagai berikut:

• *Hardware*:
- Komputer AMD *Athlon* XP dengan prosesor 1.5 Ghz
- Memori 512MB
- *Harddisk* 40 GB

• *Software*:
- *Notepad* ++ versi 4.0.2
- *Web browser Mozilla* versi 2.0.0.3
- *Xampp*-win32-1.5.3a dengan spesifikasi *apache* 2.2.2 dan mySql 5.0.21
- Sistem operasi *Windows* XP *Service Pack* 2

10. KESIMPULAN

Dengan adanya metode *extended Boolean* OR mendapatkan dokumen yang relevan dengan liai rata-rata precession 80% di titik *recall*. 50 % untuk *query* pendek OR dan nilai rata-rata 93% dititik *recall* 50%. Tetapi jumlah dokumen tidak relevan yang terambil cukup banyak. Jadi keakuratan cukup tinggi untuk menemukan dokumen dengan cepat.

REFERENSI

Akbar, Adam Salnor. 2006. Query-Sensitive Measure dalam Temu Kembali Dokumen Bebahasa Indonesia. Departemen Ilmu Komputer, http://www.arbiedesign.com/index.php?option:= com content&task=view&

Arifin, Agus Zainal & Ari Novan Setiono. 2001. Klasifikasi Dokumen Berita Kejadian Berbahasa Indonesia dengan Algoritma Single Pass Clustering. Jurnal Teknik Informatika. ITS, Surabaya.

Azis, M. Farid. 2005. Object Oriented Programming dengan PHP 5. PT. Elex Media Komputindo, Jakarta. Fitriyanti, Masayu. 1997.

Sistem Temu Kembali Informasi dengan Mengimplementasikan Operasi Boolean, Sistem Peringkat, Perbaikan Query, dan Pemanfaatan Tesaurus.

Fakultas Ilmu Komputer Indonesia Universitas Indonesia, Depok. Fox, Edward A. Extended Boolean Queries and Retrieval: 1 him. http://ei.cs.vt.edu/~cs5604/f96/cs5604cnIF/IF3.html. 13 September 2006, pk. 15:05 WIB

Grossman, David A. & Ophir Frieder. 2004. Information Retrieval: Algorithms and Heuristics; Second Edition. Springer, Netherlands. Garcia, E. 2006.

Kadir, Abdul. 1999. Konsep dan Tuntunan Praktis Basis Data. Andi, Yogyakarta

Liana, Ade. 2001. Universitas Indonesia, Depok. Mandala, Ridha & Hendra Setiawan. Peningkatan Performansi Sistem TemuKembali Informasi dengan Perluasan Query Secara Otomatis.

Martha, Taufik Leo. 2002. Penerapan Relevan Feedback Berbasis Konsep Untuk Pencarian Dokumen pada Sistem Temu Kembali Informasi. Fakultas Teknologi Informasi, Institut Teknologi Sepuluh November, Surabaya.

Prasetyo, Didik Dwi. 2003. Belajar Sendiri Administrasi Database Server MySql. PT. Elex Media Komputindo, Jakarta.

Pressman, Roger S. 2002. Rekayasa Perangkat Lunak: Pendekatan Praktisi (bukuI). Andi, Yogyakarta.

Ridha, Ahmad, Julio Adi Santoso & Fahren Bukhari. 2004. Pengindeksan Otomatis dengan Istilah Tunggal untuk Dokumen Berbahasa Indonesia.

Rijsbergen, C. J. van. 1979. Information Retrieval. Departement Of Computing Science, University Of Glasgow.

Sa'adah, Naily Ulva. 2005. Implementasi Model Ruang Vektor pada Sistem Temu Kembali Informasi. UIN, Jakarta.

Salton, G., E. A. Fox, &. H. Wu. 1983. Extended Boolean Information Retrieval. Communication of the ACM. 26(11): 1022-1036.

Salton, Gerard. 1989. Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. Addison-Wesley Publishing Company.

Copyright © 2009, Studia Informatika: Jurnal Sistem Informasi, p-ISSN 1979-0767