BUKU TESIS

REKOMENDASI PERBAIKAN PERNYATAAN KEBUTUHAN YANG RANCU DALAM SPESIFIKASI KEBUTUHAN PERANGKAT LUNAK MENGGUNAKAN TEKNIK BERBASIS ATURAN

DEPANDI ENDA

NRP. 5116201036

DOSEN PEMBIMBING :

Daniel Oranova Siahaan, S.Kom, M.Sc, PD.Eng.

PROGRAM MAGISTER

BIDANG KEAHLIAN REKAYASA PERANGKAT LUNAK

JURUSAN TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

INSTITUT TEKNOLOGI SEPULUH NOPEMBER

SURABAYA

2018

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar

Magister Komputer (M.Kom.)

di

Institut Teknologi Sepuluh Nopember Surabaya

oleh :

Depandi Enda
Nrp. 5116201036

Dengan judul :

REKOMENDASI PERBAIKAN PERNYATAAN KEBUTUHAN YANG RANCU DALAM SPESIFIKASI KEBUTUHAN PERANGKAT LUNAK MENGGUNAKAN TEKNIK BERBASIS ATURAN

Tanggal Ujian : 4-1-2018
Periode Wisuda : 2018 Gasal

Disetujui oleh :

Daniel Oranova Siahaan, S.Kom, M.Sc, PD.Eng (Pembimbing)
NIP. 197411232006041001

Dr. Ir. Siti Rochimah, M.T (Penguji 1)
NIP. 196810021994032001

Dr. Eng. Chastine Fatichah, S.Kom, M.Kom (Penguji 2)
NIP. 197512202001122002

Dr. Eng. Darlis Herumurti, S.Kom, M.Kom (Penguji 3)
NIP. 197712172003121001

Dekan Fakultas Teknologi Informasi dan Komunikasi,

Dr. Agus Zainal Arifin, S. Kom, M. Kom.
NIP. 197208091995121001

{Halaman ini sengaja dikosongkan}

REKOMENDASI PERBAIKAN PERNYATAAN KEBUTUHAN YANG RANCU DALAM SPESIFIKASI KEBUTUHAN PERANGKAT LUNAK MENGGUNAKAN TEKNIK BERBASIS ATURAN

Nama mahasiswa	:	Depandi Enda
NRP	:	5116201036
Pembimbing	:	Daniel Oranova Siahaan, S.Kom, M.Sc, PD.Eng.

ABSTRAK

Tahap awal dalam pengembangan perangkat lunak ialah menelusuri, mengumpulkan dan menyajikan segala kebutuhan pengguna ke dalam sebuah dokumen spesifikasi kebutuhan perangkat lunak (SKPL). Latar belakang akademik yang beragam, pengalaman yang berbeda, dan keterbatasan pengetahuan yang dimiliki oleh perekayasa kebutuhan memungkinkan adanya kesalahan dalam pembuatan dokumen SKPL. Salah satu kesalahan yang sering muncul pada sebuah dokumen SKPL ialah terdapatnya penggunaan kata-kata yang rancu. Hal ini tentunya dapat menyebabkan kesalahan penafsiran dan kesulitan dalam memahami kebutuhan perangkat lunak yang hendak dibangun bagi pemangku kepentingan dalam proses pengembangan perangkat lunak.

Penelitian ini bertujuan mengusulkan sebuah pendekatan untuk memberikan rekomendasi perbaikan pernyataan kebutuhan perangkat lunak yang rancu. Adapun metode yang diusulkan adalah teknik berbasis aturan dengan menggunakan model bahasa *n-gram*. Realibilitas metode usulan di evaluasi menggunakan indeks statistik Gwet's AC1. Hasil analisis metode rekomendasi yang diusulkan memiliki tingkat proporsi kesepakatan yang lebih baik dibandingkan dengan metode rekomendasi menggunakan teknik statistik berbasis frekuensi *n-gram*. Metode rekomendasi yang diusulkan memiliki nilai indeks statistik Gwet's AC1 tertinggi sebesar 0.5263 dengan tingkat proporsi kesepakatan sedang.

Kata kunci: Metode Rekomendasi, Pernyataan Kebutuhan Perangkat Lunak, Kata Rancu, Teknik Berbasis Aturan, Indeks Statistik Gwet's AC1, Teknik Statistik, N-Grams.

{Halaman ini sengaja dikosongkan}

RECOMMENDATION OF REPAIRING AMBIGUOUS REQUIREMENT STATEMENT IN SOFTWARE REQUIREMENT SPESIFICATION USING RULE BASED TECHNIQUE

Name : Depandi Enda
Student Identity Number : 5116201036
Supervisor : Daniel Oranova Siahaan, S.Kom, M.Sc, PD.Eng.

ABSTRACT

The first stage in software development is to investigate, collect and provide all user requirements into a software requirements specification document (SRS's). Diverse academic background, different experiences, and the limitations of knowledge possessed by the requirement engineer make possible mistakes in the creation of SRS's documents. One of the most common mistakes in SRS's document is the use of ambiguous words. This can certainly lead to misinterpretation and difficulties in understanding the software requirement that stakeholders to built in the software development process.

The purpose of this research is to build an approach that gives recommendation improvement of ambiguous software requirement statement. The proposed method is a rule-based technique using n-gram language model. Analysis of the test results from the recommendation method can be evaluated using Gwet's AC1 statistics. The reliability of the proposed method is evaluated using Get's AC1 statistical index. The analysis results of the proposed recommendation method have a better level of agreement proportion than the recommendation method using the n-gram frequency-based statistical technique. The proposed recommendation method has the highest Gwet's AC1 statistic value of 0.5263 with a moderate agreement proportion rate.

Keywords: Recommendation Method, Software Requirement Statement, Ambigous Word, Rule Based Technique, Gwet's AC1 Statistics Index, Statistical Technique, N-Grams.

{Halaman ini sengaja dikosongkan}

KATA PENGANTAR

Segala puji dan syukur atas kehadirat Allah SWT yang telah memberikan rahmat, hidayah dan karunia-Nya sehingga penulis dapat menyelesaikan tesis yang berjudul "Rekomendasi Perbaikan Pernyataan Kebutuhan Yang Rancu Dalam Spesifikasi Kebutuhan Perangkat Lunak Menggunakan Teknik Berbasis Aturan".

Selama pengerjaan tesis ini penulis banyak mendapat bantuan dan bimbingan dari berbagai pihak, baik berupa arahan maupun motivasi yang membangun sehingga dengan kerja keras, do'a dan kesabaran tesis ini dapat diselesaikan. Pada kesempatan ini juga penulis ingin menyampaikan ucapan terima kasih dan penghormatan yang sebesar-besarnya kepada :

1. Ibunda serta abang, kakak dan adikku tercinta yang selalu memberikan do'a dan dukungan.
2. Bapak Daniel Oranova Siahaan, S.Kom, M.Sc, P.D.Eng selaku dosen pembimbing yang telah bersedia meluangkan waktu untuk memberikan ilmu, arahan, petunjuk dan motivasi selama proses pengerjaan tesis ini.
3. Ibu Dr. Ir. Siti Rochimah, M.T, Ibu Chastine Fatichah, S.Kom, M.Kom, Ph.D, dan Bapak Dr. Eng. Darlis Heru Murti, S.Kom, M.Kom selaku dosen penguji yang telah memberikan saran, arahan dan koreksi dalam tesis ini.
4. Bapak dan Ibu dosen Jurusan Teknik Informatika ITS yang telah banyak memberikan ilmu dan bimbingan yang tak ternilai harganya.
5. Seluruh staf dan karyawan Teknik Informatika ITS yang telah banyak memberikan kelancaran administrasi akademik
6. Seluruh teman-teman seperjuangan, sahabat dan kakak tingkat di lab Pasca Sarjana Teknik Informatika ITS yang selalu memberikan dukungan dan motivasi selama pengerjaan tesis ini.
7. Keluarga besar Asrama Ikatan Pelajar Mahasiswa Bengkalis di Surabaya yang selalu memberikan dorongan dan pencerahan selama proses perkuliahan hingga selesainya pengerjaan tesis ini.

Surabaya, Januari 2018

Depandi Enda

{Halaman ini sengaja dikosongkan}

DAFTAR ISI

DAFTAR GAMBAR

{Halaman ini sengaja dikosongkan}

DAFTAR TABEL

{Halaman ini sengaja dikosongkan}

BAB I
PENDAHULUAN

1.1 Latar Belakang

Tahap awal dalam pengembangan perangkat lunak ialah menelusuri, mengumpulkan dan menyajikan segala kebutuhan pengguna kedalam sebuah dokumen spesifikasi kebutuhan perangkat lunak (SKPL). Dokumen spesifikasi kebutuhan perangkat lunak memiliki peranan yang sangat penting dalam menghasilkan perangkat lunak yang berkualitas tinggi. Sebuah dokumen spesifikasi kebutuhan yang berkualitas sebaiknya memiliki beberapa karakteristik seperti ketepatan, kelengkapan, konsisten, terverifikasi, tidak rancu dan mudah dilacak apabila terdapat modifikasi dokumen spesifikasi di masa yang akan datang dengan menyatakan dimana persyaratan telah dirujuk (Armitage, 1996).

Menelusuri dan mengidentifikasi sebuah kebutuhan adalah suatu hal yang rumit untuk dilakukan. Selalu terdapat kesalahan interpretasi kebutuhan antara pelanggan dan pengembang. Hal ini dapat menurunkan kualitas dokumen spesifikasi kebutuhan perangkat lunak. Kegiatan ini melibatkan proses penerjemahan informasi dari satu bentuk ke bentuk yang lainnya yaitu, dari kebutuhan pelanggan ke dalam bentuk arsitektur rancangan kode. Aktivitas ini biasanya dilakukan melalui komunikasi dua arah antara pelanggan dan pengembang baik secara lisan maupun tulisan yang dalam pelaksanaannya mungkin terdapat kesalahan penerjemahan kebutuhan. Hal ini tentunya memberikan tantangan tersendiri kepada pengembang dalam menyusun dan menyajikan dokumen spesifikasi kebutuhan yang berkualitas dan minim terhadap kesalahan.

Berbagai aspek latar belakang pendidikan yang beragam dan keterbatasan pengetahuan yang dimiliki oleh tim penyusun kebutuhan perangkat lunak memungkinkan adanya kesalahan didalam pembuatan dokumen spesifikasi kebutuhan perangkat lunak. Untuk itu, seorang tenaga ahli dibutuhkan untuk memvalidasi dan menganalisis kesalahan-kesalahan yang terdapat pada dokumen spesifikasi kebutuhan perangkat lunak. Tentunya hal ini dapat menambah biaya

dan waktu pengerjaan proyek perangkat lunak. Di sisi lain pengembang memiliki sumber daya yang terbatas.

Kesalahan yang sering muncul pada sebuah dokumen spesifikasi kebutuhan perangkat lunak ialah terdapatnya penggunaan kata-kata yang rancu (ambigu), hal ini tentunya dapat menyebabkan kesalahan penafsiran dan sulit untuk dipahami oleh pihak yang terlibat langsung dalam proses pengembangan. Walau bagaimanapun kesalahan ini dapat memberikan dampak secara signifikan dalam proses pengembangan tahap selanjutnya. Oleh karena itu, sebuah metode untuk mengidentifikasi dan mengurangi penggunaan kata-kata yang berpotensi bermakna rancu pada sebuah dokumen spesifikasi kebutuhan perangkat lunak sangat diperlukan.

Penelitian ini berfokus membahas pendekatan yang diusulkan untuk memberikan rekomendasi perbaikan pernyataan kebutuhan yang rancu dalam spesifikasi kebutuhan perangkat lunak. Dimana untuk mencapai tujuan tersebut peneliti mencoba untuk mengkaji beberapa penelitian sebelumnya yang telah membahas beberapa pendekatan yang digunakan untuk mendeteksi pernyataan kebutuhan yang rancu dalam dokumen spesifikasi kebutuhan perangkat lunak dan beberapa teknik untuk memberikan rekomendasi perbaikan kesalahan tata bahasa dalam bahasa Inggris.

Penelitian yang membahas tentang metode identifikasi kerancuan dalam dokumen SKPL telah dilakukan oleh beberapa peneliti antara lain, Kamsties *et al*., (2001) menggunakan pendekatan inspeksi yang dilakukan secara manual dalam mendeteksi kerancuan. Pendekatan ini masih kurang efektif karena analisis pernyataan kebutuhan dilakukan secara manual dan sangat bergantung terhadap subyektivitas dan kemampuan seorang pakar dalam menganalisis kerancuan. Penelitian berikutnya berfokus dalam mendeteksi kerancuan yang dilakukan secara otomatis seperti yang telah dibahas oleh Lami *et al.*, (2005), Hussain (2007) dan Bussel (2009). Penelitian tersebut mencoba untuk mendeteksi kerancuan dengan memanfaatkan sebuah repositori yang dibuat khusus untuk memuat daftar kata-kata yang tidak diinginkan (kata rancu). Penelitian Lami *et al.*, (2005), Hussain (2007) dan Bussel (2009) memiliki kekurangan yaitu repositori yang digunakan masih bersifat statis dan ruang lingkup permasalahan

juga bersifat terbatas hal ini tentunya akan mempengaruhi kinerja dari peforma metode.

Dalam menyelesaikan permasalahan terbatasnya ruang lingkup domain repositori, peneliti lainnya yaitu Kiyavitskaya *et al.*, (2008) mencoba untuk menggunakan sumber daya kamus kata leksikal seperti *WordReference, WordNet* dan *Babylon's* untuk memperluas repositori kata rancu. Berdasarkan penelitian Kiyavitskaya *et al.*, (2008) menunjukkan bahwa penggunaan sumber daya kamus kata *WordNet* lebih efektif jika dibandingkan dengan penggunaan kamus kata lainnya. Namun, penelitian ini memiliki kekurangan yaitu proses identifikasi kata rancu yang kurang efesien dimana teknik pencocokan kata rancu dilakukan dengan membandingkan tiap kata rancu yang terdapat didalam repositori dengan tiap bagian kata yang terdapat dalam pernyataan kebutuhan. Peneliti selanjutnya Muliawan *et al*., (2011) mengusulkan sebuah metode yang lebih dinamis dalam mendeteksi kerancuan, dimana metode ini menggunakan teknik pemrosesan bahasa alamiah dengan menerapkan aturan SMART dalam mendeteksi kerancuan pada pernyataan kebutuhan perangkat lunak. Kinerja metode ini dibantu dengan kamus kata leksikal *WordNet* untuk memperluas repositori aturan dan bekerja berdasarkan teknik pencocokan aturan. Dimana teknik pencocokan aturan lebih dapat diandalkan dalam mendeteksi kata yang rancu dengan membandingkan pola-pola aturan yang telah dibuat.

Teknik pemeriksaan dan perbaikan kesalahan tata bahasa umumnya dapat digolongkan menjadi tiga pendekatan yaitu pendekatan berbasis pencocokan pola, berbasis statitik dan berbasis aturan (Henrich dan Reuter, 2009). Pendekatan berbasis statistik telah dibahas oleh Henrich dan Reuter (2009) yang mengusulkan sebuah metode dan kakas bantu untuk memeriksa dan memperbaiki kesalahan tata bahasa yang independen terhadap jenis bahasa berdasarkan pendekatan statistik. Bahasa yang digunakan sebagai bahasa uji coba penelitian ialah bahasa Inggris dan bahasa Jerman. Hasil evaluasi menunjukkan bahwa pendekatan Henrich dan Reuter (2009) dapat bekerja untuk bahasa yang berbeda walaupun ketepatan pemeriksaan tata bahasa bervariasi, alasannya adalah karena perbedaan kekayaan morfologis bahasa. Keterbatasan metode ini ialah kuantitas dan kualitas data latih sangat menentukan peforma metode. Metode perbaikan yang diusulkan oleh

Henrich dan Reuter (2009) menunjukkan bahwa masih banyak terdapat masalah dalam menemukan semua kesalahan gramatikal. Henrich dan Reuter (2009) mengatasi masalah tersebut dengan mengusulkan sebuah gagasan yang mengkombinasikan dua pendekatan yaitu pendekatan berbasis statistik dan aturan (*hybrid*). Teknik *hybrid* akan bekerja dengan sangat baik dalam kasus bahasa yang tergantung pada satu bahasa tertentu (bahasa yang dependen), misalnya sebuah kasus permasalahan yang diselesaikan dibatasi hanya menggunakan bahasa Inggris (Henrich dan Reuter, 2009).

Teknik perbaikan kesalahan berbasis statistik lainnya juga dibahas oleh Athanaselis *et al.*, (2011) yang mengusulkan sebuah metode dalam memperbaiki kesalahan tata bahasa dalam urutan kata menggunakan model bahasa statistik *bigram* dan *trigram* yang diekstrak menggunakan sebuah korpus yang besar yaitu *British National Corpus (BNC).* Penelitian Athanaselis *et al.*, (2011) menunjukkan hasil yang cukup baik dalam mendeteksi dan memperbaiki kesalahan tata bahasa dalam urutan kata menggunakan model bahasa *bigram* dan *trigram*. Wu *et al*., (2013) mengusulkan sebuah metode berdasarkan cara kerja mesin terjemahan untuk mengoreksi kesalahan serial gramatikal pada kalimat tertentu dalam penulisan tata bahasa. Evaluasi terhadap serangkaian kalimat pada korpus latih menunjukkan bahwa metode ini dapat memperbaiki kesalahan serial dengan cukup baik menggunakan model bahasa *trigram*.

Pendekatan berbasis aturan telah dibahas oleh Naber (2003) yang mengembangkan sebuah metode dan kakas bantu yang dapat memeriksa kesalahan tata letak dan tata bahasa dalam kalimat bahasa Inggris menggunakan teknik berbasis aturan. Untuk menambah peforma metode yang diusulkan, Naber (2003) menggunakan sebuah korpus yang besar yaitu *British National Corpus* untuk memastikan bahwa sistem yang dibangun tidak melaporkan terlalu banyak kesalahan pada kalimat yang benar, sehingga akurasi sistem dapat meningkat. Teknik berbasis aturan yang diusulkan dapat digunakan untuk mengekspresikan aturan yang menggambarkan kesalahan pada tingkat frasa dan tidak hanya pada tingkat kata. Sehingga metode ini menjadi salah satu metode yang dapat diandalkan dalam memeriksa kesalahan tata bahasa (Naber, 2003).

Dari beberapa penelitian yang telah dijabarkan serta beberapa pendekatan solusi dari permasalahan yang telah dibahas, maka pada penelitian akan menerapkan metode NLP berbasis aturan SMART untuk mendeteksi kerancuan pada pernyataan kebutuhan yang ditulis menggunakan bahasa alamiah (Muliawan *et al*., 2011). Sedangkan teknik rekomendasi yang diusulkan dalam memberikan rekomendasi perbaikan pernyataan kebutuhan yang rancu adalah teknik berbasis aturan (Naber, 2003). Dimana teknik ini adalah salah satu teknik yang dapat diandalkan dalam memberikan rekomendasi perbaikan kesalahan tata bahasa. Untuk menambah peforma metode pemberian rekomendasi yang diusulkan, penelitian ini menggunakan teknik *hybrid* yang digagas oleh Henrich dan Reuter (2009) dalam menutupi keterbatasan dan kelemahan dari teknik berbasis statistik.

Teknik *hybrid* bekerja dengan mengkombinasikan dua teknik yaitu teknik berbasis aturan dan teknik berbasis statistik *n-gram* dalam meningkatkan kinerja metode. Teknik statistik *n-gram* berguna untuk menentukan pilihan kandidat kata yang akan direkomendasikan. Model bahasa *n-gram* yang digunakan pada penelitian ini terdiri dari dua yaitu model bahasa *bigram* dan *trigram* yang menunjukkan kinerja yang cukup baik dalam memberikan rekomendasi kesalahan tata bahasa Athanaselis *et al.*, (2011) dan Wu *et al*., (2013). Sedangkan untuk meningkatkan kinerja model bahasa *n-gram* Athanaselis *et al.*, (2011) dan Wu *et al*., (2013) memanfaatkan sebuah korpus yang besar sebagai kamus kata referensi pencarian pasangan *n-gram.* Sehingga pada penelitian ini juga memanfaatkan sebuah korpus yang besar yaitu *British National Corpus* (*BNC*) dalam meningkatkan kinerja metode.

Pendekatan yang diusulkan diharapkan dapat mengurangi kesalahan khususnya penggunaan kata-kata yang rancu pada sebuah pernyataan kebutuhan perangkat lunak dengan menggantikan peran seorang tenaga ahli dalam menganalisis kebutuhan.

1.2 Perumusan Masalah

Dari uraian permasalahan yang telah dipaparkan, berikut beberapa permasalahan yang dibahas pada penelitian ini antara lain sebagai berikut.

1. Mencari dan menemukan kata-kata yang menyebabkan pernyataan kebutuhan menjadi rancu.

2. Bagaimana memberikan rekomendasi kepada pengguna untuk mengubah kata-kata yang bermakna rancu dengan beberapa pilihan rekomendasi perubahan kata yang tidak rancu?
3. Bagaimana melakukan pengujian terhadap metode rekomendasi yang dibangun, apakah telah memenuhi kriteria rekomendasi perbaikan yang sesuai standar?

1.3 Batasan Masalah

Batasan dan ruang lingkup dari metode yang diusulkan pada penelitian ini antara lain sebagai berikut.

1. Pernyataan kebutuhan yang diuji berasal dari penyataan kebutuhan dalam bahasa alamiah.
2. Metode yang dibangun dapat digunakan untuk mendeteksi dan memberikan rekomendasi perbaikan kata-kata yang rancu pada pernyataan kebutuhan perangkat lunak dalam bahasa Inggris.
3. Karakteristik pernyataan kebutuhan yang diuji ialah pernyataan kebutuhan yang memiliki struktur dan tata bahasa yang baku.

1.4 Tujuan Penelitian

Adapun tujuan dari penelitian ini ialah membangun sebuah pendekatan untuk memberikan rekomendasi perbaikan pernyataan kebutuhan perangkat lunak yang rancu.

1.5 Manfaat Penelitian

Metode rekomendasi yang diusulkan diharapkan dapat memberikan manfaat kepada perekayasa dan penyusun dokumen spesifikasi kebutuhan perangkat lunak dengan menyediakan sebuah kakas bantu untuk melakukan analisis pernyataan kebutuhan perangkat lunak.

Metode rekomendasi juga dapat memberikan manfaat kepada perekayasa kebutuhan dalam menghemat waktu pengerjaan pembuatan dokumen spesifikasi kebutuhan perangkat lunak dengan menyediakan pilihan kata rekomendasi yang disarankan untuk memperbaiki pernyataan kebutuhan yang rancu.

1.6 Kontribusi Penelitian

Kontribusi pada penelitian ini adalah mengusulkan sebuah metode rekomendasi perbaikan pernyataan kebutuhan perangkat lunak yang rancu.

BAB II
KAJIAN PUSTAKA

2.1 Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak (*software requirements*) adalah atribut-atribut yang bersifat spesifik yang merupakan spesifikasi kebutuhan fungsional dan kebutuhan non fungsional dari sebuah sistem perangkat lunak. Dokumen SKPL umumnya disusun menggunakan bahasa alamiah, yaitu bahasa yang kita gunakan dalam percakapan sehari-hari. Walaupun bahasa alamiah lebih mudah dimengerti oleh pemangku kepentingan dan sangat fleksibel untuk menampung berbagai kebutuhan pengguna yang sering berubah-ubah. Namun, penggunaan bahasa alamiah memiliki kelemahan yaitu pada saat makna dari pernyataan kebutuhan perangkat lunak memiliki arti dan penafsiran yang berbeda-beda pada masing-masing orang atau bisa disebut sebagai kerancuan dalam pernyataan kebutuhan perangkat lunak.

2.2 Kerancuan Kebutuhan Perangkat Lunak

Ada beberapa jenis kerancuan yang terdapat pada pernyataan kebutuhan perangkat lunak antara lain yaitu, kerancuan leksikal, kerancuan sintaksis, kerancuan semantik, kerancuan pragmatis, kerancuan ketidakjelasan dan keumuman (Berry dan Kamsties, 2003). Kerancuan dalam sebuah pernyataan kebutuhan dapat terjadi dalam berbagai bentuk bahasa alamiah, hal ini disebabkan penggunaan kata yang rancu dalam menyusun pernyataan kebutuhan. Misalnya saja kita jumpai sebuah pernyataan kebutuhan berikut :

"Design a program that allows a network operator to plan changes in every parameter of every cell in the network. These planned changes should be verified for consistency and correctness and then applied to the network in the least disturbing way".

Dalam pernyataan diatas, tidak dijelaskan bagaimana sebuah sistem memverifikasi sebuah konsistensi dan kebenaran lalu mengimplementasikannya dalam jaringan tersebut. Sehingga pernyataan ini tidak dapat diimpementasikan dalam rancangan sistem apalagi dibangun menjadi sebuah fungsi dari sistem.

2.3 Metode Identifikasi Kerancuan Kebutuhan Perangkat Lunak

Penelitian sebelumnya yang membahas tentang metode identifikasi kerancuan kebutuhan perangkat lunak telah dilakukan oleh beberapa peneliti diantaranya ialah Kamsties *et al*., (2001) dengan judul penelitian "*Detecting Ambiguities in Requirements Documents Using Inspections*". Kamsties *et al*., (2001) mengajukan sebuah teknik inspeksi untuk mendeteksi kerancuan dalam dokumen kebutuhan yang tidak formal sebelum spesifikasi kebutuhan yang formal dibuat, agar tidak terjadi kesalahan penafsiran. Teknik inspeksi yang digunakan ialah berbasis daftar periksa (*checklist based*) dan berbasis pembacaan skenario (*scenario-based reading*). Teknik inspeksi ini memiliki kekurangan yaitu pemeriksaan kerancuan pada dokumen kebutuhan dilakukan secara manual yang membutuhkan kemampuan dan pengalaman seorang pakar rekayasa kebutuhan dalam memeriksa dan mengidentifikasi kerancuan. Hal ini tentunya kurang efektif dan sangat bergantung pada subyektivitas seorang pakar dalam menganalisis kerancuan.

Lami *et al*., (2005) mengusulkan sebuah metode dan kakas bantu yang secara otomatis dapat melakukan analisis terhadap kecacatan dokumen kebutuhan perangkat lunak yang ditulis dalam bahasa alamiah seperti mendeteksi kerancuan, ketidaklengkapan, dan kebutuhan perangkat lunak yang kontradiksi. Perkakas bantu ini dikenal dengan nama *Quality Analyzer of Requirements Specifications* (*QuARS*). *QuARS* menggunakan algoritma linguistik untuk mendeteksi potensi kecacatan pada dokumen kebutuhan. Kakas bantu ini menganalisis kualitas model seperti struktur, leksikal, sintaktik dan semantik dari tiap-tiap pernyataan kebutuhan. Kualitas model struktur, leksikal, sintaktik dan semantik diperiksa terhadap satu set kata kunci yang telah ditetapkan yang merupakan kata-kata yang tidak diinginkan. Tiap-tiap domain permasalahan membutuhkan repositori khusus yang memuat kata-kata yang tidak diinginkan pada ruang lingkup permasalahan tersebut. Hal ini sekaligus sebagai kelemahan kakas bantu ini karena ruang lingkup repositori kata-kata yang tidak diinginkan bersifat terbatas.

Hussain (2007) mengembangkan sebuah metode dan kakas bantu yang mampu mendeteksi kerancuan dalam dokumen kebutuhan secara otomatis pada tahapan elisitasi. Kakas bantu yang dihasilkan diberi nama *Requirements*

Specification Ambiguity Checker (ReqSAC). Metode yang digunakan untuk mendeteksi kerancuan ialah metode klasifikasi teks berbasis algoritma pembelajaran pohon keputusan C4.5. Kualitas, kuantitas, dan ruang lingkup data latih akan sangat menentukan performa *ReqSAC* dalam menganalisis ambiguitas kebutuhan perangkat lunak. Jumlah ahli yang terlibat akan menentukan kualitas data latih (Hussain, 2007). Sedangkan di dalam penelitiannya (Hussain, 2007) hanya melibatkan tiga orang ahli untuk membentuk data latih pada analisis tingkat kalimat *(sentence-level analysis)*. Ditinjau dari sisi kuantitas dan ruang lingkup data latih yang digunakan masih bersifat terbatas pada 25 buah topik proyek perangkat lunak. Hussain membuat sebuah repositori kata-kata yang bersifat rancu secara otomatis berdasarkan data latih yang digunakan. Hal ini merupakan kelemahan *ReqSAC* karena repositori yang dihasilkan tersebut tentu saja belum handal dan ruang lingkupnya masih bersifat terbatas.

Bussel (2009) mengusulkan sebuah metode untuk mendeteksi kerancuan secara otomatis dalam dokumen spesifikasi kebutuhan perangkat lunak menggunakan algortima klasifikasi pada mesin pembelajaran yaitu IB1 standar, IB1 *Modified Value Difference Metric* (MVDM) dan *Inverse Learning Weighting*. Hasil penelitian yang telah dilakukan menunjukkan bahwa algoritma IB1 standar memiliki peforma yang lebih baik jika dibandingkan dengan algoritma lainnya. Namun, karena penggunaan korpus yang kecil dalam menguji algoritma untuk data yang baru dan tidak berada pada ruang lingkup domain data latih, maka metode yang dikembangkan belum mampu untuk mengenali kerancuan dalam dokumen spesifikasi kebutuhan secara akurat. Sehingga hal ini menjadi kelemahan dari metode ini karena akurasi dan peforma metode sangat bergantung terhadap banyaknya data dan ruang lingkup data latih.

Kiyavitskaya *et al.*, (2008) mengusulkan dua langkah pendekatan dalam mengidentifikasi kerancuan pada pernyataan kebutuhan yang ditulis menggunakan bahasa alamiah. Pada langkah pertama, kakas bantu akan menerapkan satu set langkah-langkah untuk mengukur kerancuan dan mengidentifikasi kalimat yang berpotensi rancu pada pernyataan kebutuhan. Langkah kedua, kakas bantu lainnya akan menunjukkan secara spesifik bagian mana pada masing-masing kalimat yang berpotensi rancu. Untuk memperluas

penggunaan repositori kata-kata yang rancu agar tidak hanya terbatas pada beberapa ruang lingkup domain, Kiyavitskaya *et al.*, (2008) membandingkan penggunaan tiga sumber daya kamus kata yang dapat diakses secara publik dalam mengidentifikasi kerancuan leksikal yaitu, *WordReference*, *WordNet* dan *Babylon's*. Berdasarkan hasil penelitian Kiyavitskaya *et al.*, (2008) menunjukkan bahwa penggunaan sumber daya kamus kata *WordNet* lebih efektif jika dibandingkan dengan penggunaan kamus kata lainnya. Namun, penelitian ini memiliki kekurangan yaitu proses identifikasi kata rancu yang kurang efesien dimana teknik pencocokan kata rancu dilakukan dengan membandingkan tiap kata rancu yang terdapat di dalam repositori dengan pernyataan kebutuhan.

Muliawan *et al*., (2011) mengusulkan sebuah metode untuk mendeteksi kerancuan pada pernyataan kebutuhan perangkat lunak menggunakan pemrosesan bahasa alamiah (NLP) dengan menerapkan aturan SMART (*Spesific*, *Measureable, Attainable, Reliazable, and Trackable*). Kinerja metode yang dibangun dibantu oleh kamus kata leksikal *WordNet* sebagai repositori dari kata-kata yang tidak diinginkan (*non-desired words)* yang bersifat andal dan dinamis, yaitu mudah menyesuaikan diri dengan sistem yang akan dibangun. Metode ini bekerja berbasis sekumpulan aturan yang berbentuk pola frasa tiap bagian kata atau POS (*Part of Speech*) beserta kata ambigu. Aturan ini dihasilkan oleh NLP dari daftar aturan yang ada dalam SMART *Requirements*. Adapun pola aturan yang dihasilkan pada penelitian ini berjumlah 27 buah pola yang mengindikasikan kerancuan dalam dokumen SKPL. Teknik pencocokan pola aturan yang digunakan pada penelitian Muliawan *et al*., (2011) dapat lebih cepat dan efesien dalam mendeteksi kata yang rancu dengan membandingkan pola-pola aturan yang telah dibuat.

Untuk memberikan gambaran yang lebih jelas mengenai kelebihan dan kekurangan dari beberapa metode yang telah dibahas maka dapat ditinjau pada tabel komparasi metode berikut.

Tabel 2.1 Komparasi metode identifikasi kerancuan kebutuhan perangkat lunak

No	Judul Penelitian	Metode Usulan	Kelebihan	Kekurangan
1	*Detecting Ambiguities in Requirements Documents Using Inspections* (Kamsties *et al*., 2001)	Mengusulkan sebuah metode inspeksi berbasis daftar periksa (*checklist based*) dan berbasis pembacaan skenario (*scenario-based reading*) dalam menganalisis kerancuan dokumen kebutuhan perangkat lunak.	Metode yang sederhana dan konvensional dalam melakukan analisis kerancuan kebutuhan perangkat lunak.	Pemeriksaan kerancuan pada dokumen kebutuhan dilakukan secara manual yang membutuhkan kemampuan dan pengalaman seorang pakar rekayasa kebutuhan dalam memeriksa dan mengidentifikasi kerancuan.
2	*An Automatic Tool for the Analysis of Natural Language Requirements* (Lami *et al.*, 2005)	Mengusulkan sebuah metode dan perkakas bantu yang secara otomatis dapat melakukan analisis terhadap kecacatan dokumen kebutuhan perangkat lunak yang ditulis dalam bahasa alamiah.	Identifikasi kerancuan dapat dilakukan secara otomatis menggunakan sebuah perkakas bantu yang dikenal dengan nama *Quality Analyzer of Requirements Specifications (QuARS).*	Ruang lingkup repositori kata-kata yang tidak diinginkan bersifat terbatas.
3	*Using Text Classification to Automate Ambiguity Detection in SRS Documents* (Hussain, 2007)	Mengembangkan sebuah metode klasifikasi teks berbasis algoritma pembelajaran pohon keputusan C4.5 dalam mendeteksi kerancuan	Identifikasi kerancuan dapat dilakukan secara otomatis menggunakan sebuah perkakas bantu yang dikenal dengan nama	Ruang lingkup dan domain permasalahan dari repositori kata-kata yang rancu masih bersifat terbatas tergantung dari data latih

No	Judul Penelitian	Metode Usulan	Kelebihan	Kekurangan
		dokumen spesifikasi kebutuhan perangkat lunak secara otomatis pada tahap elisitasi.	*Requirements Specification Ambiguity Checker (ReqSAC)*.	yang digunakan.
4	*Detecting ambiguity in requirements specifications* (Bussel, 2009)	Mengusulkan sebuah metode untuk mendeteksi kerancuan secara otomatis dalam dokumen spesifikasi kebutuhan perangkat lunak menggunakan algortima klasifikasi pada mesin pembelajaran yaitu IB1 standar, IB1 *Modified Value Difference Metric* (MVDM) dan *Inverse Learning Weighting*.	Identifikasi kerancuan dapat dilakukan secara otomatis.	Akurasi dan peforma metode sangat bergantung terhadap banyaknya data dan ruang lingkup data latih.
5	*Requirements for Tools for Ambiguity Identification and Measurement in Natural Language Requirements Specifications* (Kiyavitskaya *et al.*, 2008)	Metode ini memperluas penggunaan repositori kata-kata yang rancu agar tidak terbatas hanya pada beberapa ruang lingkup domain dengan membandingkan penggunaan tiga sumber daya kamus kata yang dapat	Repositori kata-kata yang rancu menjadi tidak terbatas hanya pada beberapa ruang lingkup domain permasalahan.	Proses identifikasi kata ambigu yang kurang efesien dimana teknik pencocokan kata ambigu dilakukan dengan membandingkan tiap kata ambigu yang terdapat didalam repositori dengan pernyataan

No	Judul Penelitian	Metode Usulan	Kelebihan	Kekurangan
		diakses secara publik dalam mengidentifikasi kerancuan leksikal yaitu, *WordReference*, *WordNet* dan *Babylon's*.		kebutuhan.
6	Analisis Ambiguitas Kebutuhan Perangkat Lunak Berdasarkan Acuan SMART *Requirements* (Muliawan *et al*., 2011)	Mengusulkan sebuah metode untuk mendeteksi ambiguitas pada pernyataan kebutuhan perangkat lunak menggunakan pemrosesan bahasa alamiah (NLP) dengan menerapkan aturan SMART (*Spesific, Measureable, Attainable, Reliazable, and Trackable*).	Teknik pencocokan pola yang digunakan dapat lebih cepat dan efesien dalam mendeteksi kata yang ambigu dengan membandingkan pola-pola aturan yang telah dibuat.	Pola aturan penyebab rancu tidak begitu banyak yaitu sejumlah 27 pola aturan.

{Halaman ini sengaja dikosongkan}

Dari beberapa penelitian yang telah diuraikan, Kamsties *et al*., (2001) menggunakan pendekatan inspeksi yang dilakukan secara manual dalam mendeteksi kerancuan. Pendekatan ini masih kurang efektif karena analisis dilakukan secara manual dan sangat bergantung terhadap subyektivitas dan kemampuan seorang pakar dalam menganalisis kerancuan. Penelitian berikutnya berfokus dalam mendeteksi kerancuan yang dilakukan secara otomatis seperti yang telah dibahas oleh Lami *et al.*, (2005), Hussain (2007) dan Bussel (2009). Penelitian tersebut mencoba untuk mendeteksi ambiguitas dengan memanfaatkan sebuah repositori yang dibuat khusus untuk memuat daftar kata-kata yang tidak diinginkan. Penelitian Lami *et al.*, (2005), Hussain (2007) dan Bussel (2009) memiliki kekurangan yaitu repositori yang digunakan masih bersifat statis dan ruang lingkup permasalahan juga bersifat terbatas hal ini tentunya akan mempengaruhi kinerja dari peforma metode.

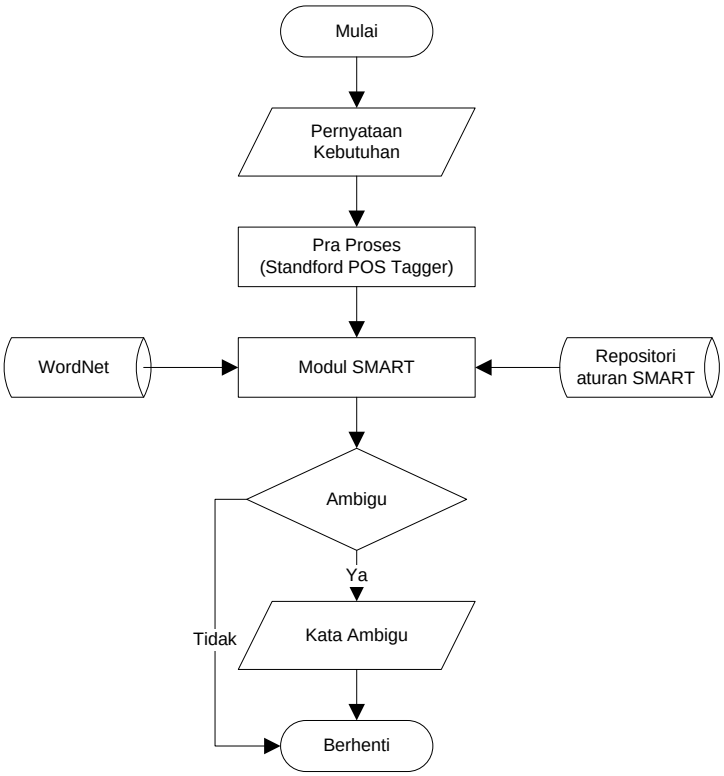
Dalam menyelesaikan permasalahan terbatasnya ruang lingkup domain repositori, peneliti lainnya yaitu Kiyavitskaya *et al.*, (2008) mencoba untuk menggunakan sumber daya kamus kata leksikal seperti *WordReference*, *WordNet* dan *Babylon's* untuk memperluas repositori kata. Berdasarkan penelitian Kiyavitskaya *et al.*, (2008) menunjukkan bahwa penggunaan sumber daya kamus kata *WordNet* lebih efektif jika dibandingkan dengan penggunaan kamus kata lainnya. Namun, penelitian ini memiliki kekurangan yaitu proses identifikasi kata rancu yang kurang efesien dimana teknik pencocokan kata rancu dilakukan dengan membandingkan tiap kata rancu yang terdapat didalam repositori dengan pernyataan kebutuhan. Peneliti selanjutnya Muliawan *et al*., (2011) mengusulkan sebuah metode yang lebih dinamis dalam mendeteksi kerancuan, dimana metode ini menggunakan teknik pemrosesan bahasa alamiah dengan menerapkan aturan SMART dalam mendeteksi kerancuan pada pernyataan kebutuhan perangkat lunak. Kinerja metode ini dibantu dengan kamus kata leksikal *WordNet* untuk memperluas repositori aturan dan bekerja berdasarkan teknik pencocokan pola. Dimana teknik pencocokan pola lebih dapat diandalkan dalam mendeteksi kata yang ambigu dengan membandingkan pola-pola aturan yang telah dibuat.

Pada penelitian ini akan menerapkan metode NLP berbasis aturan SMART dalam mendeteksi kerancuan pada pernyataan kebutuhan yang ditulis

menggunakan bahasa alamiah (Muliawan *et al*., 2011). Kinerja metode ini cukup baik karena dibantu oleh kamus kata leksikal *WordNet* dalam memperluas repositori kata-kata rancu agar tidak terbatas hanya pada beberapa ruang lingkup domain permasalahan. Selain itu metode ini juga dapat bekerja secara efesien dan dinamis dengan mencocokan pola aturan penyebab rancu, dimana jika terdapat kerancuan baru yang tidak terdapat didalam repositori aturan yang telah dibuat pengguna dapat menambahkannya pada tabel pola aturan penyebab rancu. Sehingga memungkinkan penelitian tersebut untuk dikembangkan lagi dengan mengkolaborasikan metode yang diusulkan pada penelitian tersebut dengan mekanisme rekomendasi perbaikan pernyataan kebutuhan yang rancu.

2.4 Metode NLP berbasis aturan SMART

Metode NLP berbasis aturan SMART digunakan untuk mengidentifikasi bagian dari pernyataan kebutuhan yang ambigu dimana arsitektur sistem yang diusulkan Muliawan *et al*., (2011) adalah sebagai berikut :



Gambar 2.1 Arsitektur sistem metode NLP berbasis aturan SMART

Berikut akan dijabarkan penjelasan dari tiap-tiap komponen dalam arsitektur sistem di atas.

Preprocessing

Tahapan awal yang dilakukan ialah melakukan pemrosesan terhadap data input yang berupa pernyataan kebutuhan. *Stanford POS Tagger* digunakan untuk menandai POS setiap kata pada pernyataan kebutuhan agar bisa di proses lebih lanjut oleh Modul SMART.

Repositori Aturan SMART

Repositori aturan SMART dibuat menggunakan *Database Management System* MySQL. Kata-kata yang bersifat rancu beserta POS-nya akan disimpan dalam sebuah tabel yang nantinya digunakan oleh Modul SMART dalam proses pemilihan aturan yang tepat dalam rangka menganalisis kerancuan sebuah pernyataan kebutuhan.

WordNet

WordNet bertugas untuk memberikan rekomendasi berupa sinonim dari kata-kata yang bersifat rancu yang ada di dalam tabel aturan SMART.

Modul SMART

Modul *SMART* adalah modul utama yang bertugas untuk menganalisis kerancuan setiap pernyataan kebutuhan yang telah diproses sebelumnya. Proses analisis dibantu oleh repositori aturan SMART dan *WordNet*. Modul *SMART* memiliki 6 tugas utama, yaitu memilih aturan yang tepat, mendeteksi frasa rancu, mendeteksi kerancuan tanpa bantuan sinonim dari *WordNet*, menanyakan sinonim kata rancu pada *WordNet*, mendeteksi kerancuan dengan bantuan sinonim dari *WordNet*, dan mencetak hasil analisis yang disertai dengan catatan mengapa sebuah pernyataan kebutuhan dinilai rancu dengan menunjukkan bagian kata mana yang bersifat rancu.

2.5 Metode Rekomendasi Perbaikan Kesalahan Tata Bahasa

Setelah kata-kata rancu ditemukan dalam pernyataan kebutuhan perangkat lunak, langkah selanjutnya ialah memberikan rekomendasi perbaikan kata-kata yang teridentifikasi menggunakan sebuah metode rekomendasi yang akan memberikan pilihan perbaikan kata ranu yang disarankan. Adapun untuk memperbaiki kesalahan pada kata-kata rancu, pada penelitian ini menerapkan konsep dasar pengecekan dan perbaikan kesalahan tata bahasa. Beberapa

pembahasan metode yang digunakan untuk memberikan rekomendasi perbaikan pada kesalahan tata bahasa antara lain telah dibahas oleh beberapa peneliti sebelumnya diantaranya ialah Naber (2003) mengembangkan sebuah metode dan kakas bantu yang dapat memeriksa kesalahan tata letak dan tata bahasa dalam kalimat bahasa inggris menggunakan teknik berbasis aturan. Teknik ini mencoba untuk mewakili pengetahuan tata bahasa ke dalam sebuah pola berbasis aturan untuk mengubah kesalahan yang terjadi menjadi sebuah kata yang valid. Proses pemeriksaan kesalahan dilakukan dengan mencocokan semua pola aturan yang ditulis pada sebuah *file* XML dengan hasil ekstraksi pola kalimat masukkan. Jika terdapat sebuah aturan yang cocok, maka kalimat masukkan tersebut memiliki kesalahan pada posisi pola aturan yang ditemukan. Aturan menggambarkan sebagai kesalahan pola kata, penandaan tiap bagian kata dan potongan kata. Setiap aturan juga mencakup sebuah penjelasan kesalahan yang ditunjukkan kepada pengguna dan rekomendasi perbaikan yang diusulkan. Hasil dari kakas bantu yang dibangun dapat digunakan baik sebagai aplikasi pemeriksa tata bahasa yang berdiri sendiri maupun sebagai bagian dari pengolah kata yang terintegrasi.

Untuk meningkatkan peforma metode yang diusulkan, Naber (2003) menggunakan sebuah korpus yang besar yaitu *British National Corpus* untuk memastikan bahwa sistem yang di bangun tidak melaporkan terlalu banyak kesalahan pada kalimat yang benar, sehingga akurasi sistem dapat meningkat. Teknik berbasis aturan yang diusulkan dapat digunakan untuk mengekspresikan aturan yang menggambarkan kesalahan pada tingkat frasa dan tidak hanya pada tingkat kata. Sehingga metode ini menjadi salah satu metode yang dapat diandalkan dalam memeriksa kesalahan tata bahasa (Naber, 2003). Peneliti lainnya yang juga membahas teknik berbasis aturan dalam memeriksa kesalahan tata bahasa yaitu Singh *et al.,* (2016) yang mengembangkan sebuah metode untuk memperbaiki kesalahan ejaan menggunakan teknik prediksi rekomendasi berbasis frekuensi dan pemeriksa kesalahan tata bahasa menggunakan teknik berbasis aturan. Adapun pemeriksa tata bahasa yang diusulkan hanya berfokus pada mendeteksi dan memperbaiki aturan *tenses* yang salah dalam bahasa Inggris.

Henrich dan Reuter (2009) mengusulkan sebuah metode dan kakas bantu untuk memeriksa dan memperbaiki kesalahan tata bahasa yang independen

terhadap jenis bahasa berdasarkan pendekatan statistik. Bahasa yang digunakan sebagai bahasa uji coba penelitian ialah bahasa Inggris dan bahasa Jerman. Kakas bantu yang dihasilkan diberi nama *LISGrammarChecker*. *LISGrammarChecker* bekerja melalui dua tahapan yaitu tahap pembelajaran (*training mode*) dan tahap pemeriksaan (*checking mode*). Pada tahap pembelajaran kakas bantu akan mengekstrak *n-gram* dari kalimat yang benar untuk membangun basis data statistik. Basis data ini digunakan untuk menemukan kesalahan dan mengusulkan koreksi kesalahan pada tahap pemeriksaan. Pada basis data terdapat *n-gram* kata dan hasil penandaan tiap bagian *n-gram* kata yang terdiri dari *bigram*, *trigram*, *quadgram* dan *pentagram*. Untuk mendeteksi kesalahan setiap kalimat dianalisis berkaitan dengan *n-gram* nya. *N-Gram* ini akan dibandingkan dengan basis data. Jika *n-gram* tidak ditemukan pada basis data, maka dapat diasumsikan bahwa kalimat tersebut tidak benar. Untuk setiap titik kesalahan yang salah pada *n-gram* tergantung dari jenis *n-gram* yang digunakan.

Hasil evaluasi menunjukkan bahwa pendekatan Henrich dan Reuter (2009) dapat bekerja untuk bahasa yang berbeda walaupun ketepatan pemeriksaan tata bahasa bervariasi. Alasannya adalah karena perbedaan kekayaan morfologis bahasa. Keandalan data statistik sangat penting, yaitu wajib menyediakan data latih yang cukup dengan kualitas yang baik untuk semua kesalahan tata bahasa. Semakin banyak *tag* yang digunakan pada kumpulan *tagset*, maka semakin banyak pula fitur gramatikal yang dapat diwakili. Metode perbaikan yang diusulkan oleh Henrich dan Reuter (2009) menunjukkan bahwa masih banyak terdapat masalah dalam menemukan semua kesalahan gramatikal. Henrich dan Reuter (2009) mengatasi masalah tersebut dengan mengusulkan sebuah gagasan yang mengkombinasikan dua pendekatan yaitu berbasis statistik dengan aturan (*hybrid*). Teknik *hybrid* akan bekerja dengan sangat baik dalam kasus bahasa yang tergantung pada satu bahasa tertentu (bahasa yang dependen), misalnya sebuah kasus permasalahan yang diselesaikan dibatasi hanya menggunakan bahasa Inggris (Henrich dan Reuter, 2009).

Teknik perbaikan kesalahan tata bahasa menggunakan model statistik *n-gram* juga dibahas oleh Athanaselis *et al.*, (2011). Penelitian tersebut mengusulkan sebuah metode untuk memperbaiki kesalahan tata bahasa dalam

urutan kata menggunakan model bahasa statistik *bigram* dan *trigram* yang diekstrak menggunakan sebuah korpus yang besar yaitu *British National Corpus* (Athanaselis *et al.*, 2011). Metode dievaluasi menggunakan tiga dataset yaitu ***Test of English as a Foreign Language*** (*TOEFL*), *Wall Street Journal* (*WSJ's*) dan *Non native English Corpus*. Hasil uji coba menunjukkan bahwa metode dapat mendeteksi dan memperbaiki kesalahan tata bahasa dalam urutan kata dengan akurasi 95% pada kalimat *TOEFL*, 73% pada kalimat *WSJ's* dan 70% pada kalimat *Non native English Corpus*. Penelitian Athanaselis *et al.*, (2011) menunjukkan hasil yang cukup baik dalam mendeteksi dan memperbaiki kesalahan tata bahasa dalam urutan kata menggunakan model bahasa *bigram* dan *trigram*.

Wu *et al*., (2013) mengusulkan sebuah metode berdasarkan cara kerja mesin terjemahan untuk mengoreksi kesalahan serial gramatikal pada kalimat tertentu dalam penulisan tata bahasa. Metode ini melibatkan dua model terjemahan yang didasarkan kepada *web-scale n-gram*. Model pertama menerjemahkan *trigram* yang mengandung kesalahan preposisi-kata kerja menjadi kata yang benar. Model kedua adalah model *back-off*, model ini digunakan dalam kasus dimana *trigram* tidak ditemukan dalam data latih. Evaluasi terhadap serangkaian kalimat pada korpus latih menunjukkan bahwa metode ini memperbaiki kesalahan serial dengan cukup baik menggunakan model bahasa *trigram*.

Untuk memberikan gambaran yang lebih rinci mengenai kelebihan dan kekurangan dari beberapa metode yang telah dibahas maka dapat ditinjau pada tabel komparasi metode berikut.

Tabel 2.2 Komparasi metode rekomendasi perbaikan kesalahan tata bahasa

No	Judul Penelitian	Metode Usulan	Kelebihan	Kekurangan
1	*A Rule-Based Style and Grammar Checker* (Naber, 2003)	Mengembangkan sebuah metode dan kakas bantu yang dapat memeriksa kesalahan tata letak dan tata bahasa dalam kalimat bahasa Inggris menggunakan teknik berbasis aturan	Teknik berbasis aturan yang diusulkan dapat digunakan untuk mengekspresikan aturan yang menggambarkan kesalahan pada tingkat frasa dan tidak hanya pada tingkat kata	Kompleksitas metode yang cepat meningkat tergantung struktur morfologi dan frasa suatu bahasa (Henrich dan Reuter, 2009)
2	*Frequency based Spell Checking and Grammar Checker* (Singh *et al.*, 2016)	Mengembangkan sebuah metode untuk memperbaiki kesalahan ejaan menggunakan teknik prediksi rekomendasi berbasis frekuensi dan pemeriksa kesalahan tata bahasa menggunakan teknik berbasis aturan.	Metode dapat menangani kesalahan ejaan dan kesalahan tata bahasa dalam bahasa Inggris.	Metode diusulkan hanya berfokus pada mendeteksi dan memperbaiki aturan *tenses* yang salah dalam bahasa Inggris.
3	*LISGrammarChecker: Language Independent Statistical Grammar Checking* (Henrich dan Reuter, 2009)	Mengusulkan sebuah metode dan kakas bantu untuk memeriksa dan memperbaiki kesalahan tata bahasa yang independen terhadap jenis bahasa berdasarkan pendekatan	Metode dapat bekerja untuk bahasa yang berbeda (bahasa Inggris dan bahasa Jerman)	Kuantitas dan kualitas data latih sangat menentukan peforma metode

No	Judul Penelitian	Metode Usulan	Kelebihan	Kekurangan
		statistik.		
4	*A Corpus based Technique for Repairing Ill-Formed Sentences with Word Order Errors using Co-Occurences of N-GRAMS* (Athanaselis *et al.*, 2011)	Mengusulkan sebuah metode untuk memperbaiki kesalahan tata bahasa dalam urutan kata menggunakan model bahasa statistik *bigram* dan *trigram* yang diekstrak menggunakan sebuah korpus yang besar yaitu *British National Corpus.*	Metode dapat bekerja secara efesien dalam mengurangi jumlah permutasi awal tiap pasangan *bigram* dan *trigram* kata. Metode menunjukkan kinerja yang cukup baik dalam memperbaiki kesalahan urutan tata bahasa.	Penelitian hanya berfokus menangani kesalahan urutan tata bahasa.
5	*Correcting Serial Grammatical Errors based on N-grams and Syntax* (Wu *et al*., 2013)	Mengusulkan sebuah metode berdasarkan cara kerja mesin terjemahan untuk mengoreksi kesalahan serial gramatikal pada kalimat tertentu dalam penulisan tata bahasa	Metode dapat memperbaiki kesalahan serial dengan cukup baik menggunakan model bahasa *trigram*.	Penelitian hanya berfokus menangani kesalahan serial tata bahasa.

Adapun mekanisme yang digunakan untuk memberikan rekomendasi perbaikan kata yang rancu, pada penelitian ini menerapkan domain pengetahuan perbaikan kaidah tata bahasa. Berdasarkan beberapa metode yang telah diuraikan teknik pemeriksaan dan perbaikan kesalahan tata bahasa umumnya dapat digolongkan menjadi dua basis pendekatan yang sering digunakan yaitu pendekatan berbasis statitik dan berbasis aturan.

Pendekatan berbasis statistik telah dibahas oleh Henrich dan Reuter (2009) yang mengusulkan sebuah metode dan kakas bantu untuk memeriksa dan memperbaiki kesalahan tata bahasa yang independen terhadap jenis bahasa berdasarkan pendekatan statistik. Bahasa yang digunakan sebagai bahasa uji coba penelitian ialah bahasa Inggris dan bahasa Jerman. Hasil evaluasi menunjukkan bahwa pendekatan Henrich dan Reuter (2009) dapat bekerja untuk bahasa yang berbeda walaupun ketepatan pemeriksaan tata bahasa bervariasi, alasannya adalah karena perbedaan kekayaan morfologis bahasa. Keterbatasan metode ini ialah kuantitas dan kualitas data latih sangat menentukan peforma metode. Metode perbaikan yang diusulkan oleh Henrich dan Reuter (2009) menunjukkan bahwa masih banyak terdapat masalah dalam menemukan semua kesalahan gramatikal. Henrich dan Reuter (2009) mengatasi masalah tersebut dengan mengusulkan sebuah gagasan yang mengkombinasikan dua pendekatan yaitu pendekatan berbasis statistik dan berbasis aturan (*hybrid*).

Teknik perbaikan kesalahan berbasis statistik lainnya juga dibahas oleh Athanaselis *et al.,* (2011) yang mengusulkan sebuah metode dalam memperbaiki kesalahan tata bahasa dalam urutan kata menggunakan model bahasa statistik *bigram* dan *trigram* yang diekstrak menggunakan sebuah korpus yang besar yaitu *British National Corpus.* Penelitian Athanaselis *et al.,* (2011) menunjukkan hasil yang cukup baik dalam mendeteksi dan memperbaiki kesalahan tata bahasa dalam urutan kata menggunakan model bahasa *bigram* dan *trigram*. Wu *et al*., (2013) mengusulkan sebuah metode berdasarkan cara kerja mesin terjemahan untuk mengoreksi kesalahan serial gramatikal pada kalimat tertentu dalam penulisan tata bahasa. Evaluasi terhadap serangkaian kalimat pada korpus latih menunjukkan bahwa metode ini memperbaiki kesalahan serial dengan cukup baik menggunakan model bahasa *trigram*.

Pendekatan berbasis aturan telah dibahas oleh Naber (2003) yang mengembangkan sebuah metode dan kakas bantu yang dapat memeriksa kesalahan tata letak dan tata bahasa dalam kalimat bahasa Inggris menggunakan teknik berbasis aturan. Untuk meningkatkan peforma metode yang diusulkan, Naber (2003) menggunakan sebuah korpus yang besar yaitu *British National Corpus* untuk memastikan bahwa sistem yang dibangun tidak melaporkan terlalu banyak kesalahan pada kalimat yang benar, sehingga akurasi sistem dapat meningkat. Teknik berbasis aturan yang diusulkan dapat digunakan untuk mengekspresikan aturan yang menggambarkan kesalahan pada tingkat frasa dan tidak hanya pada tingkat kata. Sehingga metode ini menjadi salah satu metode yang dapat diandalkan dalam memeriksa kesalahan tata bahasa (Naber, 2003).

Penelitian ini menggunakan teknik berbasis aturan dalam memberikan rekomendasi kata yang rancu (Naber, 2003). Adapun untuk meningkatkan peforma metode pemberian rekomendasi, penelitian ini menggunakan teknik *hybrid* yang digagas oleh Henrich dan Reuter (2009) untuk menutupi keterbatasan dan kelemahan dari teknik berbasis statistik. Teknik *hybrid* mengkombinasikan dua pendekatan yaitu teknik berbasis aturan dan teknik berbasis statistik *n-gram* dalam meningkatkan kinerja metode. Teknik statistik *n-gram* berguna untuk menentukan pilihan kandidat kata yang akan direkomendasikan. Model statistik *n-gram* yang digunakan pada penelitian ini terdiri dari dua model yaitu model bahasa *bigram* dan *trigram* yang menunjukkan kinerja cukup baik dalam memperbaiki kesalahan tata bahasa Athanaselis *et al.*, (2011) dan Wu *et al*., (2013). Sedangkan untuk meningkatkan kinerja model bahasa *n-gram* penelitian ini juga memanfaatkan sebuah korpus yang besar yaitu *British National Corpus* (Wu *et al*., 2013)

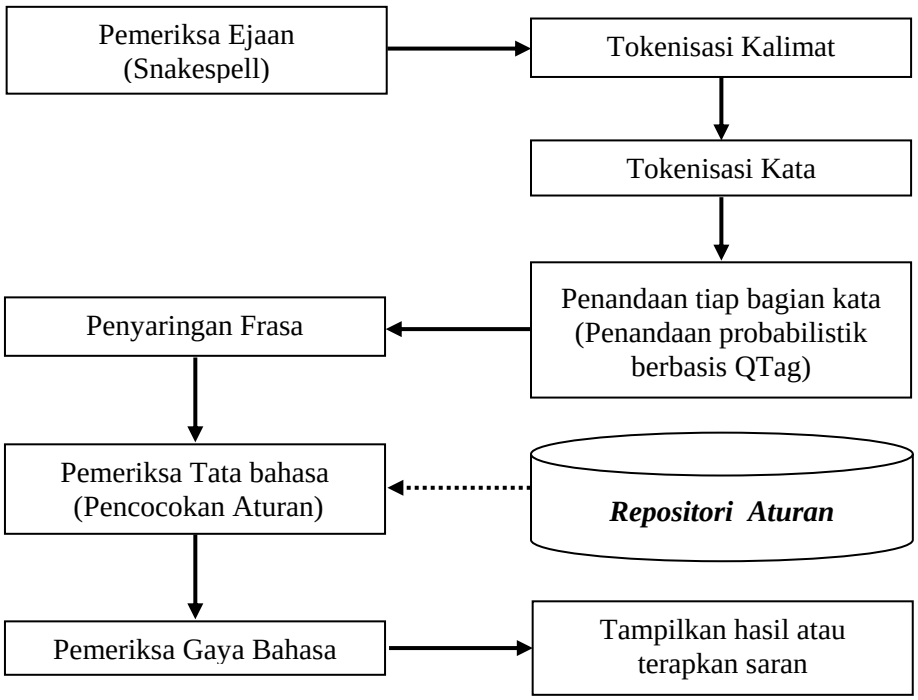
2.6 Teknik Berbasis Aturan

Pendekatan berbasis aturan digunakan untuk pemeriksaan tata bahasa yang melibatkan kegiatan membangun aturan yang dilakukan secara manual dalam mendeteksi kesalahan tata bahasa. Aturan ini kemudian digunakan untuk menemukan kesalahan didalam teks yang telah dianalisis, biasanya ditandai dengan telah dilakukannya pemrosesan tiap bagian kata. Aturan ini sering

mengandung saran bagaimana caranya memperbaiki kesalahan yang ditemukan didalam teks.

Salah satu teknik berbasis aturan yang digunakan untuk pemeriksaan tata bahasa telah diajukan oleh Naber didalam tesisnya yang berjudul pemeriksaan gaya dan tata bahasa berbasis aturan (Naber, 2003). Penelitian tersebut menghasilkan sebuah perkakas bantu bersifat bebas terbuka yang dapat digunakan baik berupa aplikasi yang berdiri sendiri maupun sebagai pemeriksa tata bahasa dan sebagai bagian terpadu dari pengolah kata. Dalam implementasinya sebuah perkakas bantu bahasa juga digunakan dalam membangun sistem. Perkakas bantu ini sangat membantu dalam melakukan beberapa operasi pemrosesan awal pada data masukan dan beberapa proses lainnya seperti pemeriksa ejaan dan tokenisasi kalimat. Pemeriksaan gaya dan tata bahasa yang dibangun akan memproses sebuah teks atau kalimat dan mengembalikan daftar kemungkinan kesalahan yang terjadi beserta rekomendasi perbaikannya.

Untuk memberikan deskripsi yang jelas mengenai arsitektur sistem yang dibangun dapat dilihat pada Gambar 2.2. Tanda panah solid menunjukkan arah aliran dari satu proses ke proses berikutnya dan tanda panah putus-putus menunjukkan sebuah proses membutuhkan data referensi untuk mengeksekusi proses.



Gambar 2.2 Arsitektur sistem yang diajukan oleh Naber (2003)

Pemeriksa Ejaan

Tahap awal ialah memeriksa ejaan tiap kata pada sebuah kalimat menggunakan modul yang tersedia pada kakas bantu bahasa yaitu *Snakespell*. Metode *chek*() akan mengambil teks atau kalimat masukan dan mengembalikan jumlah kesalahan ejaan yang terdapat pada kalimat masukkan tersebut. Kesalahan dapat dikueri dengan metode *getPositions*() yang menyediakan akses ke daftar kesalahan ejaan beserta kemungkinan koreksi yang direkomendasikan yaitu kata-kata yang ditulis mirip dengan kata yang salah.

Misal terdapat satu contoh kalimat "*This job is requering more experiance*" maka dengan mengeksekusi metode *check*() akan mengembalikan nilai jumlah kesalahan ejaan yaitu 2. Metode *getPositions*() dengan parameter 1 maka akan mengembalikan *string* kata "***requering***" dan menampilkan kemungkinan koreksi yaitu "*requiring*".

Tokenisasi Kalimat

Tokenisasi kalimat dilakukan untuk memisahkan bagian kata yang menyusun sebuah kalimat utuh menjadi beberapa bagian kata.

Misalnya terdapat sebuah kalimat "*This job is requiring more experience*" jika dilakukan proses tokenisasi maka akan dikembalikan sebuah daftar bagian kata berikut : ["*This*", "*job*", "*is*", "*requiring*", "*more*", "*experience*"].

Tokenisasi Kata

Tokenisasi kata dilakukan untuk merubah sebuah kata agar menjadi kata dasar pembentuk kata. Misalnya kata "*requiring*", setelah melalui proses tokenisasi kata maka kata dasarnya adalah "*require*".

Penandaan tiap bagian kata (*Part-of-Speech Tagging*)

Penandaan tiap bagian kata dilakukan menggunakan penandaan probabilistik berbasis Qtag dengan ekstensi berbasis aturan. Tujuan dari penandaan ini ialah memberikan keterangan dan nilai probabilitas setiap bagian kata apakah termasuk subjek, kata kerja, preposisi, penentu, kata benda, kata keterangan waktu dan lain-lain berdasarkan aturan yang telah dibuat pada sebuah kamus aturan. Terdapat tiga

kamus kata untuk menandai tiap bagian kata. Kamus kata pertama yang digunakan ialah *British National Corpus* (BNC). Pada kamus ini terdapat sejumlah kata dasar dengan keterangan dan nilai probabilitasnya.

Misalnya kata yang akan ditandai adalah "*walk*" maka didalam kamus kata dasar akan terdapat informasi pemetaan kata berikut *word_to_tag*[*'walk'*] = [(VB, 0.6), (NN, 0.4)], hasil pemetaan ini berarti kata "*walk*" jika dijadikan sebagai kata kerja (VB) nilai probabilitasnya adalah 0.6 dan apabila dijadikan sebagai kata benda (NN) maka nilai probabilitasnya ialah sebesar 0.4.

Kamus kata yang kedua akan memetakan dua urutan kata beserta nilai probabilitasnya. Misalnya terdapat kata yang berurutan berikut "*a good*", maka hasil pemetaan kata adalah seperti berikut *sequence*[(DET, AJ0)] = 0.1. Hasil ini berarti probabilitas sebuah penentu (DET) diikuti oleh kata sifat (AJ0) yaitu sebesar 0.1. Begitu juga dengan kamus kata yang ketiga yang memiliki prinsip kerja yang sama yaitu memetakan dua urutan kata beserta nilai probabilitasnya, akan tetapi aturannya bertolak belakang dari kamus kata yang kedua, dimana aturan yang dihasilkan yaitu kata sifat diikuti oleh penentu (AJ0, DET).

Jika sebuah kata yang akan ditandai tidak terdapat pada kamus kata, maka kata tersebut akan ditandai dengan beberapa aturan sederhana berikut:

1. Jika kata tersebut dimulai dengan huruf besar, maka dianggap sebagai kata benda umum (NN0).
2. Jika kata tersebut diakhiri dengan akhiran *dom, ment, tion, sion, ance, er, or, ist, ness, icity,* maka kata tersebut dianggap sebagai kata benda tunggal (NN1).
3. Jika kata tersebut diakhiri dengan akhiran *ly,* maka kata tersebut dianggap sebagai kata keterangan (AV0).
4. Jika kata tersebut diakhiri dengan akhiran *ive, ic, al, able, y, ous, ful, less,* maka kata tersebut dianggap sebagai kata sifat (AJ0).
5. Jika kata tersebut diakhiri dengan akhiran *ize, ise, ate, fy,* maka kata tersebut dianggap sebagai bentuk infinitif dari kata kerja (VVI).

Berikutnya urutan probabilitas penanda dari kata yang ditandai dan tetangganya akan dihitung. Perhitungan dilakukan dengan cara mengkombinasikan seluruh kemungkinan kata yang ditandai dengan tiga tetangganya yang berurut.

Misalnya terdapat sebuah urutan tiga kata "*like fat food*", langkah awal yang dilakukan ialah mendapatkan nilai probabilitas penanda dari setiap kata, misalkan hasilnya adalah seperti pada tabel berikut:

Tabel 2.3 Nilai probabilitas pada masing-masing penanda

Kata	**Nilai Probabilitas Penanda**
like	PRP = 0.7, VVI = 0.3
fat	NN1 = 0.6, AJ0 = 0.4
food	NN1 = 1

Kemudian semua kombinasi penanda yang mungkin beserta bobotnya akan diseleksi pada kamus aturan penanda, hasilnya dapat dilihat sebagai berikut:

Tabel 2.4 Kombinasi penanda dan bobotnya

Kombinasi Penanda	**Bobot**
PRP + NN1 + NN1	0.1
VVI + NN1 + NN1	0.02
PRP + AJ0 + NN1	0.2
VVI + AJ0 + NN1	0.05

Nilai urutan probabilitas penanda dihitung dengan cara mengalikan nilai probabilitas penanda dengan bobot kombinasi penanda.

P(like/PRP) * P(PRP + NN1 + NN1) = 0.7 * 0.1 = 0.07

P(like/PRP) * P(PRP + AJ0 + NN1) = 0.7 * 0.05 = 0.035

P(like/VVI) * P(VVI + NN1 + NN1) = 0.3 * 0.2 = 0.006

P(like/VVI) * P(VVI + AJ0 + NN1) = 0.3 * 0.2 = 0.06

Dari hasil yang diperoleh maka diambil nilai urutan probabilitas penanda tertinggi. Jika ada kata masukkan berikutnya pada akhir kalimat misalnya kata *Dummy*, maka proses iterasi perhitungan akan terus dilakukan dengan tiga urutan kata yang baru "*fat food Dummy*" (jika kata masukan berada diawal kalimat maka proses perhitungan diabaikan). Nilai urutan probabilitas penanda yang dihasilkan juga akan dibandingkan dengan probabilitas penanda sebelumnya dan dipilih yang tertinggi.

Penyaringan Frasa

Penyaringan frasa diimplementasikan menggunakan teknik berbasis aturan. Aturan yang dihasilkan misalkan seperti berikut :

NP : NN1 NN1

NPP : NN1 NN2

Dimana NP adalah frasa kata benda (NP) dan frasa kata benda jamak (NPP). Biasanya sebuah frasa kata benda dianggap terdiri dari sebuah penentu, satu atau beberapa kata sifat (opsional) dan satu atau lebih kata benda. Dalam contoh aturan ini, apa yang disebut frasa kata benda hanyalah urutan dua kata benda. Aturan kesalahan ini bisa jadi digunakan untuk menemukan kesalahan dalam kalimat berikut :

"*You can measure a **baseball teams** quality by other non subjective means, …"*

baseball teams dikenali sebagai frasa kata benda jamak. Berdasarkan aturan yang ada pada sistem akan mendeteksi aturan *error* "a"_NPP, sehingga penentu jika diikuti oleh frasa kata benda jamak dianggap sebagai kesalahan. Sebenarnya, kesalahan dalam kasus ini adalah apostrof yang hilang, frasa yang benar adalah "*a baseball team's quality*".

Pemeriksa tata bahasa (Pencocokan Aturan)

Proses selanjutnya ialah pencocokan aturan yang tersedia pada repositori aturan. Repositori aturan ditulis menggunakan format *file* XML, dimana setiap aturan juga memuat penjelasan kesalahan. Jika sebuah aturan cocok dengan struktur kata yang dimasukkan, maka kalimat atau teks tersebut berisi kesalahan pada posisi tertentu.

Pemeriksa gaya bahasa

Pemeriksaan gaya bahasa dilakukan untuk membandingkan gaya penulisan kalimat yang tidak baku (*whitespace*) dengan sejumlah kesalahan aturan yang ditemukan pada repositori aturan pemeriksa gaya bahasa. Aturan yang dihasilkan dapat diuraikan pada contoh berikut.

Misalkan terdapat kata ***don't*** dan ***we'll*** pada sejumlah aturan yang salah maka setelah dibandingkan dengan aturan pemerikasa gaya bahasa yang ada maka kata tersebut sebenarnya tidaklah salah dan akan di ganti menjadi ***do not*** dan ***we will***.

Tampikan hasil atau terapkan saran

Setelah melakukan pemeriksaan gaya dan tata bahasa, langkah selanjutnya ialah menampilkan hasil yang diperoleh apakah mengandung kesalahan tata bahasa

ataupun tidak. Jika terdapat kesalahan tata bahasa pada sebuah kalimat maka sistem akan menampilkan penjelasan kesalahan yang terdapat didalam repositori aturan serta saran atau rekomendasi perbaikan kata tersebut.

2.7 Teknik Statistik Berbasis N-Gram

Teknik lainnya yang digunakan untuk memperbaiki kesalahan tata bahasa ialah teknik statistik berbasis *n-gram*. Teknik ini memetakan sebuah kalimat kedalam *n*-huruf sub urutan kata atau *string* dimana *n* biasanya terdiri dari satu kata (*unigram*), dua kata (*bigram*) atau tiga kata (*trigram*) (Wu *et al*., 2013). Model *n-gram* adalah salah satu teknik statistik yang memodelkan urutan kata kedalam nilai probabilitasnya. Jika sebuah kalimat dinyatakan dengan {w_1, w_2, ..., w_{n-1}, w_n} dan *n* adalah jumlah kata dalam kalimat, maka untuk menghitung nilai probabilitas kalimat menggunakan model statistik *unigram* dapat dihitung menggunakan persamaan berikut :

$$P(W_1^n) \approx \prod_i P(w_i) \tag{2.1}$$

Sedangkan untuk menghitung nilai probabilitas tiap pasangan dua kata didalam sebuah kalimat dapat dihitung menggunakan persamaan perkiraan kemungkinan maksimum (*maximum likehood estimation*) berikut :

$$P(w_i|w_{i-1}) = \frac{N(w_{i-1}, w_i)}{N(w_{i-1})} \tag{2.2}$$

Dimana N(w_{i-1}, w_i) dan N(w_{i-1}) menunjukkan jumlah kemunculan kata atau frekuensi N(w_{i-1}, w_i) dan N(w_{i-1}) pada korpus. Nilai probabilitas sebuah kalimat menggunakan model statistik *bigram* dapat dihitung dengan aturan Chain yang ditulis dengan persamaan berikut :

$$P(W_1^n) \approx \prod_{i=1}^{n} P(w_i|w_{i-1}) \tag{2.3}$$

Nilai probabilitas tiap pasangan tiga kata didalam kalimat dapat dihitung dengan menggunakan persamaan kemungkinan maksimum berikut :

$$P(w_i|w_{i-2},w_{i-1}) = \frac{N(w_{i-2}, w_{i-1}, w_i)}{N(w_{i-2},w_{i-1})} \tag{2.4}$$

Dimana N(w_{i-2}, w_{i-1}, w_i) dan N(w_{i-2}, w_{i-1}) menunjukkan jumlah kemunculan kata atau frekuensi N(w_{i-2}, w_{i-1}, w_i) dan N(w_{i-2}, w_{i-1}) pada korpus. Nilai probabilitas sebuah kalimat menggunakan model statistik *trigram* dapat dihitung menggunakan aturan berikut :

$$P(W_1^n) \approx \prod_{i=3}^{n} P(w_i|w_{i-2}, w_{i-1}) \quad (2.5)$$

Untuk melakukan evaluasi dan analisis terhadap model bahasa *n-gram* dengan korpus maka digunakan persamaan *perplexity* (Jurafsky dan Martin, 2017). *Perplexity* memanfaatkan persamaan dasar *cross entropy* dalam mengevaluasi model.

Adapun *cross entropy* dari model bahasa M = P(w_1, w_2, … w_{n-1}, w_n) pada sebuah urutan kata W dapat ditulis dengan persamaan berikut :

$$H(W) = -\frac{1}{n}\log P(w_1, w_2, \ldots w_n) \quad (2.6)$$

Perplexity dari model P pada sebuah urutan kata W dapat didefenisikan sebagai exponensial dari *cross entropy*.

$$\begin{aligned} \text{Perplexity}(W) &= 2^{H(W)} \\ &= P(w_1, w_2, \ldots, w_n)^{-\frac{1}{n}} \\ &= \sqrt[n]{\frac{1}{P(w1, w2, \ldots, wn)}} \end{aligned} \quad (2.7)$$

Berdasarkan aturan Chain yang digunakan untuk menghitung probabilitas kalimat berdasarkan model statistik *bigram* maka *perplexity* dari probabilitas kalimat (W) dapat didefenisikan melalui persamaan berikut :

$$\text{Perplexity}(W) = \sqrt[n]{\prod_{i=1}^{n} \frac{1}{P(w_i|\, w_{i-1})}} \quad (2.8)$$

Perplexity dari probabilitas kalimat (W) untuk model statistik *trigram* dapat didefenisikan sebagai berikut :

$$\text{Perplexity}(W) = \sqrt[n]{\prod_{i=1}^{n} \frac{1}{P(w_i|w_1 \ldots w_{i-1})}} \quad (2.9)$$

Nilai *perplexity* yang minimum sama halnya dengan nilai probabilitas yang maksimum. Sehingga semakin rendah nilai *perplexity* yang diperoleh maka probabilitas kalimat yang dihasilkan akan semakin tinggi.

2.8 Gwet's AC1

Gwet's AC1 adalah sebuah metode pengujian yang digunakan untuk mengukur tingkat kesepakatan antara dua pengamat. Gwet's AC1 menunjukkan pendekatan yang lebih dapat diandalkan jika dibandingkan dengan indeks statistik Cohen Kappa (Gwet, 2002). Hal ini disebabkan karena nilai indeks Cohen Kappa statistik (Cohen, 1984) akan sangat berpengaruh jika nilai kesepakatan antara dua

pengamat sangat rendah. Adapun nilai statistik AC1 dapat dihitung dengan menggunakan tabel hasil observasi pengamat yang ditulis ke dalam bentuk tabel 2x2 seperti yang ditunjukkan pada Tabel 2.5.

Tabel 2.5 Hasil Pengamatan

Pengamat 1	Pengamat 2		
	Ya	Tidak	Total
Ya	A	B	B1 = A + B
Tidak	C	D	B2 = C + D
Total	A1 = A + C	A2 = B + D	N

Pada tabel 2.5 terdapat dua pengamat yang mengelompokkan sejumlah N subjek pengamatan kedalam dua kemungkinan kategori. Dua kategori dilabelkan sebagai 'Ya' dan 'Tidak'. A dikelompokkan oleh dua pengamat sebagai Ya, B dikelompokkan oleh pengamat 1 sebagai Ya dan pengamat 2 sebagai Tidak. C dikelompokkan oleh pengamat 1 sebagai Tidak dan pengamat 2 sebagai Ya. D dikelompokkan oleh pengamat 1 dan 2 sebagai Tidak. A1 dan A2 menunjukkan jumlah subjek yang dikelompokkan pada setiap kategori oleh pengamat 1. Sedangkan B1 dan B2 menunjukkan jumlah subjek yang dikelompokkan pada setiap kategori oleh pengamat 2.

Nilai AC1 statistik dapat dihitung dengan persamaan berikut :

$$AC1 = \frac{P - e(\gamma)}{1 - e(\gamma)} \tag{2.10}$$

Dimana P adalah nilai kesepakatan yang terobservasi dapat dihitung dengan persamaan berikut :

$$P = \frac{A+D}{N} \tag{2.11}$$

Probabilitas *chanche-agreement* atau $e(\gamma)$ dihitung dengan persamaan :

$$e(\gamma)=2P_1 (1- P_1) \tag{2.12}$$

Dimana P_1 merepresentasikan perkiraan kemungkinan seorang pengamat 1 atau 2 mengelompokkan data kedalam kategori 'Ya'. P_1 dapat dihitung dengan persamaan berikut :

$$P_1=\frac{(A1+ B1) / 2}{N} \tag{2.13}$$

Nilai indeks Kappa yang diperoleh berdasarkan pendekatan Gwet's AC1 dapat dikelompokkan menurut rentang nilai pada tabel interpretasi nilai indeks Kappa

(Landis dan Koch, 1977). Tabel 3.3 merupakan tabel interpretasi rentang nilai indeks Kappa yang diperoleh dari hasil pengujian dan proporsi kesepakatan pengamat.

Tabel 2.6 Interpretasi nilai indeks Kappa

Nilai Indeks Kappa	Proporsi Kesepakatan
< 0	Rendah
0.01 – 0.20	Sedikit
0.21 – 0.40	Cukup
0.41 – 0.60	Sedang
0.61 – 0.80	Banyak
0.81- 1.00	Hampir Sempurna

Berikut merupakan contoh perbandingan hasil perhitungan nilai indeks Kappa berdasarkan pendekatan statistik Cohen Kappa dan pendekatan statistik Gwet's AC1 pada sebuah hasil pengamatan yang dilakukan oleh dua orang pengamat terhadap 100 jenis padanan warna halaman *website* apakah memiliki komposisi warna yang baik atau tidak. Hasil pengamatan dapat dilihat pada Tabel 2.6.

Tabel 2.7 Data pengamatan contoh kasus Kappa

Pengamat 1	Pengamat 2		
	Ya	Tidak	Total
Ya	80	10	90
Tidak	5	5	10
Total	85	15	100

$$P = \frac{80+5}{100} = 0.85$$

Jika dihitung menggunakan pendekatan indeks Cohen Kappa statistik maka didapat nilai indeks Kappa sebagai berikut :

$$e(K) = \left(\frac{90}{100}\right) x \left(\frac{85}{100}\right) + \left(\frac{10}{100}\right) + \left(\frac{15}{100}\right) = 0.78$$

$$Kappa = (0.85 - 0.78) / (1-0.78) = 0.318$$

Nilai indeks Cohen Kappa yang dihasilkan pada data pengamatan Tabel 2.6 sangatlah rendah yaitu 0.318, hal ini disebabkan oleh kesepakatan antara dua pengamat yaitu pengamat 1 dan 2 sangat rendah. Sehingga menyebabkan nilai indeks Kappa yang dihasilkan menurun secara signifikan.

Sedangkan jika dihitung menggunakan pendekatan Gwet's AC1 maka nilai indeks Kappa dapat diketahui melalui perhitungan berikut.

$$e(\gamma) = 2\left(\frac{90+85}{2 \times 100}\right)\left(1 - \frac{90+85}{2 \times 100}\right) = 0.21875$$

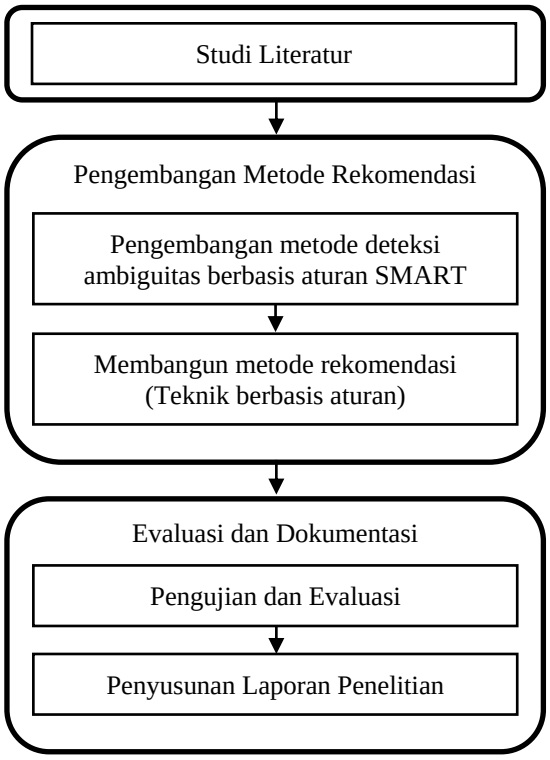
AC1 = (0.85 - 0.21875) / (1 - 0.21875) = 0.808

Nilai indeks Kappa yang diperoleh menggunakan pendekatan statistik Gwet's AC1 yaitu sebesar 0.808, dimana lebih konsisten dan dapat diandalkan jika dibandingkan dengan nilai indeks Cohen Kappa statistik.

BAB III
METODOLOGI PENELITIAN

Penelitian ini telah mengalami beberapa tahapan penelitian yang semuanya terstruktur di dalam prosedur penelitian. Prosedur penelitian dimulai dengan studi literatur, melakukan pengembangan metode untuk mendeteksi ambiguitas berbasis aturan SMART, membangun metode rekomendasi menggunakan teknik berbasis aturan. Kemudian dilanjutkan dengan pengujian dan evaluasi terhadap sistem yang dibangun serta mendokumentasikan hasil pengujian kedalam bentuk laporan.

Metodologi penelitian yang digunakan penulis tersusun dalam sebuah prosedur penelitian seperti dibawah ini. Prosedur ini memperlihatkan tahap-tahap proses penelitian yang dilakukan penulis mulai dari tahap awal sampai akhir kegiatan penelitian, dapat dilihat pada Gambar 3.1.



Gambar 3.1 Prosedur Penelitian

Secara terperinci prosedur penelitian ini terdiri dari beberapa tahap pengerjaan mulai dari studi literatur, pengembangan metode rekomendasi deteksi ambiguitas berbasis aturan SMART, membangun metode rekomendasi

menggunakan teknik berbasis aturan, pengujian dan evaluasi, hingga penyusunan laporan yang dapat dilihat pada uraian berikut ini :

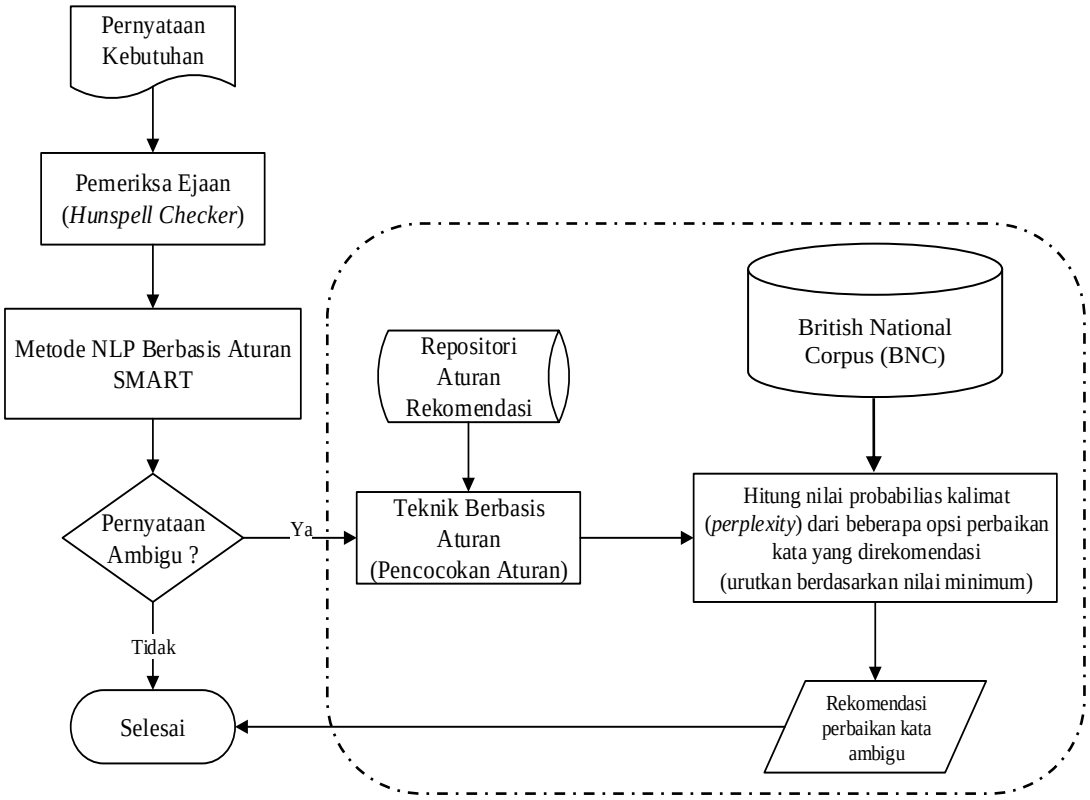
3.1 Studi Literatur

Studi literatur dilakukan dengan mengumpulkan berbagai literatur baik berupa jurnal, buku-buku maupun sumber lainnya yang berkaitan pada pembahasan penelitian. Sehingga dapat dijadikan referensi dan rujukan terhadap penelitian ini. Berdasarkan studi literatur yang telah dilakukan terdapat beberapa informasi yang menjadi acuan penulis dalam melakukan penelitian ini, diantaranya adalah sebagai berikut :

1. Permasalahan yang sering terjadi ketika seorang perekayasa dan penyusun kebutuhan perangkat lunak menggunakan bahasa alamiah ialah kerancuan.
2. Dari beberapa penelitian yang telah dilakukan peneliti sebelumnya hanya berfokus pada metode untuk mendeteksi kerancuan.
3. Belum adanya sebuah sistem rekomendasi perbaikan pada pernyataan kebutuhan yang rancu.
4. Penerapan metode pemeriksaan kesalahan tata bahasa dapat digunakan untuk memberikan pilihan rekomendasi perbaikan pernyataan kebutuhan yang rancu.

3.2 Pengembangan Metode Rekomendasi

Tahapan pengembangan metode rekomendasi meliputi tahap desain dan implementasi sistem yang dilakukan untuk memberikan gambaran umum sistem yang akan dibangun. Pada penelitian ini akan dibangun sebuah perkakas bantu yang akan memberikan rekomendasi perbaikan pernyataan kebutuhan yang rancu. Tahapan desain dan implementasi dilakukan dengan mengembangkan metode NLP berbasis aturan SMART untuk mendeteksi kata-kata rancu pada pernyataan kebutuhan dan membangun sebuah metode rekomendasi perbaikan pernyataan kebutuhan perangkat lunak yang rancu menggunakan teknik berbasis aturan. Adapun mekanisme solusi yang diusulkan pada penelitian ini dapat dilihat pada *flowchart* usulan metode pada Gambar 3.2 berikut.



Gambar 3.2 *Flowchart* usulan metode

Ruang lingkup dan fokus utama permasalahan penelitian ini dapat dilihat pada bagian kotak dengan garis putus-putus pada Gambar 3.2. Proses utama dalam usulan metode rekomendasi perbaikan pernyataan kebutuhan ini terdiri dari 2 proses yaitu proses mengidentifikasi kata yang rancu dengan mengembangkan metode deteksi ambiguitas berbasis aturan SMART dan membangun metode rekomendasi menggunakan teknik berbasis aturan.

3.2.1 Pengembangan Metode Deteksi Ambiguitas Berbasis Aturan SMART

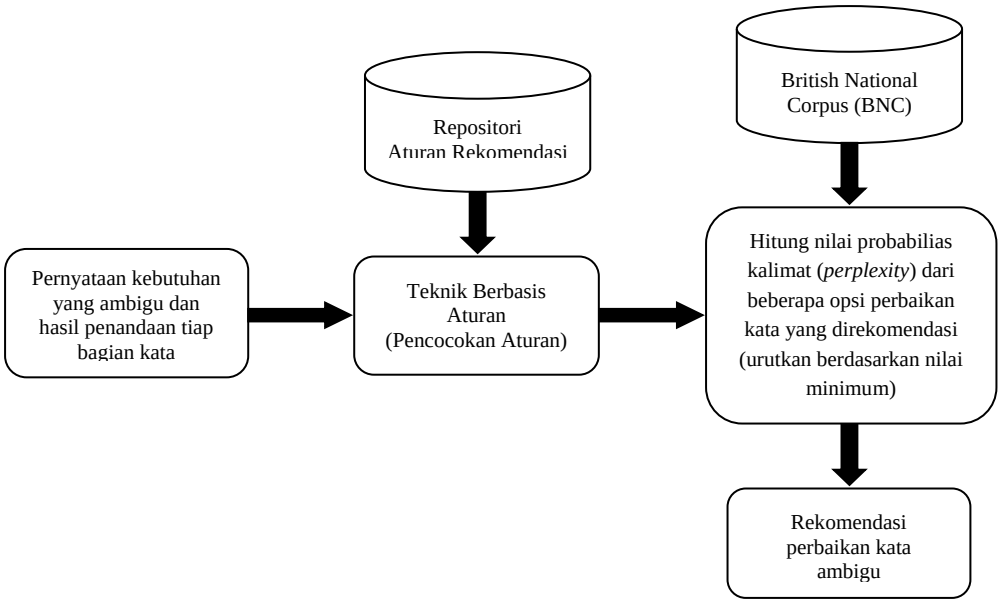
Data yang digunakan sebagai masukkan dari sistem ialah pernyataan kebutuhan perangkat lunak. Kalimat atau pernyataan kebutuhan terlebih dahulu akan dilakukan pemrosesan awal yaitu pemerikasaan kesalahan ejaan dimana modul yang digunakan pada penelitian ini ialah *Hunspell Checker*. Modul ini bersifat terbuka dan telah banyak digunakan pada beberapa aplikasi pengolah kata seperti *LibreOffice* dan *OpenOffice*. Modul ini bekerja menggunakan teknik kesamaan *n-gram*, berbasis aturan dan kamus pengucapan. Hasil dari modul ini ialah daftar kata yang mengandung kesalahan ejaan beserta rekomendasi perbaikannya.

Langkah berikutnya ialah melakukan tokenisasi kalimat dan merubah tiap kata hasil dari proses tokenisasi kalimat menjadi kata dasar. Penyaringan frasa dan pemeriksaan gaya bahasa dilakukan untuk menambah peforma sistem dalam mendeteksi kesalahan tata bahasa. Penandaan tiap bagian kata dilakukan dengan menggunakan pustaka *Stanford POS Tagger*. Selanjutnya pencocokkan aturan yang tepat dilakukan pada kamus aturan yang telah dibuat dalam *file* XML. Kamus aturan kata rancu yang digunakan pada penelitian ini berjumlah 189 aturan, dimana 52 aturan berasal dari aturan yang telah digunakan pada penelitian Muliawan *et. al.*, (2011) dan 137 aturan lainnya berasal dari aturan yang digunakan pada penelitian Tjong (2008) dalam mendeteksi kerancuan pada pernyataan kebutuhan perangkat lunak. Proses analisis dimulai dengan mencocokkan masing-masing kata dalam kalimat dengan kata-kata rancu dalam kamus aturan. Jika terjadi kecocokan salah satu kata dengan kata yang tersimpan dalam kamus aturan, maka kalimat tersebut dinilai rancu. Saat tidak ditemukan kata rancu yang persis sama dengan kata-kata rancu pada kamus aturan, dilakukan penyimpanan POS kalimat yang memiliki POS sama dengan POS kata yang rancu. Disinilah pustaka *WordNet* bekerja untuk mencari sinonim dari kata-kata rancu yang terdapat dalam kamus aturan. Pencarian sinonim hanya dilakukan pada kata yang memiliki POS sama dengan kata yang sedang diperiksa. Apabila sinonim ditemukan, maka kalimat tersebut akan dinyatakan rancu.

Hasil dari proses ini ialah apakah pada pernyataan kebutuhan terdapat kata rancu ataupun tidak. Jika kata rancu teridentifikasi maka proses akan berlanjut pada proses pemberian rekomendasi perbaikan kata rancu, dimana kata rancu yang terdeteksi dan hasil penandaan tiap bagian kata pada kalimat akan dijadikan sebagai input untuk tahap selanjutnya.

3.2.2 Membangun Metode Rekomendasi (Teknik Berbasis Aturan)

Metode rekomendasi yang diusulkan pada penelitian ini menggunakan teknik berbasis aturan, dimana setiap kata yang rancu pada pernyataan kebutuhan akan diberikan rekomendasi perbaikan dengan mengacu pada repositori aturan rekomendasi kata rancu. Adapun usulan metode rekomendasi pada penelitian ini dapat dilihat pada Gambar 3.3.



Gambar 3.3 Usulan metode rekomendasi

Pemberian rekomendasi dilakukan dengan mencocokkan pola penyebab rancu yang terdapat pada sebuah pernyataan kebutuhan dengan kata rancu yang ada pada repositori aturan rekomendasi. Dalam repositori aturan rekomendasi terdapat sejumlah kata rancu dan beberapa kandidat kata yang direkomendasikan untuk mengganti kata rancu. Hasil keluaran dari proses pencocokan aturan adalah daftar kata pengganti kata rancu. Setelah mendapatkan daftar kata-kata rekomendasi yang menjadi kandidat kata yang akan menjadi pengganti dari kata rancu, langkah selanjutnya ialah menghitung nilai padanan dari setiap kata yang berada pada daftar kandidat kata-kata rekomendasi dengan kata sebelum dan sesudah kata yang teridentifikasi sebagai kata rancu. Kemungkinan kata rekomendasi dan kata sebelum dan sesudahnya akan digabung menjadi sebuah kalimat utuh yang akan dihitung nilai probabilitasnya. Hal ini dapat dilakukan dengan menggunakan teknik statistik model bahasa *n-gram*. Model bahasa *n-gram* yang digunakan pada penelitian ini terdiri dari model bahasa *bigram* dan *trigram*. Model bahasa *n-gram* memanfaatkan sebuah korpus BNC untuk menghitung frekuensi dan probabilitas *bigram* dan *trigram*.

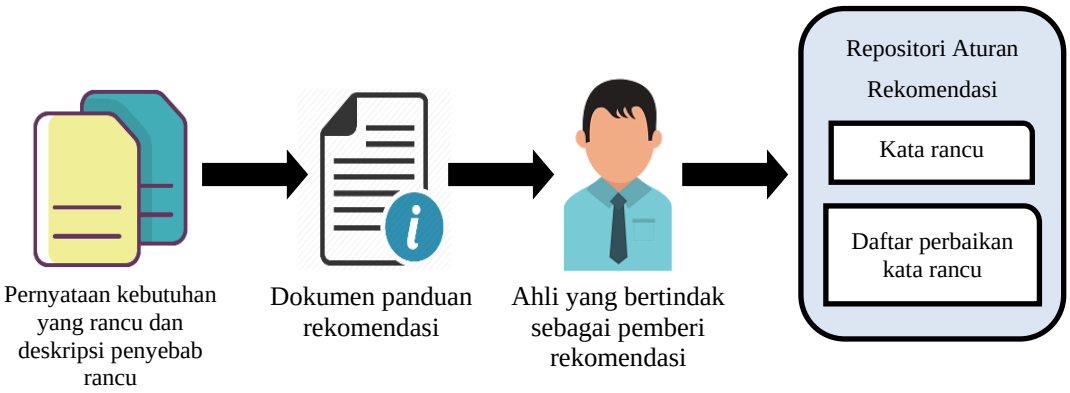
Selanjutnya, nilai *perplexity* dari tiap masing-masing kombinasi kata yang direkomendasi akan dihitung. Nilai *perplexity* yang diperoleh akan diurutkan berdasarkan nilai minimum. Hasil dari proses perhitungan nilai *perplexity* ialah daftar kalimat yang telah diurutkan berdasarkan nilai *perplexity* yang minimum. Kombinasi kata rekomendasi yang memiliki nilai *perplexity* yang minimum

menunjukkan nilai probabilitas kalimat yang tinggi, artinya kalimat tersebut memiliki kemungkinan yang besar dapat diterima sebagai kalimat pengganti kata rancu.

Secara garis besar metode yang diusulkan dalam memberikan rekomendasi perbaikan kata yang rancu terdiri dari tiga tahapan yaitu :

1. Pembuatan repositori aturan rekomendasi.

Kata rancu yang terdeteksi pada sebuah pernyataan kebutuhan akan diperbaiki dengan mencocokan aturan yang terdapat pada sebuah repositori aturan rekomendasi. Dalam repositori aturan rekomendasi terdapat kata-kata rancu beserta beberapa pilihan perbaikannya. Dimana untuk membangun sebuah repositori aturan rekomendasi, pada penelitian ini dibutuhkan 2 orang ahli yang mengusai dan memahami bidang rekayasa kebutuhan maupun bidang rekayasa perangkat lunak khususnya dalam ruang lingkup bahasa Inggris. Ahli akan dibekali dokumen panduan untuk memberikan rekomendasi pernyataan kebutuhan yang rancu, dimana pemberian rekomendasi dari kata yang teridentifikasi sebagai kata yang rancu pada pernyataan kebutuhan dilakukan dengan memberikan beberapa pilihan perbaikan kata rancu yang terdeteksi agar menjadi sebuah pernyataan kebutuhan yang tidak rancu. Gambar 3.4 mengilustrasikan beberapa tahapan yang digunakan dalam pembuatan repositori aturan.



Gambar 3.4 Tahapan pembuatan repositori aturan rekomendasi

Sumber data yang digunakan untuk membangun sebuah repositori aturan rekomendasi ialah beberapa pernyataan kebutuhan yang rancu berasal dari sumber data uji coba beberapa penelitian sebelumnya yang membahas tentang deteksi kerancuan pada dokumen spesifikasi kebutuhan perangkat lunak. Adapun pernyataan kebutuhan yang akan digunakan pada penelitian ini antara lain yaitu

pernyataan kebutuhan pada penelitian Hussain (2007) *ACM's OOPSLA DesignFest®,* spesifikasi kebutuhan *Lift Controller System,* dokumen spesifikasi kebutuhan *Yacht Race Results, Batch Poster System, Cask Loader Software, EVLA Array Operations, Large Area Telescope* (LAT) *Science Analysis Software* (SAS) *Level III Specification, PESA High-Level Trigger Selection, Sort Algorithm Demonstration Program, New Adelaide Airport System* dan *MCSS System.*

Berdasarkan hasil uji coba awal pengumpulan dataset pembuatan repositori aturan rekomendasi yang telah dilakukan menggunakan metode NLP berbasis aturan SMART dalam mendeteksi kerancuan terhadap sumber data yang digunakan, dapat diketahui bahwa jumlah pernyataan yang rancu sebanyak 58 pernyataan kebutuhan pada dataset penelitian Hussain (2007). Selanjutnya, dari 58 sampel pernyataan yang rancu beserta kumpulan kata-kata penyebab rancu yang ada pada aturan SMART akan dikumpulkan ke dalam daftar kata rancu. Daftar kata-kata rancu yang telah dikumpulkan selanjutnya akan di cari kemungkinan solusi atau rekomendasi perbaikan dari kata yang rancu tersebut dengan bantuan dua orang ahli. Ahli akan bertindak sebagai penganalisis kata-kata apa saja yang tepat dan disarankan untuk menggantikan kata rancu tersebut, dimana kemungkinan kata yang direkomendasi bisa saja berjumlah lebih dari satu rekomendasi kata pengganti.

Misalkan terdapat sebuah pernyataan kebutuhan yang rancu berikut :

Pernyataan kebutuhan :

"*The system can be managed by many users*"

Hasil penandaan tiap bagian kata (*POS Tagging*):

The_DT system_NN can_MD be_VB managed_VBN by_IN many_JJ users_NNS ._.

Hasil deskripsi :

Terdapat pola rancu pada pos1 : many_JJ dan pola mengikutinya pos2 : users_NNS di dalam tabel aturan SMART.

Hasil analisis :

Pernyataan kebutuhan rancu.

Kata rancu (kata yang digaris bawahi) :

"*The system can be managed by <u>many</u> users*"

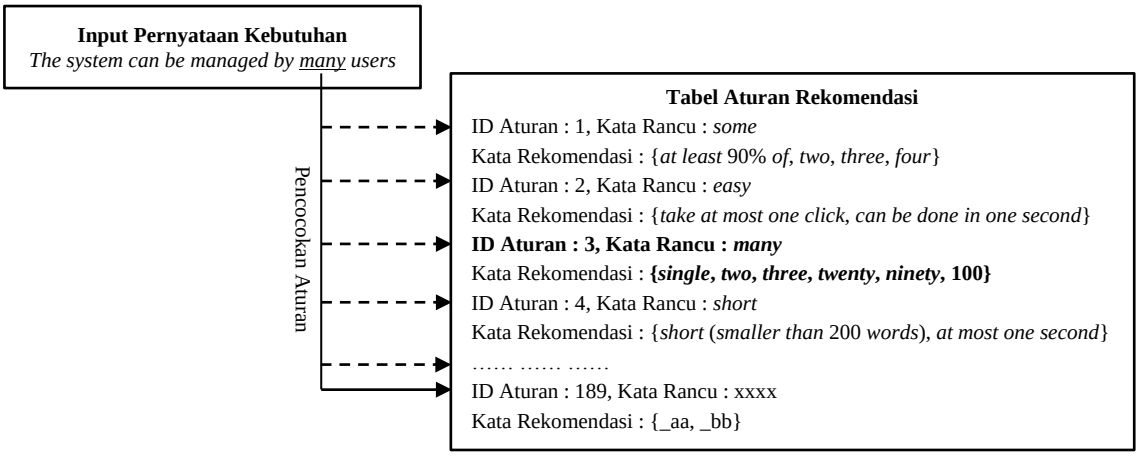
Contoh kandidat kata rekomendasi dari ahli :

one, two, three, five, ten, 100.

Daftar kata rancu beserta kata rekomendasi yang disarankan oleh ahli akan dikumpulkan dan dimasukkan ke dalam tabel aturan rekomendasi, untuk selanjutnya akan digunakan sebagai repositori aturan rekomendasi perbaikan pernyataan kebutuhan yang rancu menggunakan teknik berbasis aturan.

2. Pencocokan aturan rekomendasi menggunakan teknik berbasis aturan.

Pemrosesan awal dataset menghasilkan keluaran berupa daftar kata rancu yang terdeteksi pada sebuah pernyataan kebutuhan. Tahapan selanjutnya ialah mencocokan kata rancu yang terdeteksi dengan daftar aturan rekomendasi yang terdapat pada tabel aturan rekomendasi. Gambaran proses pencocokan aturan rekomendasi dari pernyataan kebutuhan "*The system can be managed by many users*" dengan kata rancu yang ditemukan yaitu kata "*many*" pada tabel aturan rekomendasi dapat dilihat pada Gambar 3.5.



Gambar 3.5 Proses pencocokan aturan rekomendasi

Pada Gambar 3.5 input dari proses pencocokan aturan rekomendasi ialah pernyataan kebutuhan yang rancu. Kata rancu yang ditemukan pada pernyataan kebutuhan akan digunakan sebagai kata kunci pencarian untuk mencari aturan yang cocok pada tabel aturan rekomendasi. Adapun jumlah aturan yang terdapat pada tabel aturan rekomendasi berjumlah 189 aturan yang memuat daftar kata rancu beserta daftar kata rekomendasinya. Selanjutnya, proses pencocokan aturan dilakukan dengan input kata "*many*" sebagai kata kunci pencarian, hasil dari proses pencocokan aturan pada Gambar 3.5 ditemukan salah satu aturan yang cocok pada tabel aturan rekomendasi untuk menggantikan kata rancu "*many*",

yaitu id aturan nomor 3, dengan kata rancu "*many*" dan daftar kata rekomendasinya adalah {*single, two, three, twenty, ninety,* 100}. Hasil dari proses pencocokan aturan rekomendasi ialah mengembalikan daftar kata rekomendasi dari kata rancu yang ditemukan pada pernyataan kebutuhan.

3. Menentukan kandidat kata rekomendasi yang terbaik berdasarkan nilai probabilitas kalimat menggunakan teknik statistik model bahasa *n-gram*.

Misalkan terdapat sebuah pernyataan kebutuhan yang ambigu berikut "*The system can be managed by <u>many</u> users*" dengan mengacu pada tabel aturan rekomendasi maka pilihan perbaikan yang disarankan contohnya menjadi :

- ✓ *The system can be managed by <u>single</u> user*
- ✓ *The system can be managed by <u>two</u> users*

Pada contoh diatas untuk mencari probabilitas tiap kalimat menggunakan model statistik *n-gram* maka diperlukan sebuah kamus kata atau korpus yang berguna untuk mencari frekuensi satu kata (*unigram*), dua kata (*bigram*) dan tiga kata (*trigram*). Korpus yang digunakan pada penelitian ini ialah *British National Corpus* (BNC) yang terdiri dari sekitar 6,25 juta kalimat dan 100 juta kata, dimana dari 100 juta kata terdapat 90% kata-kata bahasa Inggris yang tertulis dan 10% berasal dari konteks percakapan.

Tahap awal untuk mencari probabilitas kalimat ialah mencari frekuensi tiap kata dan nilai probabilitasnya pada korpus. Berikut adalah hasil frekuensi tiap kata dan nilai probabilitasnya pada pilihan perbaikan kata rekomendasi yang pertama.

Tabel 3.1 Frekuensi dan Probabilitas *Unigram*

Kata	the	system	can	be	managed	by	single	user
Frekuensi(N)	6041234	44156	232335	650082	7313	512214	18115	5980
Probabilitas	0,061449	0,00045	0,00236	0,00661	0,00007	0,00521	0,00018	0,00006

Pada Tabel 3.1 frekuensi kemunculan kata '*the'* pada BNC korpus ialah sebanyak 6.041.234 dengan jumlah seluruh kata yang terdapat dikorpus ialah sebanyak 98.313.429 kata. Sehingga nilai probabilitas kata '*the*' dapat dihitung dengan persamaan berikut :

P(*the*) = N(*the*) / Jumlah seluruh kata pada korpus
= 6.041.234 / 98.313.429
= 0,061449

Langkah selanjutnya ialah mencari frekuensi tiap *bigram* pada BNC korpus, dimana hasil frekuensi *bigram* yang diperoleh pada pilihan perbaikan kata rekomendasi pertama dapat dilihat pada tabel frekuensi *bigram* berikut.

Tabel 3.2 Frekuensi *Bigram*

Kata	the	system	can	be	managed	by	single	user
the	0	6930	237	48	33	89	2043	2116
system	361	0	342	11	2	215	0	9
can	826	2	0	54160	0	50	3	0
be	19022	0	15	0	207	370	28	5
managed	85	2	1	0	0	410	0	1
by	130792	7	6	0	2	0	40	9
single	8	30	1	0	0	14	0	40
user	25	3	182	1	1	8	0	0

Pada Tabel 3.2 dapat diketahui bahwa jumlah kemunculan *bigram* pasangan kata '*the*' dan '*system*' pada BNC korpus ialah sebanyak 6930. Tabel frekuensi *bigram* ini nantinya akan berguna untuk menghitung nilai probabilitas *bigram*. Nilai probabilitas *bigram* dapat dihitung menggunakan persamaan perkiraan kemungkinan maksimum. Dimana hasil probabilitas *bigram* yang dihitung menggunakan persamaan perkiraan kemungkinan maksimum dapat dilihat pada Tabel 3.3 berikut.

Tabel 3.3 Probabilitas *Bigram*

Kata	the	system	can	be	managed	by	single	user
the	0	0,00115	0,00004	0,00001	0,00001	0,00001	0,00034	0,00035
system	0,00818	0	0,00775	0,00025	0,00005	0,00487	0	0,00020
can	0,00356	0,00001	0	0,23311	0	0,00022	0,00001	0
be	0,02926	0	0,00002	0	0,00032	0,00057	0,00004	0,00001
managed	0,01162	0,00027	0,00014	0	0	0,05606	0	0,00014
by	0,25535	0,00001	0,00001	0	0,000004	0	0,00008	0,00002
single	0,00044	0,00166	0,00006	0	0	0,00077	0	0,00221
user	0,00418	0,00050	0,03043	0,00017	0,00017	0,00134	0	0

Pada Tabel 3.3 probabilitas *bigram* dari pasangan kata '*the*' dan '*system*' dapat dihitung menggunakan persamaan perkiraan kemungkinan maksimum berikut :

$$P(system|the) = \frac{N(the,\ system)}{N(the)}$$

$$= \frac{6930}{6041234}$$

$$= 0{,}00115$$

Mengacu pada tabel probabilitas *unigram* dan *bigram* yang telah diperoleh maka probabilitas kalimat "*The system can be managed by single user*" pada pilihan perbaikan rekomendasi yang pertama dapat dihitung menggunakan aturan Chain berikut.

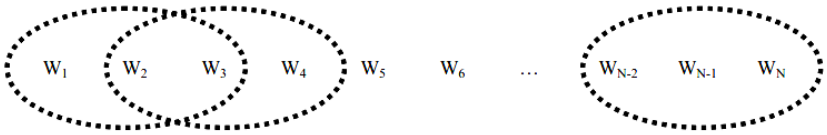
P(*The system can be managed by single user*)

= P(*the*) * P(*system*|*the*) * P(*can*|*system*) * P(*be*|*can*) * P(*managed*|*be*) * P(*by*|*managed*) * P(*single*|*by*) * P(*user*|*single*) * P(*user*)

= 0,061449 * 0,00115 * 0,00775 * 0,23311 * 0,00032 * 0,05606 * 0,00008 * 0,00221 * 0,00006

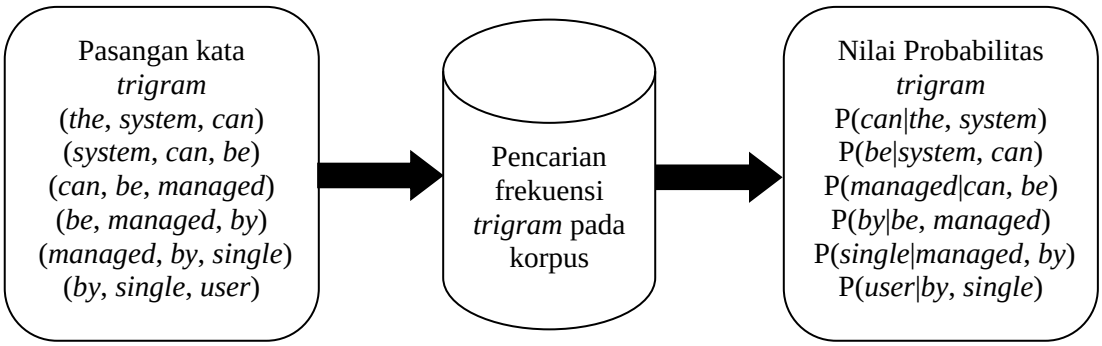
= 2,38303 x 10^{-23}

Tahapan selanjutnya ialah mencari nilai probabilitas kalimat yang direkomendasikan menggunakan model statistik *trigram*. Konsep perhitungan nilai probabilitas *trigram* dapat dilakukan dengan mencari nilai probabilitas tiga kata yang berurut dari kata pertama, kedua dan ketiga (w_1, w_2, w_3) bergeser satu kata ke kanan hingga kata ke-*n* jumlah kata dikurangi dua (w_{n-2}).



Gambar 3.6 Pergeseran jendela *trigram*

Merujuk pada Gambar 3.6 maka proses pencarian menggunakan metode pergeseran jendela *trigram* dapat dilihat pada Gambar 3.7.



Gambar 3.7 Proses pencarian frekuensi dan probabilitas *trigram*

Proses pencarian dimulai dari menemukan kemungkinan pasangan tiga kata yang berurut, kemudian mencocokan tiga pola pasangan kata tersebut dengan pola yang ada pada korpus dan dilanjutkan dengan menghitung nilai probabilitas kalimat menggunakan aturan Chain.

Setelah mendapatkan nilai probabilitas tiap *bigram* dan *trigram* kalimat, langkah selanjutnya ialah menghitung nilai *perplexity* kalimat. Berikut adalah hasil perhitungan *perplexity* dari model statistik *bigram* pada pilihan perbaikan kalimat rekomendasi yang pertama atau W1 "*The system can be managed by single user*".

$$\textit{Perplexity}\ (W1) = \sqrt[n]{\prod_{i=1}^{n} \frac{1}{P(w_i|\, w_{i-1})}}$$

$$= \sqrt[8]{\begin{matrix} 1/P(\text{the})* 1/P(\text{system}|\text{the})* 1/P(\text{can}|\text{system})* \\ 1/P(\text{be}|\text{can}) * 1/P(\text{managed}|\text{be})* 1/P(\text{by}|\text{managed})* \\ 1/P(\text{single}|\text{by}) * 1/P(\text{user}|\text{single}) * 1/P(\text{user}) \end{matrix}}$$

$$= \sqrt[8]{\begin{matrix} 1/0{,}061449* 1/0{,}00115* 1/0{,}00775* \\ 1/0{,}23311 * 1/0{,}00032* 1/0{,}05606* \\ 1/0{,}00008 * 1/0{,}00221 * 1/0{,}00006 \end{matrix}}$$

$$= 672.75785$$

Nilai *perplexity* dari setiap masing-masing kandidat kata rekomendasi selanjutnya akan diurutkan dari nilai terkecil ke terbesar dan menampilkan kata yang memiliki nilai minimum untuk memberikan daftar urutan pilihan perubahan kata yang rancu.

3.3 Pengujian dan Evaluasi

3.3.1 Dataset

Sumber data pengujian pada penelitian ini berasal dari beberapa pernyataan kebutuhan yang digunakan oleh beberapa peneliti di bidang rekayasa kebutuhan khususnya dalam membahas tentang kerancuan pada dokumen spesifikasi kebutuhan perangkat lunak. Hussain (2007) mengumpulkan dataset penelitian yang dipilih secara acak dari 25 domain permasalahan yang berasal dari beberapa pernyataan kebutuhan *ACM's OOPSLA DesignFest*® sumber *online* (*http://designfest.acm.org/*). Selain itu Hussain (2007) juga menambahkan beberapa data yang berada diluar 25 domain permasalahan sebagai dataset. Setelah dilakukan pemrosesan awal (pemeriksaan ejaan dan pemeriksaan

kerancuan) pada 90 pernyataan kebutuhan Hussain (2007) maka diperoleh 58 pernyataan kebutuhan yang rancu pada dataset. Selanjutnya 58 pernyataan kebutuhan ini akan digunakan sebagai data input pada penelitian ini.

Penelitian ini juga menggunakan beberapa pernyataan kebutuhan yang rancu pada penelitian Tjong (2008). Tjong (2008) mengumpulkan beberapa dataset pernyataan kebutuhan yang berasal dari beberapa domain permasalahan yang berbeda untuk melakukan pengujian terhadap penelitiannya dalam mendeteksi kerancuan pada dokumen spesifikasi kebutuhan perangkat lunak. Berikut adalah dataset yang digunakan oleh Tjong (2008).

1) Studi kasus spesifikasi kebutuhan : *Lift Controller System*

   Sistem pengontrol *lift* bertanggung jawab dalam mengelola dan memantau aktivitas *lift* seperti mengirim informasi *lift*, merespon panggilan, mengarahkan *lift*, mengatur waktu tunggu, mengatur waktu operasi dan persyaratan keselamatan beban *lift* untuk memastikan tidak terjadi kecelakaaan.

2) Studi kasus spesifikasi kebutuhan : program *Yacht Race Results* (YRR)

   Sistem YRR bertanggung jawab untuk mendokumentasikan rincian masing-masing kapal dan masing-masing catatan balapan, dan menghasilkan laporan.

3) Spesifikasi kebutuhan persyaratan bisnis terperinci : *Batch Poster System* (BPS)

   Sistem BPS bertanggung jawab untuk memasukkan data inventaris, kontrol akses pengguna, data perawatan, penanganan laporan, pembersihan basis data otomatis, dan antarmuka dokumen.

4) Spesifikasi dokumen kebutuhan umum : *Cask Loader Software* (CLS)

   Sistem CLS bertujuan untuk membantu utilitas nuklir dalam tugas memasukkan bahan bakar bekas ke tong untuk penyimpanan permanen. CLS juga memungkinkan mengoptimalkan beban radiasi dan pemuatan panas dalam batas aturan tertentu.

5) Spesifikasi kebutuhan : *Data Cycle System* (DCS)

   DCS adalah spesifikasi kebutuhan untuk proyek *Stratospheric Observatory for Infrared Astronomy* (SOFIA) yang menyediakan sebuah

kerangka kerja yang seragam, dapat diperluas dan dapat diandalkan untuk semua aspek instrumen ilmiah SOFIA pada saat ini dan yang akan datang.

6) Spesifikasi kebutuhan : *EVLA Array Operations* (EVLA)

   EVLA *Array Operations* adalah spesifikasi kebutuhan yang merupakan bagian dari proyek karya ilmiah EVLA yang bekerja dengan memastikan keamanan peralatan dan orang yang bekerja pada peralatan maupun membantu kelompok teknis dalam pemeliharaan kinerja instrumen yang maksimal.

7) Spesifikasi kebutuhan *: Large Area Telescope* (LAT) *Science Analysis Software* (SAS) *Level III Specification*

   Sistem LAT SAS menyediakan utilitas dasar untuk manipulasi data dan visualisasi.

8) Spesifikasi kebutuhan : *PESA High-Level Trigger Selection*

   Sistem PESA menyiapkan data yang akan dijalankan pada lingkungan *offline* dan *online* untuk kebutuhan pengembangan, pengujian, integrasi, rekonfigurasi, verifikasi, validasi, dan optimasi.

9) Spesifikasi kebutuhan : *Sort Algorithm Demonstration Program* (SAD)

   Program SAD bertujuan untuk mengajarkan bagaimana berbagai algoritma pengurutan beroperasi dan menghemat *startup* dan waktu pelatihan staff baru.

10) Spesifikasi kebutuhan : *New Adelaide Airport System*

    Terdiri dari 68 pernyataan kebutuhan dalam proyek terminal pesawat udara *New Adelaide*.

11) Spesifikasi kebutuhan : *MCSS System*

    Terdiri dari 256 pernyataan kebutuhan dalam proyek *MCSS system*.

3.3.2 Pengujian

Pada tahapan ini akan dilakukan pengujian terhadap metode yang diusulkan dengan melakukan uji coba terhadap hasil keluaran dari kakas bantu dan analisis ahli. Hal ini bertujuan untuk membuktikan apakah metode yang diusulkan memberikan rekomendasi perbaikan yang akurat. Uji coba dilakukan terhadap beberapa pernyataan kebutuhan perangkat lunak yang rancu.

Adapun skenario uji coba yang dilakukan pada penelitian ini dapat diuraikan melalui beberapa tahapan berikut.

1) Memilih pernyataan kebutuhan yang ada pada dataset yang akan dijadikan sebagai data uji coba.
2) Melakukan proses uji coba metode rekomendasi yang diusulkan menggunakan dua model bahasa yang digunakan pada penelitian ini yaitu model bahasa *bigram* dan *trigram* pada setiap data uji coba yang ada pada dataset. Hal ini dilakukan untuk mendapatkan hasil keluaran metode rekomendasi yang diusulkan.
3) Melakukan proses uji coba metode rekomendasi menggunakan teknik berbasis statistik *n-gram* (Henrich dan Reuter, 2009) untuk mendapatkan hasil keluaran metode rekomendasi sebagai komparasi terhadap metode yang diusulkan.
4) Menganalisa hasil keluaran dari kedua metode rekomendasi dengan bantuan seorang pakar yang mempunyai latar belakang akademik dibidang rekayasa kebutuhan perangkat lunak, memiliki pengalaman membangun dokumen SKPL pada proyek perangkat lunak dan pernah menyelesaikan pendidikan formal yang diselenggarakan dalam bahasa Inggris.
5) Menghitung nilai reliabilitas antara hasil keluaran metode rekomendasi dan hasil analisis pakar dengan cara menghitung nilai indeks Kappa dari hasil keluaran metode rekomendasi menggunakan indeks statistik *Gwet's* AC1.
6) Membandingkan nilai indeks Kappa yang dihasilkan dari metode rekomendasi yang diusulkan dan metode berbasis statistik *n-gram* (Henrich dan Reuter, 2009).
7) Memberikan kesimpulan terhadap hasil uji coba.

3.3.3 Evaluasi

Hasil uji coba akan dikoreksi oleh seorang ahli dibidang rekayasa kebutuhan maupun dibidang rekayasa perangkat lunak. Sedangkan untuk mengetahui kinerja metode yang diusulkan dapat dilakukan dengan cara membandingkan pengetahuan yang dimiliki oleh ahli dengan keluaran yang dihasilkan oleh kakas bantu dengan cara menghitung nilai reliabilitas metode yang diusulkan. Ahli

bertindak sebagai penguji pertama, sedangkan kakas bantu bertindak sebagai penguji kedua. Penelitian ini menggunakan teknik statistik Gwet's AC1 untuk mengukur nilai reliabilitas antara kakas bantu dan ahli. Selanjutnya untuk mengevaluasi peforma metode yang diusulkan, maka nilai realibilitas dari metode rekomendasi yang diusulkan akan dibandingkan dengan nilai realibilitas metode rekomendasi berbasis statistik *n-gram* (Henrich dan Reuter, 2009) untuk mendapatkan hasil perbandingan metode.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Implementasi

Untuk mempermudah proses uji coba dan evaluasi terhadap metode yang diusulkan, maka sebuah kakas bantu (*tool*) dirancang untuk mengimplementasikan metode rekomendasi perbaikan pernyataan kebutuhan yang rancu. Kakas bantu di bangun dalam lingkungan bahasa pemrograman Java yang diberi nama *SMART Requirement Suggestion Application* atau disingkat dengan SRSApp.

4.1.1 Pengumpulan Dataset

Dataset yang digunakan pada penelitian ini terdiri dari beberapa pernyataan kebutuhan yang rancu yang dikumpul dari beberapa penelitian sebelumnya yang membahas tentang kerancuan pada dokumen spesifikasi kebutuhan perangkat lunak. Dataset diambil dari berbagai domain permasalahan yang berbeda, dimana hal ini diharapkan dapat mengenali berbagai jenis kerancuan yang terjadi pada beberapa domain spesifikasi kebutuhan perangkat lunak. Adapun 12 dataset yang digunakan pada penelitian ini dapat dilihat pada Tabel 4.1 berikut.

Tabel 4.1 Dataset Penelitian

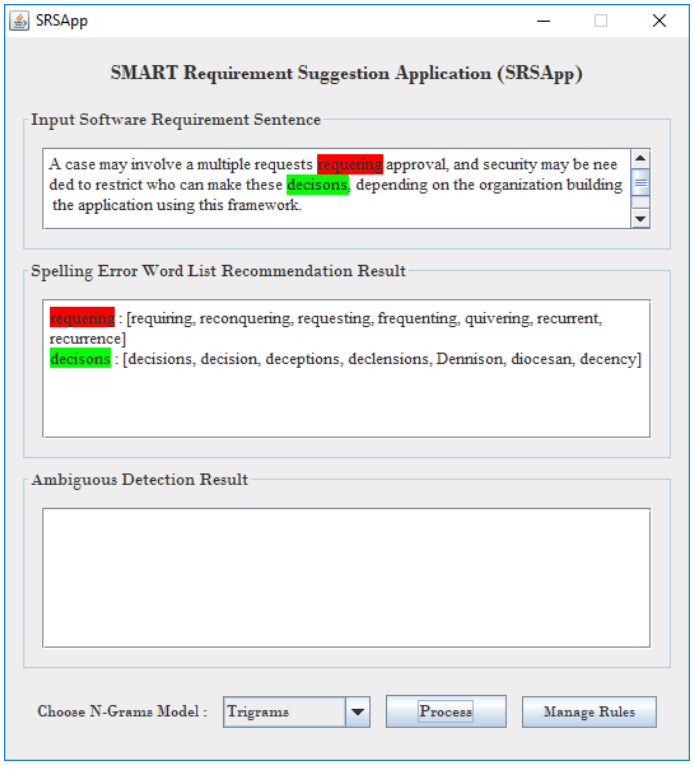
No	Dataset	Sumber
1	*ACM's OOPSLA DesignFest®*	(Hussain, 2007)
2	*Lift Controller System*	(Bray, I.K, 2002)
3	*Yacht Race Results* (YRR)	(Bray, I.K, 2002)
4	*Batch Poster System* (BPS)	(BPS, 2005)
5	*Cask Loader Software* (CLS)	http://www.epri.com/eprisoftware/processguide/docs/srdexdoc.doc.
6	*Data Cycle System* (DCS)	http://www.astro.ucla.edu/~shuping/SOFIA/Documents/DCS_SRD_Rev1.pdf
7	*EVLA Array Operations* (EVLA)	http://www.aoc.nrao.edu/evla/techdocs/computer/workdocs/array-sw-rqmts.%pdf
8	*Large Area Telescope* (LAT) *Science Analysis Software* (SAS) *Level III Specification*	http://www.last.slac.stanford.edu/IntegrationTest/DataHandling/docs/LA%T-SS-00020-6.pdf
9	*PESA High-Level Trigger Selection*	http://www.pp.rhul.ac.uk/atlas/newsw/requirements/1.0.2/

No	Dataset	Sumber
10	*Sort Algorithm Demonstration Program* (SAD)	(Stevenson et al., 2005)
11	*New Adelaide Airport System*	(Tjong, 2008)
12	*MCSS System*	(Tjong, 2008)

Jumlah pernyataan kebutuhan perangkat lunak yang dikumpulkan dari 12 dataset yaitu berjumlah 647 pernyataan kebutuhan baik ambigu maupun tidak ambigu. Untuk selanjutnya dataset yang telah dikumpulkan akan dijadikan sebagai input dari tahap pemrosesan awal dataset.

4.1.2 Pemrosesan Awal Dataset

Setelah dataset dikumpulkan langkah selanjutnya pernyataan kebutuhan akan melalui pemrosesan awal yaitu pemeriksaan kesalahan ejaan dan pendeteksian kerancuan. Dimana pernyataan kebutuhan yang akan dijadikan masukan untuk tahap pengujian ialah pernyataan kebutuhan yang tidak memiliki kesalahan ejaan dan merupakan pernyataan kebutuhan yang rancu. Berikut adalah implementasi sistem dalam melakukan pemrosesan awal yaitu pemeriksaan ejaan.

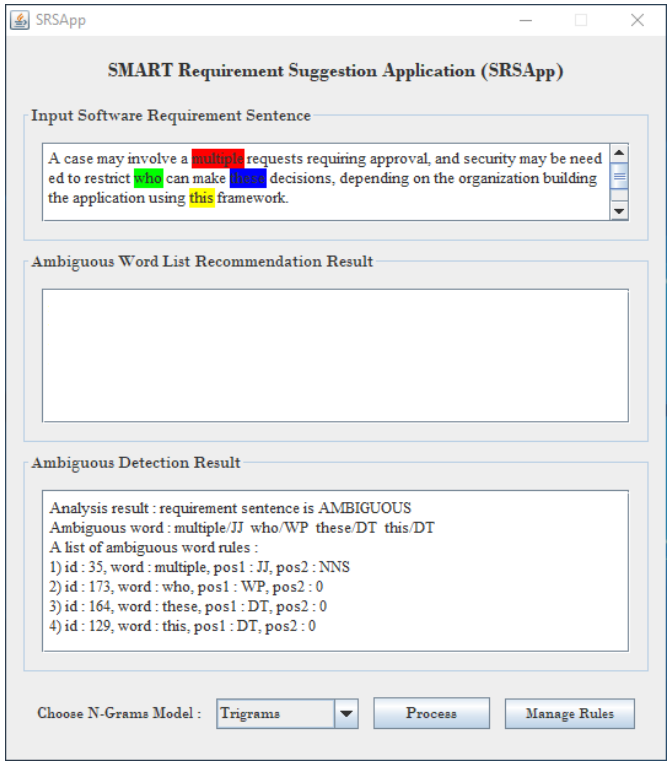


Gambar 4.1 Implementasi pemeriksaan ejaan pada pernyataan kebutuhan

Pernyataan kebutuhan yang dimasukkan pada Gambar 4.1 terdapat kata yang salah eja yaitu kata "*requiring*" dan "*decisions*" (bagian kata yang diberi

warna atau *highlight* pada kotak panel *Input Software Requirement Sentence*). Sistem tidak akan melanjutkan proses berikutnya yaitu mendeteksi kata yang rancu pada pernyataan kebutuhan sebelum kata yang salah ejaan diperbaiki atau kata tersebut dimasukkan kedalam daftar pengecualian dari kata yang salah ejaan. Perbaikan kata yang salah ejaan dapat dilakukan dengan mengetuk salah satu bagian kata yang ada pada daftar kata hasil rekomendasi (kata-kata yang tidak diberi warna atau *highlight* pada kotak panel "*Spelling Error Word List Recommendation Result*"), sedangkan apabila kata yang terdeteksi sebagai kata yang salah ejaan mau dimasukkan kedalam daftar kata pengecualian dapat dilakukan dengan cara mengetuk kata yang diberi warna (*highlight*) pada kotak panel "*Spelling Error Word List Recommendation Result*". Setelah pilihan perbaikan kata yang salah ejaan dipilih maka kata yang salah ejaan pada pernyataan kebutuhan akan berubah sesuai pilihan perbaikan kata yang dipilih.

Langkah pemrosesan awal berikutnya ialah memeriksa kerancuan pada pernyataan kebutuhan. Jika terdapat kerancuan pada pernyataan kebutuhan maka sistem akan menampilkan hasil deteksi pada kotak panel "*Ambiguous Detection Result*" seperti pada Gambar 4.2 berikut.



Gambar 4.2 Hasil deteksi ambiguitas pada pernyataan kebutuhan

Hasil analisis deteksi kerancuan pernyataan kebutuhan "*A case may involve a multiple requests requiring approval, and security may be needed to restrict who can make these decisions, depending on the organization building the application using this framework.*" pada Gambar 4.2 terdapat empat kata yang berpotensi sebagai kata rancu pada kotak panel "*Ambiguous Detection Result*" yaitu kata *multiple*, *who*, *these* dan *this*. Pernyataan kebutuhan dinilai rancu dikarenakan interpretasi pembaca bisa beragam dalam memaknai pernyataan kebutuhan tersebut, misal kata "***multiple** requests*" atau dalam bahasa Indonesia dapat diartikan sebagai beberapa permintaan, dimana jumlah permintaan yang dimaksud pada pernyataan kebutuhan tidak ditentukan sehingga setiap orang mungkin bisa memaknai lebih dari satu permintaan, bisa dua, tiga, empat atau lebih permintaan. Begitu juga halnya dengan kata "*restrict **who** can make these decisions*" atau dapat diartikan sebagai membatasi siapa yang bisa membuat keputusan ini, dalam konteks kalimat ini kata siapa tidak ditentukan aktornya sehingga setiap orang mungkin bebas menyebut siapa itu sebagai pengguna, manajer, pengelola, pengembang dan lain-lain. Kata "***these** decision*" atau bisa diartikan sebagai keputusan ini, kata keputusan ini pada pernyataan kebutuhan dinilai rancu karena tidak mendefenisikan keputusan yang lebih spesifik, sehingga setiap orang dapat memaknai lebih dari satu pemahaman. Kata "***this** framework*" atau bisa diartikan kerangka kerja ini dinilai sebagai kata yang rancu karena tidak menjelaskan kerangka kerja yang lebih spesifik.

4.1.3 Repositori Aturan Rekomendasi

Repositori aturan rekomendasi dibuat berdasarkan hasil pengumpulan data rekomendasi pernyataan kebutuhan dari tim ahli. Dataset yang digunakan untuk membuat respositori aturan rekomendasi berjumlah 12 dataset merujuk pada Tabel 4.1, dimana jumlah data pernyataan kebutuhan yang akan dijadikan sebagai dataset pembuatan repositori aturan rekomendasi adalah sebanyak 96 data pernyataan kebutuhan. Karakteristik dataset yang digunakan untuk membangun repositori aturan rekomendasi diupayakan dapat mencakup seluruh aturan yang ada pada kamus aturan kata rancu pada sistem deteksi kerancuan. Beberapa sampel pernyataan kebutuhan akan dipilih secara acak dari 12 dataset yang ada agar dapat mencakup seluruh aturan yang ada pada kamus aturan kata rancu.

Berikut adalah contoh hasil rekomendasi dari tim ahli (*anotator*) dalam memberikan rekomendasi perbaikan pernyataan kebutuhan yang ambigu.

KUISIONER PEMBERIAN REKOMENDASI

Tabel pernyataan kebutuhan yang rancu dan rekomendasi perbaikan

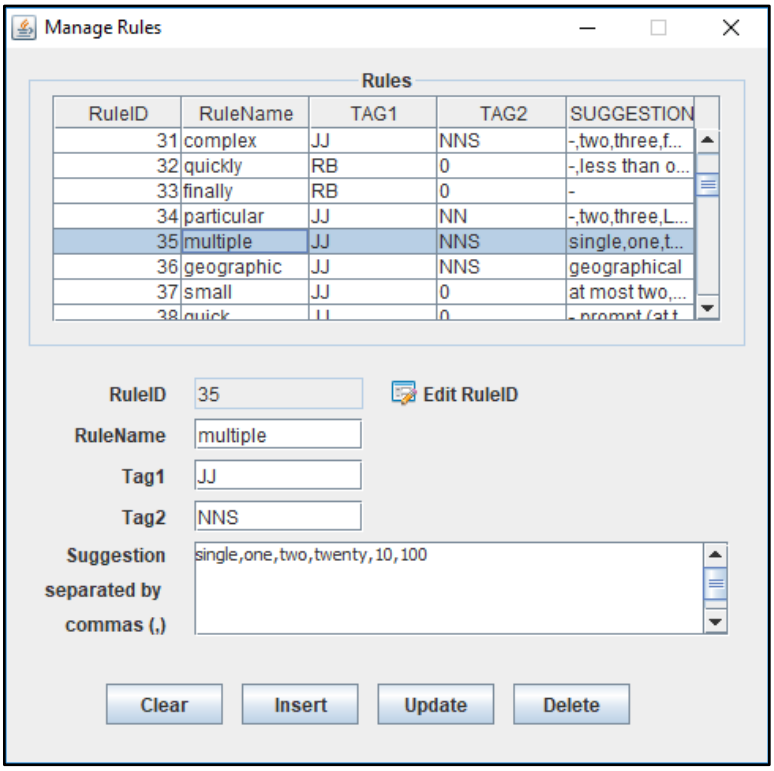
4	Pernyataan kebutuhan	A case may involve a multiple requests requiring approval, and security may be needed to restrict who can make these decisions, depending on the organization building the application using this framework.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	A_DT case_NN may_MD involve_VB a_DT multiple_JJ requests_NNS requiring_VBG approval_NN ,_, and_CC security_NN may_MD be_VB needed_VBN to_TO restrict_VB who_WP can_MD make_VB these_DT decisions_NNS ,_, depending_VBG on_IN the_DT organization_NN building_VBG the_DT application_NN using_VBG this_DT framework_NN ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) multiple_JJ → requests_NNS (aturan 35) 2) who_WP → can_MD (aturan 173) 3) these_DT → decisions_NNS (aturan 164) 1) this_DT → framework_NN (aturan 129)
	Kata rancu (kata yang berwarna merah)	A case may involve a multiple[1] requests requiring approval, and security may be needed to restrict who[2] can make these[3] decisions, depending on the organization building the application using this[4] framework.
	Kandidat kata rekomendasi	1) single, one, two, twenty, 10, 100 2) user, actor, worker, manager, LABEL 3) approval, LABEL 4) LABEL

Gambar 4.3 Hasil rekomendasi perbaikan pernyataan kebutuhan dari ahli (*anotator*)

Pada Gambar 4.3 terdapat empat kata rancu yang terdeteksi pada pernyataan kebutuhan "*A case may involve a multiple requests requiring approval, and security may be needed to restrict who can make these decisions, depending on the organization building the application using this framework.*" dimana kata rancu yang terdeteksi ditandai dengan huruf warna merah dan *caption* 1, 2, 3 dan 4 yaitu kata "*multiple*", "*who*", "*these*" dan "*this*". *Anotator* memberikan rekomendasi pada bagian kandidat kata rekomendasi. Dimana pada pernyataan kebutuhan Gambar 4.3 *anotator* memberikan rekomendasi pada kata ambigu pertama yaitu (*single, one, two, twenty,* 10, 100) kata ambigu kedua yaitu (*user, actor, worker, manager,* LABEL) kata ambigu ketiga yaitu (*approval,* LABEL) dan keempat (LABEL). Kata LABEL pada daftar kata hasil rekomendasi adalah sebuah kata pengganti yang lebih spesifik dan memenuhi kriteria sebagai kata yang tidak rancu misalnya ***New Adelaide Airport*** *framework.*

Daftar kata rancu beserta rekomendasinya akan dikumpulkan dan dimasukkan ke dalam tabel aturan rekomendasi. Hasil implementasi tabel aturan

kata rancu beserta daftar kata rekomendasi yang terdapat didalamnya dapat dilihat pada Gambar 4.4 berikut.

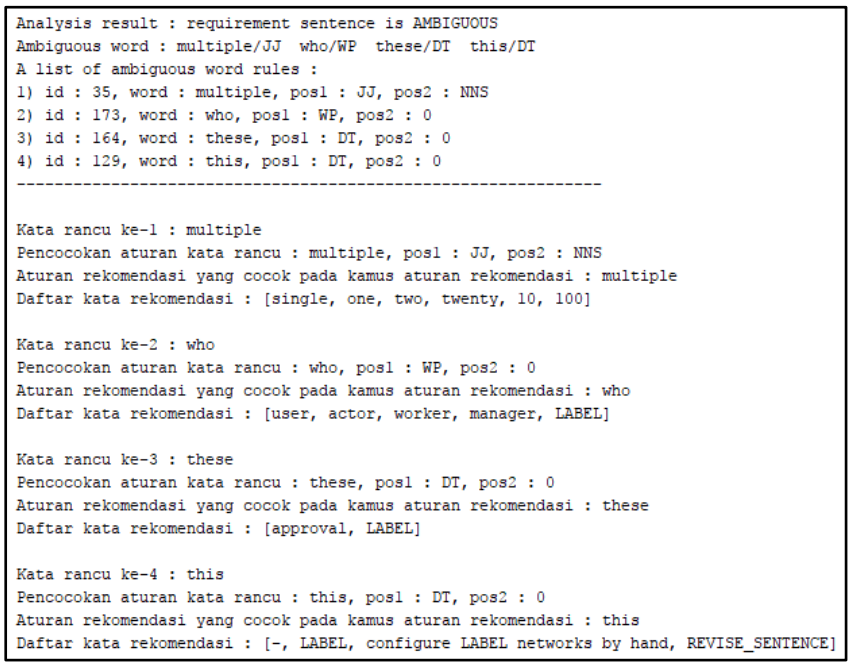


Gambar 4.4 Tabel aturan rekomendasi

Pada Gambar 4.4 terdapat tabel aturan rekomendasi (*Rules*) dimana id aturan (*RuleID*) yang terpilih ialah 35 dengan nama aturan (*RuleName*) "*multiple*", Tag1 "JJ" (adjektif), Tag2 "NNS" (kata benda plural) dan daftar kata rekomendasinya ialah "*single, one, two, twenty, 10, 100*". Adapun jumlah keseluruhan aturan rekomendasi yang terdapat pada tabel aturan rekomendasi ialah berjumlah 189 aturan, dimana daftar aturan rekomendasi ini akan disimpan ke dalam sebuah *file* berformat *Extensible Markup Language* (XML) yang akan digunakan sebagai kamus kata pencocokan aturan rekomendasi.

Pemrosesan awal dataset menghasilkan keluaran berupa daftar kata rancu yang terdeteksi pada sebuah pernyataan kebutuhan. Tahapan selanjutnya ialah mencocokan kata rancu yang terdeteksi dengan aturan rekomendasi yang terdapat pada tabel aturan rekomendasi. Hasil keluaran dari proses pencocokan aturan rekomendasi yang terdapat pada tabel aturan rekomendasi dari pernyataan kebutuhan "*A case may involve a multiple requests requiring approval, and security may be needed to restrict who can make these decisions, depending on*

the organization building the application using this framework." dengan empat kata rancu yang terdeteksi yaitu *multiple*, *who*, *these* dan *this* dapat dilihat pada Gambar 4.5 berikut.



Gambar 4.5 Hasil keluaran proses pencocokan aturan rekomendasi

Hasil keluaran proses pencocokan aturan rekomendasi pada Gambar 4.5 adalah "*single*, *one*, *two*, *twenty*, 10, 100" untuk kata rancu pertama yaitu "*multiple*", sedangkan untuk kata rancu kedua "*who*" adalah "*user*, *actor*, *worker*, *manager*, LABEL", kata rancu ketiga "*these*" adalah "*approval*, LABEL" dan kata rancu keempat "*this*" yaitu "-, LABEL, *configure* LABEL *networks by hand*, REVISE_SENTENCE". Hasil keluaran dari proses ini akan dijadikan sebagai kandidat kata rekomendasi yang akan menggantikan kata rancu pada proses uji coba metode rekomendasi.

4.1.4 Proses Uji Coba Metode Rekomendasi

Proses uji coba metode rekomendasi dilakukan berdasarkan tiga skenario uji coba yaitu uji coba metode rekomendasi menggunakan teknik berbasis aturan dan model bahasa *bigram*, uji coba metode rekomendasi menggunakan teknik berbasis aturan dan model bahasa *trigram*, dan uji coba metode rekomendasi menggunakan teknik statistik berbasis frekuensi *n-gram*.

4.1.4.1 Proses Uji Coba Metode Rekomendasi Menggunakan Teknik Berbasis Aturan dan Model Bahasa Bigram

Kandidat kata yang direkomendasi akan diurutkan berdasarkan nilai probabilitas kalimat menggunakan teknik statistik model bahasa *n-gram.* Dimana model bahasa *n-gram* yang digunakan pada penelitian ini ialah model bahasa *bigram* dan *trigram.* Hal ini dilakukan untuk memilih kandidat kata yang terbaik untuk menggantikan kata ambigu berdasarkan kriteria pilihan kata yang sering muncul pada sebuah korpus (*British National Corpus*). Setiap kandidat kata rekomendasi akan dijadikan satu kalimat dan akan dihitung nilai probabilitasnya. Hasil implementasi metode untuk menghitung nilai probabilitas kalimat dan nilai *perplexity* menggunakan model bahasa *bigram* pada pernyataan kebutuhan "*A case may involve a multiple requests requiring approval, and security may be needed to restrict who can make these decisions, depending on the organization building the application using this framework.*" dan kata yang direkomendasi (*single, one, two, twenty,* 10, 100) dapat dilihat pada Gambar 4.6.

```
Frekuensi Bigram
------------------------
A case : Probabilitas = 1.0284617581461655E-5
case may : Probabilitas = 1.6918642420320726E-5
may involve : Probabilitas = 1.9905391748100725E-5
involve a : Probabilitas = 3.115160751206292E-5
a single : Probabilitas = 5.764152567812812E-4
single requests : Probabilitas = 7.741383298491948E-8
requests requiring : Probabilitas = 7.750870887852959E-8
requiring approval : Probabilitas = 6.200462663022758E-7
approval , : Probabilitas = 2.402336869314889E-5
, and : Probabilitas = 0.023624514804829147
and security : Probabilitas = 4.0826799152334394E-5
security may : Probabilitas = 7.745430621931744E-7
may be : Probabilitas = 0.00290434268015724 5
be needed : Probabilitas = 6.835059604155569E-5
needed to : Probabilitas = 3.476214255575298E-4
to restrict : Probabilitas = 3.4599437781771194E-5
restrict who : Probabilitas = 7.751000654184455E-8
who can : Probabilitas = 1.8470626051863195E-4
can make : Probabilitas = 2.031274619490301E-4
make these : Probabilitas = 1.64158205968145E-5
these decisions : Probabilitas = 7.001341564756754E-6
decisions , : Probabilitas = 4.6019174811118015E-5
, depending : Probabilitas = 6.144819498578265E-5
depending on : Probabilitas = 1.5608905012062801E-4
on the : Probabilitas = 0.015238374628298237
the organization : Probabilitas = 7.691416439342884E-5
organization building : Probabilitas = 7.748487204729367E-8
building the : Probabilitas = 2.3379489761989828E-5
the application : Probabilitas = 1.2345597409740708E-4
application using : Probabilitas = 2.323842836649883E-7
using this : Probabilitas = 2.7160947417179615E-5
this framework : Probabilitas = 7.928466206498488E-6
framework . : Probabilitas = 3.10752670090009E-5
Probabilitas kalimat rekomendasi (Bigram) dari kata rekomendasi "single" : 3.786579506240273E-158
Nilai Perplexity : 9.07634639772373E8
```

Gambar 4.6 Nilai *perplexity bigram* pada pernyataan kebutuhan untuk kata rekomendasi yang pertama "*single*"

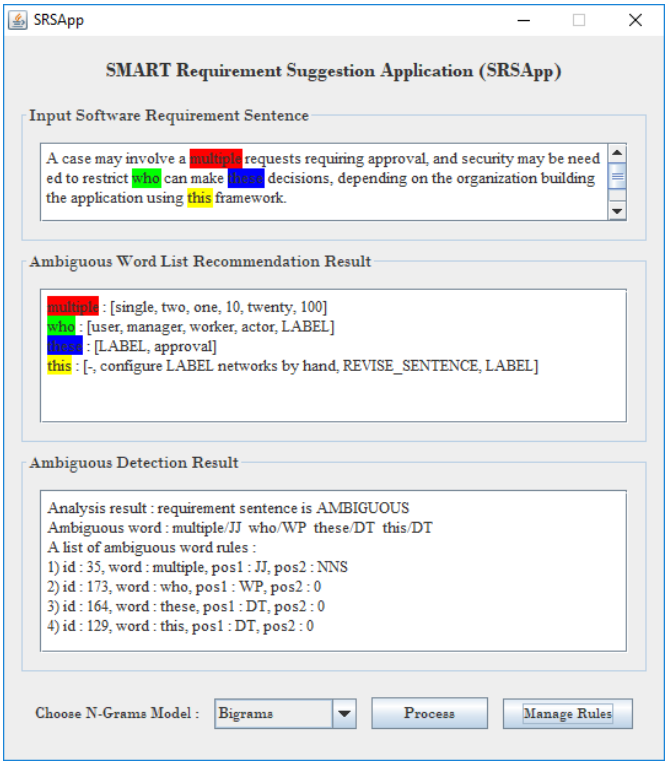
Pada Gambar 4.6 dapat diketahui bahwa nilai *perplexity bigram* untuk kata rekomendasi yang pertama ("*single*") pada kata ambigu "*multiple*" ialah sebesar 9.07 x 10^{-8}. Nilai *perplexity* dari setiap kandidat kata rekomendasi akan dihitung dan diurutkan berdasarkan nilai terendah (minimum). Hasil dari proses pengurutan nilai *perplexity bigram* berdasarkan nilai minimum dapat dilihat pada Gambar 4.7 berikut.

```
Kata Rancu :
        multiple
Hasil Rekomendasi :
         Kata rekomendasi single --> nilai perplexity bigrams : 9.07634639772373E8
         Kata rekomendasi two --> nilai perplexity bigrams : 9.741761875760134E8
         Kata rekomendasi one --> nilai perplexity bigrams : 1.0426290996578531E9
         Kata rekomendasi 10 --> nilai perplexity bigrams : 1.1052767607286303E9
         Kata rekomendasi twenty --> nilai perplexity bigrams : 1.1221813032205384E9
         Kata rekomendasi 100 --> nilai perplexity bigrams : 1.141791877432292E9
Kata Rancu :
        who
Hasil Rekomendasi :
         Kata rekomendasi user --> nilai perplexity bigrams : 1.2266061853827307E9
         Kata rekomendasi manager --> nilai perplexity bigrams : 1.2937949001876314E9
         Kata rekomendasi worker --> nilai perplexity bigrams : 1.3627577492904665E9
         Kata rekomendasi actor --> nilai perplexity bigrams : 1.4332231037023087E9
         Kata rekomendasi LABEL --> nilai perplexity bigrams : 1.6518772732245767E9
Kata Rancu :
        these
Hasil Rekomendasi :
         Kata rekomendasi LABEL --> nilai perplexity bigrams : 1.8612764323469315E9
         Kata rekomendasi approval --> nilai perplexity bigrams : 1.8613062159533699E9
Kata Rancu :
        this
Hasil Rekomendasi :
         Kata rekomendasi - --> nilai perplexity bigrams : 1.3807619209631827E9
         Kata rekomendasi configure LABEL networks by hand --> nilai perplexity bigrams : 1.9288252192283037E9
         Kata rekomendasi REVISE_SENTENCE --> nilai perplexity bigrams : 1.9288252192283037E9
         Kata rekomendasi LABEL --> nilai perplexity bigrams : 1.9288253388409066E9
Jumlah kata rancu : 4
```

Gambar 4.7 Hasil pengurutan nilai *perplexity bigram.*

Pada Gambar 4.7 dapat diketahui bahwa tiga urutan kata yang diurutkan berdasarkan nilai minimum *perplexity* pada kata rancu *"multiple*" ialah kata "*single*" dengan nilai *perplexity* 9.07 x 10^{-8}, "*two*" dengan nilai *perplexity* 9.74 x 10^{-8} dan "*one*" dengan nilai *perplexity* sebesar 1.04 x 10^{-9}. Sedangkan untuk kata rancu yang kedua "*who*" tiga urutan kata yang direkomendasikan yaitu kata "*user*" dengan nilai *perplexity* 1.22 x 10^{-9}, "*manager*" dengan nilai *perplexity* 1.29 x 10^{-9}, dan "*worker*" dengan nilai *perplexity* sebesar 1.36 x 10^{-9}. Kata rancu ketiga "*these*" urutan kata rekomendasi "LABEL" dengan nilai perplexity sebesar 1.8612 x 10^{-9} dan "*approval*" dengan nilai *perplexity* sebesar 1.8613 x 10^{-9}. Kata rancu keempat "*this*" urutan kata rekomendasi ialah "-" (yang bermaksud penghapusan kata yang ambigu) dengan nilai *perplexity* 1.38 x 10^{-9} dan "*configure* LABEL *networks by hand*, REVISE_SENTENCE, LABEL" dengan masing-masing nilai *perplexity* yang sama yaitu 1.92 x 10^{-9}.

Tampilan hasil keluaran kata rekomendasi yang telah diurut berdasarkan nilai minimum *perplexity bigram* pada antarmuka SRSApp dapat dilihat pada Gambar 4.8 berikut.



Gambar 4.8 Hasil keluaran metode rekomendasi menggunakan *bigram*

Pada Gambar 4.8 terdapat kata rancu pertama "*multiple*" dengan urutan pilihan kata rekomendasi yaitu [*single*, *two*, *one*, 10, *twenty*, 100] kata rancu kedua "*who*" dengan urutan pilihan kata rekomendasi [*user*, *manager*, *worker*, *actor*, LABEL] kata rancu ketiga "*these*" dengan urutan pilihan kata rekomendasi [LABEL, *approval*] dan kata rancu keempat "*this*" dengan urutan pilihan kata rekomendasi [-, *configure* LABEL *networks by hand*, REVISE_SENTENCE, LABEL]. Ada beberapa kata khusus yang digunakan sebagai konstanta kata hasil rekomendasi yaitu :

Tanda '-' (tanpa tanda petik) pada hasil keluaran metode rekomendasi bermaksud hasil rekomendasi terhadap kata rancu ialah penghilangan kata rancu tersebut.

LABEL yang bermaksud hasil rekomendasi dari kata rancu adalah sebuah kata pengganti yang lebih spesifik dan memenuhi kriteria sebagai kata yang tidak rancu.

REVISED_SENTENCE yang bermaksud hasil rekomendasi dari kata rancu adalah sebuah kegiatan yang menyarankan pengguna (penulis dokumen SKPL) untuk melakukan pemeriksaan terhadap pernyataan kebutuhan. Hal ini mungkin disebabkan terlalu banyak penggunaan kata rancu pada sebuah pernyataan kebutuhan atau terdapat hal yang menyebabkan pernyataan kebutuhan menjadi tidak konsisten.

REVISE_PHRASE yang bermaksud hasil rekomendasi dari kata rancu adalah sebuah kegiatan yang menyarankan pengguna untuk melakukan pemeriksaan terhadap struktur frasa dalam bahasa Inggris.

SPLIT_SENTENCE yang bermaksud hasil rekomendasi dari kata rancu adalah sebuah kegiatan yang menyarankan pengguna untuk melakukan pemisahan bagian kata sebelum dan sesudah kata rancu, sehingga menjadi dua buah pernyataan kebutuhan yang lebih disarankan.

appropriate (DESCRIPTIVE_OF_LABEL) yang bermaksud hasil rekomendasi dari kata rancu "*appropriate*" adalah saran pemberian keterangan dari kata rancu tersebut di dalam tanda buka dan tutup kurung, sehingga kata rancu tersebut bisa menjadi lebih deskriptif (pembaca memiliki satu pemahaman yang sama terhadap kata rancu).

Adapun untuk mengubah kata rancu yang terdeteksi pada kotak panel "*Input Software Requirement Sentence*" di antarmuka SRSApp dapat dilakukan dengan cara mengetuk salah satu bagian kata pada pilihan kata rekomendasi yang ada pada kotak panel "*Ambiguous Word List Recommendation Result*".

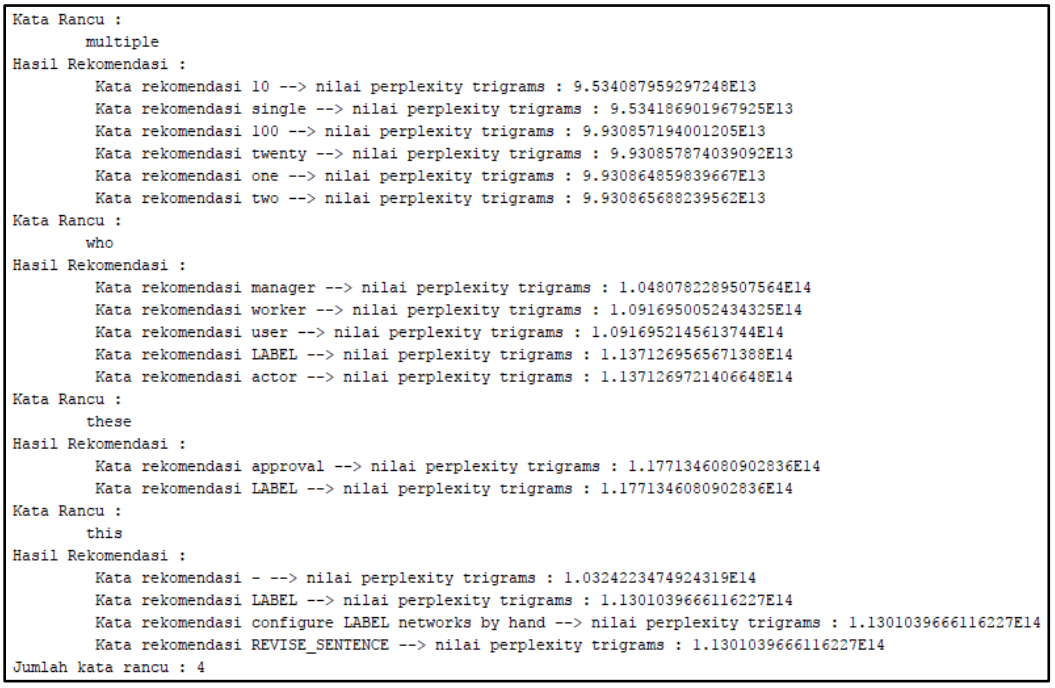
4.1.4.2 Proses Uji Coba Metode Rekomendasi Menggunakan Teknik Berbasis Aturan dan Model Bahasa Trigrams

Skenario uji coba metode rekomendasi menggunakan teknik berbasis aturan juga dilakukan dengan menghitung nilai probabilitas kalimat dan nilai *perplexity* dari kandidat kata yang direkomendasi menggunakan model bahasa *trigram.* Hasil implementasi metode untuk menghitung nilai probabilitas kalimat dan nilai *perplexity* menggunakan model bahasa *trigram* pada pernyataan kebutuhan "*A case may involve a multiple requests requiring approval, and security may be needed to restrict who can make these decisions, depending on the organization building the application using this framework.*" dan kata yang

direkomendasi (*single*, *one*, *two*, *twenty*, 10, 100) dapat dilihat pada Gambar 4.9 berikut.

Gambar 4.9 Nilai *perplexity trigram* pada pernyataan kebutuhan untuk kata rekomendasi yang pertama "*single*"

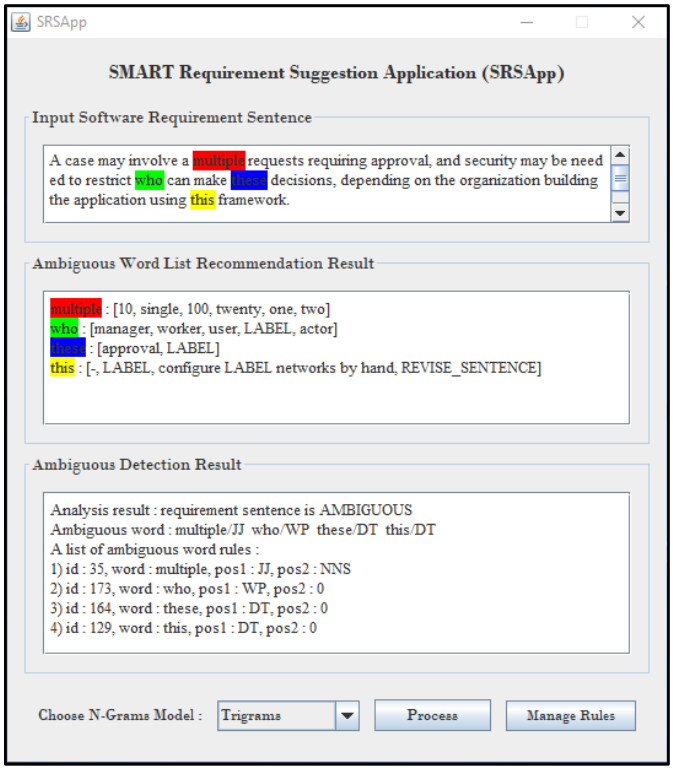
Pada Gambar 4.9 dapat diketahui bahwa nilai *perplexity trigram* untuk kata rekomendasi yang pertama ("*single*") pada kata ambigu "*multiple*" ialah sebesar 9.53×10^{-13}. Setelah mendapatkan nilai *perplexity* pada tiap-tiap kandidat kata yang direkomendasi menggunakan model statistik *trigram* langkah berikutnya ialah mengurutkan nilai *perplexity* yang diperoleh pada masing-masing kata rekomendasi berdasarkan nilai minimum. Hasil dari proses pengurutan nilai *perplexity trigram* berdasarkan nilai minimum dapat dilihat pada Gambar 4.10 berikut.



Gambar 4.10 Hasil pengurutan nilai *perplexity trigram*

Pada Gambar 4.10 dapat diketahui bahwa tiga urutan kata yang diurutkan berdasarkan nilai minimum *perplexity* pada kata rancu *"multiple*" ialah kata "10" dengan nilai *perplexity* 9.5340 x 10^{-13}, "*single*" dengan nilai *perplexity* 9.5341 x 10^{13} dan "100" dengan nilai *perplexity* sebesar 9.93 x 10^{-13}. Sedangkan untuk kata rancu yang kedua "*who*" tiga urutan kata yang direkomendasikan yaitu kata "*manager*" dengan nilai *perplexity* 1.04 x 10^{-14}, "*worker*" dengan nilai *perplexity* 1.09 x 10^{-14}, dan "*user*" dengan nilai *perplexity* sebesar 1.091 x 10^{-14}. Kata rancu ketiga "*these*" urutan kata rekomendasi "*approval*" dan LABEL dengan nilai *perplexity* sebesar 1.177 x 10^{-14}. Kata ambigu keempat "*this*" urutan kata rekomendasi "-" (yang bermaksud penghapusan kata yang ambigu) dengan nilai *perplexity* 1.03 x 10^{-14} dan "LABEL, *configure* LABEL *networks by hand*, REVISE_SENTENCE" dengan masing-masing nilai *perplexity* yang sama yaitu 1.13 x 10^{-14}.

Tampilan hasil keluaran kata rekomendasi yang telah diurutkan berdasarkan nilai minimum *perplexity trigram* pada antarmuka SRSApp dapat dilihat pada Gambar 4.11 berikut.



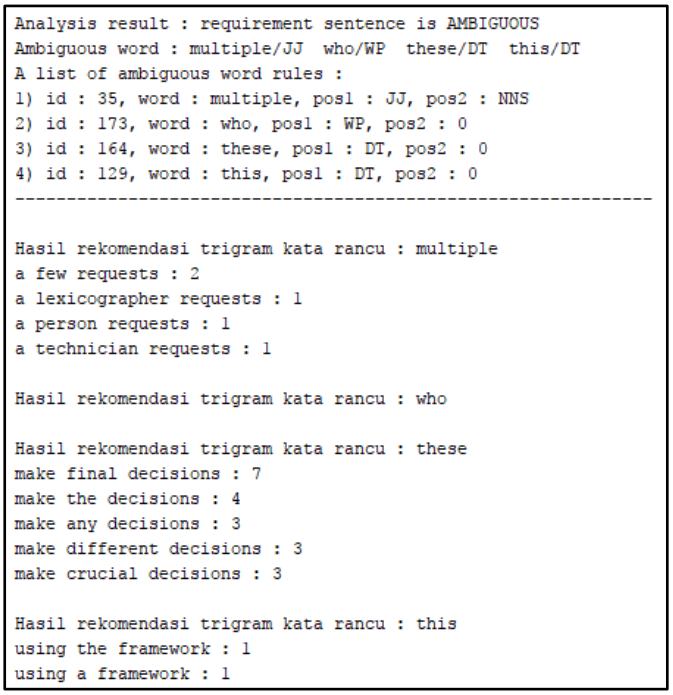
Gambar 4.11 Hasil keluaran metode rekomendasi menggunakan *trigram*

Pada Gambar 4.11 terdapat kata rancu pertama "*multiple*" dengan urutan pilihan perbaikan kata rekomendasi yaitu [10, *single*, 100, *twenty*, *one*, *two*], kata rancu kedua "*who*" dengan urutan pilihan perbaikan kata rekomendasi [*manager*, *worker*, *user*, LABEL, *actor*], kata rancu ketiga "*these*" dengan urutan pilihan perbaikan kata rekomendasi [*approval*, LABEL] dan kata rancu keempat "*this*" dengan urutan pilihan perbaikan kata rekomendasi [-, LABEL, *configure* LABEL *networks by hand*, REVISE_SENTENCE].

4.1.4.3 Proses Uji Coba Metode Rekomendasi Menggunakan Teknik Statistik Berbasis N-Gram

Metode rekomendasi menggunakan teknik statistik berbasis *n-gram* akan memberikan hasil rekomendasi untuk menggantikan kata rancu yang ditemukan berdasarkan frekuensi *trigram* kata yang sering muncul pada sebuah korpus. Dimana korpus yang digunakan pada penelitian ini ialah *British National Corpus* (BNC) versi lengkap (sumber : *http://www.natcorp.ox.ac.uk/*). Adapun jumlah pasangan *trigram* kata yang ada pada korpus BNC berjumlah 47.246.076 pasangan *trigram* kata.

Proses uji coba metode rekomendasi menggunakan teknik statistik berbasis frekuensi *n-gram* dilakukan dengan menghitung frekuensi kemunculan *trigram* kata pada korpus. Pencarian *trigram* kata pada korpus dilakukan dengan cara mencari kemunculan kata pada korpus dengan dua kata kunci pencarian yaitu kata sebelum kata rancu akan dijadikan sebagai kata kunci pertama sedangkan kata sesudah kata rancu akan dijadikan sebagai kata kunci kedua. Hasil frekuensi kemunculan *trigram* kata yang direkomendasi dari pernyataan kebutuhan "*A case may involve a multiple requests requiring approval, and security may be needed to restrict who can make these decisions, depending on the organization building the application using this framework.*" dengan empat kata rancu yang ditemukan yaitu *multiple*, *who*, *these* dan *this* dapat dilihat pada Gambar 4.12 berikut.



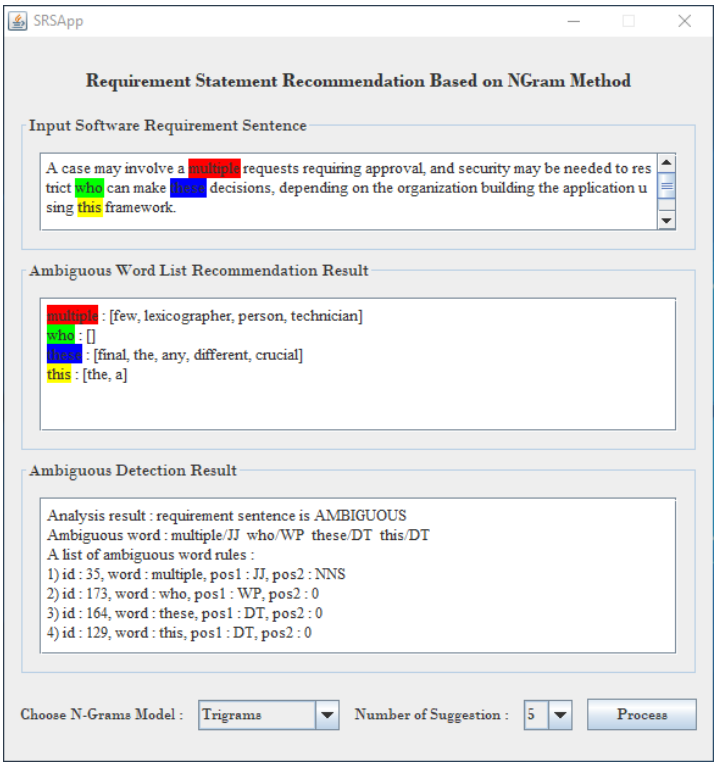
```
Analysis result : requirement sentence is AMBIGUOUS
Ambiguous word : multiple/JJ  who/WP  these/DT  this/DT
A list of ambiguous word rules :
1) id : 35, word : multiple, pos1 : JJ, pos2 : NNS
2) id : 173, word : who, pos1 : WP, pos2 : 0
3) id : 164, word : these, pos1 : DT, pos2 : 0
4) id : 129, word : this, pos1 : DT, pos2 : 0
------------------------------------------------------------

Hasil rekomendasi trigram kata rancu : multiple
a few requests : 2
a lexicographer requests : 1
a person requests : 1
a technician requests : 1

Hasil rekomendasi trigram kata rancu : who

Hasil rekomendasi trigram kata rancu : these
make final decisions : 7
make the decisions : 4
make any decisions : 3
make different decisions : 3
make crucial decisions : 3

Hasil rekomendasi trigram kata rancu : this
using the framework : 1
using a framework : 1
```

Gambar 4.12 Frekuensi kemunculan *trigram* kata hasil rekomendasi

Berdasarkan hasil frekuensi kemunculan *trigram* kata pada Gambar 4.12 dapat diketahui bahwa kata kunci untuk pencarian kata rekomendasi pada kata rancu yang pertama (*multiple*) yaitu "*a*" dan "*request*", hasil pencarian menunjukkan pasangan kata "*a few request*" memiliki frekuensi terbanyak yaitu 2 kali. Kata kunci pada kata rancu yang kedua (*who*) yaitu "*restrict*" dan "*can*", hasil pencarian menunjukkan tidak terdapat pasangan *trigram* kata dengan dua kata kunci tersebut, sehingga tidak ada kata yang akan direkomendasi untuk kata rancu yang kedua. Kata kunci untuk kata rancu ketiga (*these*) yaitu "*make*" dan

"*decisions*", hasil pencarian menunjukkan pasangan kata "*make final decisions*" memiliki frekuensi terbanyak yaitu 7 kali. Kata kunci untuk kata rancu keempat (*this*) yaitu "*using*" dan "*framework*", hasil pencarian menunjukkan pasangan kata "*using the framework*" dan pasangan kata "*using a framework*" memiliki frekuensi yang sama yaitu sebanyak 1 kali.

Hasil frekuensi kemunculan pasangan *trigram* kata akan diurutkan dan ditampilkan sebagai kata hasil rekomendasi. Tampilan hasil keluaran kata rekomendasi yang telah diurutkan berdasarkan nilai terbanyak pada antarmuka SRSApp dapat dilihat pada Gambar 4.13 berikut.



Gambar 4.13 Hasil keluaran metode rekomendasi menggunakan teknik statistik berbasis frekuensi *n-gram*

Pada Gambar 4.13 terdapat kata rancu pertama yaitu "*multiple*" dengan urutan pilihan perbaikan kata rekomendasi yaitu [*few*, *lexicographer*, *person*, *technician*], kata rancu kedua yaitu "*who*" yang tidak memiliki pilihan perbaikan kata rekomendasi, kata ambigu ketiga "*these*" dengan urutan pilihan perbaikan kata rekomendasi [*final*, *the*, *any*, *different*, *crucial*] dan kata ambigu keempat "*this*" dengan urutan pilihan perbaikan kata rekomendasi [*the*, *a*].

4.2 Analisis dan Evaluasi Metode Rekomendasi

Setelah melakukan uji coba metode rekomendasi dan mendapatkan hasil keluarannya langkah selanjutnya ialah melakukan evaluasi terhadap metode rekomendasi yang diusulkan yaitu metode rekomendasi menggunakan teknik berbasis aturan dan model bahasa *bigram* maupun *trigram*, serta metode rekomendasi menggunakan teknik statistik berbasis frekuensi *n-gram*. Evaluasi dilakukan untuk mengukur nilai reliabilitas antara metode rekomendasi dengan ahli. Hasil keluaran dari metode rekomendasi akan di evaluasi oleh ahli untuk mendapatkan indeks kesepakatan antara hasil keluaran metode rekomendasi dan analisis ahli. Data pengujian yang digunakan pada penelitian ini berjumlah 100 pernyataan kebutuhan yang diambil dari 12 dataset merujuk pada Tabel 4.1. Berikut adalah hasil keluaran metode rekomendasi dari tiga data pengujian yang telah di evaluasi oleh ahli.

Tabel 4.2 Hasil evaluasi metode rekomendasi dari ahli data pengujian nomor 1

1	Pernyataan kebutuhan	As the main incoming channel for cases is frequently the telephone, the system must have sub-second response times and be intuitive to use.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	As_IN the_DT main_JJ incoming_JJ channel_NN for_IN cases_NNS is_VBZ frequently_RB the_DT telephone_NN ,_, the_DT system_NN must_MD have_VB sub-second_JJ response_NN times_NNS and_CC be_VB intuitive_JJ to_TO use_VB ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) incoming_JJ → channel_NN (aturan 40) 2) frequently_RB → the_DT (aturan 29)
	Kata rancu (kata yang berwarna merah)	As the main incoming[1] channel for cases is frequently[2] the telephone, the system must have sub-second response times and be intuitive to use.
	Hasil keluaran metode rekomendasi	1) [-, single, formal, LABEL] 2) [-]
Jawaban : Setuju		

Daftar urutan kata rekomendasi pernyataan kebutuhan "*As the main incoming channel for cases is frequently the telephone, the system must have sub-second response times and be intuitive to use.*" telah di evaluasi oleh ahli dan ahli menyatakan setuju bahwa urutan daftar kata hasil rekomendasi dari metode dapat diterima untuk menggantikan kata rancu yang terdeteksi yaitu "*incoming*" dan "*frequently*" tanpa ada saran perubahan daftar urutan kata.

Tabel 4.3 Hasil evaluasi metode rekomendasi dari ahli data pengujian nomor 2

2	Pernyataan kebutuhan	Prioritise the cases based on specific rules (including Artificial Intelligence engines that deal with complex rules).
	Hasil penandaan tiap bagian kata (*POS Tagging*)	Prioritise_VB the_DT cases_NNS based_VBN on_IN specific_JJ rules_NNS -LRB-_-LRB- including_VBG Artificial_NNP Intelligence_NNP engines_NNS that_WDT deal_VBP with_IN complex_JJ rules_NNS -RRB-_-RRB- ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) specific_JJ → rules_NNS (aturan 30) 2) complex_JJ → rules_NNS (aturan 31)
	Kata rancu (kata yang berwarna merah)	Prioritise the cases based on specific[1] rules (including Artificial Intelligence engines that deal with complex[2] rules).
	Hasil keluaran metode rekomendasi	1) [two[2], three[3], four[4], -[1]] 2) [three[2], four[3], two[1], -[4]]
Jawaban : Setuju		

Daftar urutan kata rekomendasi pernyataan kebutuhan "*Prioritise the cases based on specific rules (including Artificial Intelligence engines that deal with complex rules).*" telah di evaluasi oleh ahli dan ahli menyatakan setuju bahwa urutan daftar kata hasil rekomendasi dari metode dapat diterima untuk menggantikan kata rancu yang terdeteksi yaitu "*specific*" dan "*complex*" dengan saran perbaikan urutan daftar kata pada kata rancu "*specific*" yaitu (-, *two*, *three*, *four*) dan pada kata rancu "*complex*" yaitu (*two*, *three*, *four*, -).

Tabel 4.4 Hasil evaluasi metode rekomendasi dari ahli data pengujian nomor 35

35	Pernyataan kebutuhan	Changing parameters in the network must be done in a short time on all related cells to keep the disturbance to the network as low as possible.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	Changing_VBG parameters_NNS in_IN the_DT network_NN must_MD be_VB done_VBN in_IN a_DT short_JJ time_NN on_IN all_DT related_JJ cells_NNS to_TO keep_VB the_DT disturbance_NN to_TO the_DT network_NN as_RB low_JJ as_IN possible_JJ ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) short_JJ → time_NN (aturan 43) 2) all_DT → related_JJ (aturan 57) 3) low_JJ → as_IN (aturan 37) 4) possible_JJ → ._. (aturan 116)
	Kata rancu (kata yang	Changing parameters in the network must be done in a

	berwarna merah)	short[1] time on all[2] related cells to keep the disturbance to the network as low[3] as possible[4].
	Hasil keluaran metode rekomendasi	1) [short (smaller than 200 words), short (smaller than a quarter A4 page), at most one second] 2) [at least in 80% of, at least 80% of related cells, at least 80 business objects, a, two, -, four, three, one, an, 100%, 100% of, LABEL] 3) [at the highest two, at most two, at most two second, at most two, at most three, at most four, at most three second, at most four second, at most 10dB, software with at most 10 module, software with at most 100 loc] 4) [-, REVISE_SENTENCE]
Jawaban : Tidak Setuju		

Daftar urutan kata rekomendasi pernyataan kebutuhan "*Changing parameters in the network must be done in a short time on all related cells to keep the disturbance to the network as low as possible.*" telah di evaluasi oleh ahli dan ahli menyatakan tidak setuju bahwa urutan daftar kata hasil rekomendasi dari metode dapat diterima untuk menggantikan kata rancu yang terdeteksi yaitu "*short*" dan "*low*".

Hasil evaluasi terhadap metode rekomendasi pada tiga skenario uji coba yaitu uji coba metode rekomendasi menggunakan teknik berbasis aturan dan model bahasa *bigram*, uji coba metode rekomendasi menggunakan teknik berbasis aturan dan model bahasa *trigram* dan uji coba metode rekomendasi menggunakan teknik statistik berbasis frekuensi *n-gram* dapat dilihat pada Tabel 4.5 berikut.

Tabel 4.5 Hasil evaluasi metode rekomendasi dari ahli (10 sampel data pengujian)

Nomor data pengujian	Penilaian Ahli (Teknik Berbasis Aturan dan Model Bigrams)	Penilaian Ahli (Teknik Berbasis Aturan dan Model Trigrams)	Penilaian Ahli (Teknik Statistik Berbasis Frekuensi N-Grams)
1	Setuju	Setuju	Tidak Setuju
2	Setuju	Setuju	Setuju
3	Tidak Setuju	Tidak Setuju	Tidak Setuju
4	Setuju	Setuju	Setuju
5	Setuju	Setuju	Setuju
6	Tidak Setuju	Tidak Setuju	Tidak Setuju
7	Setuju	Setuju	Setuju
8	Setuju	Setuju	Setuju
9	Tidak Setuju	Tidak Setuju	Tidak Setuju
35	Tidak Setuju	Tidak Setuju	Tidak Setuju

Data pengujian nomor 1 pada Tabel 4.5 ahli menjawab setuju pada skenario hasil uji coba metode rekomendasi menggunakan teknik berbasis aturan dan model bahasa *bigram* dan *trigram,* sedangkan pada hasil uji coba metode rekomendasi menggunakan teknik statistik berbasis frekuensi *n-gram* ahli menjawab tidak setuju.

Hasil evaluasi ahli terhadap data pengujian selanjutnya akan dibuat tabel kesepakatan antara penilaian ahli dan hasil keluaran kakas bantu pada masing-masing skenario uji coba. Tabel hasil kesepakatan antara ahli dan kakas bantu pada tiga skenario uji coba dari 100 data pengujian dapat dilihat pada Tabel 4.6 berikut.

Tabel 4.6 Hasil kesepakatan antara ahli dan kakas bantu

Penilaian ahli terhadap hasil keluaran kakas bantu	Skenario Uji Coba		
	Teknik Berbasis Aturan dan Model Bigrams (S_1)	Teknik Berbasis Aturan dan Model Trigrams (S_2)	Teknik Statistik Berbasis Frekuensi N-Grams (S_3)
Setuju	60	66	30
Tidak Setuju	40	34	70
Total	100	100	100

Nilai kesepakatan yang terobservasi pada masing-masing skenario uji coba dapat dihitung dengan persamaan berikut :

$$P(S_1) = \frac{60}{100} = 0{,}60$$

$$P(S_2) = \frac{66}{100} = 0{,}66$$

$$P(S_3) = \frac{30}{100} = 0{,}30$$

Dimana ;

$P(S_1)$: nilai kesepakatan yang terobservasi pada skenario uji coba metode rekomendasi menggunakan teknik berbasis aturan dan model bahasa *bigram.*

$P(S_2)$: nilai kesepakatan yang terobservasi pada skenario uji coba metode rekomendasi menggunakan teknik berbasis aturan dan model bahasa *trigram*.

$P(S_3)$: nilai kesepakatan yang terobservasi pada skenario uji coba metode rekomendasi menggunakan teknik statistik berbasis frekuensi *n-gram*.

Nilai indeks Kappa menggunakan pendekatan Gwet's AC1 dari skenario hasil uji coba metode rekomendasi menggunakan teknik berbasis aturan dan model bahasa *bigram* dapat diketahui berdasarkan perhitungan berikut.

$$e(\gamma) = 2\left(\frac{60+100}{2 \times 100}\right)\left(1 - \frac{60+100}{2 \times 100}\right) = 0.32$$

Indeks Kappa AC1(S_1) = (0.60 - 0.32) / (1 - 0.32) = **0.4117**

Nilai indeks Kappa dari skenario hasil uji coba metode rekomendasi menggunakan teknik berbasis aturan dan model *trigram* dapat diketahui berdasarkan perhitungan berikut.

$$e(\gamma) = 2\left(\frac{66+100}{2 \times 100}\right)\left(1 - \frac{66+100}{2 \times 100}\right) = 0.2822$$

Indeks Kappa AC1(S_2) = (0.66 - 0.2822) / (1 - 0.2822) = **0.5263**

Nilai indeks Kappa dari skenario hasil uji coba metode rekomendasi menggunakan teknik statistik berbasis frekuensi *n-gram* dapat diketahui berdasarkan perhitungan berikut.

$$e(\gamma) = 2\left(\frac{30+100}{2 \times 100}\right)\left(1 - \frac{30+100}{2 \times 100}\right) = 0.455$$

Indeks Kappa AC1(S_3) = (0.30 - 0.455) / (1 - 0.455) = -**0.2844**

Hasil komparasi antara metode rekomendasi yang diusulkan pada penelitian ini yaitu teknik berbasis aturan menggunakan model bahasa *bigram* dan *trigram* dengan metode rekomendasi lainnya yaitu teknik statistik berbasis frekuensi *n-gram* menunjukkan bahwa metode yang diusulkan memiliki kinerja yang lebih baik dalam memberikan rekomendasi perbaikan pernyataan kebutuhan yang rancu. Hal ini dapat dilihat dari nilai indeks Kappa yang dihasilkan dari ketiga skenario hasil uji coba yaitu hasil uji coba metode rekomendasi menggunakan teknik berbasis aturan dan model bahasa *bigram* memiliki nilai indeks Kappa sebesar 0.4117 dengan proporsi kesepakatan sedang, hasil uji coba metode rekomendasi menggunakan teknik berbasis aturan dan model bahasa *trigram* memiliki nilai indeks Kappa sebesar 0.5263 dengan proporsi kesepakatan sedang, dan nilai indeks Kappa dari hasil uji coba metode rekomendasi menggunakan teknik statistik berbasis frekuensi *n-gram* yaitu sebesar -0.2844 dengan proporsi kesepakatan rendah.

Salah satu faktor yang menyebabkan rendahnya proporsi kesepakatan antara ahli dengan metode rekomendasi menggunakan teknik statistik berbasis

frekuensi *n-gram* ialah kurang sesuainya daftar kata yang direkomendasi untuk menggantikan kata rancu. Pada beberapa kasus uji coba terdapat hasil kata rekomendasi yang masih mengandung kerancuan, sehingga pilihan kata yang direkomendasi menjadi tidak valid untuk menggantikan kata rancu yang terdeteksi. Disisi lain, metode rekomendasi yang diusulkan pada penelitian ini telah mampu mengatasi permasalahan tersebut dengan menyediakan beberapa pilihan kata rekomendasi (daftar kata rekomendasi yang tidak rancu) dari kata rancu yang terdeteksi. Sehingga daftar pilihan kata yang direkomendasi bisa menjadi lebih valid dalam memberikan pilihan kata rekomendasi.

Berdasarkan hasil uji coba terhadap tiga skenario uji coba dapat diketahui bahwa metode rekomendasi menggunakan teknik berbasis aturan dan model bahasa *trigram* memiliki nilai indeks Kappa yang tertinggi yaitu sebesar 0.5263 diantara metode rekomendasi lainnya. Kelebihan metode rekomendasi menggunakan teknik berbasis aturan dengan model bahasa *trigram* ialah metode dapat memberikan urutan pilihan daftar kata rekomendasi yang lebih baik dibandingkan dengan metode rekomendasi menggunakan teknik berbasis aturan dengan model bahasa *bigram*. Akan tetapi, juga terdapat kelemahan dari metode teknik berbasis aturan dengan model bahasa *trigram*, dimana cara kerja pengurutan daftar kata hasil rekomendasi sangat bergantung terhadap frekuensi kata pada sebuah korpus. Pada suatu kasus sering dijumpai pasangan *trigram* kata yang tidak terdapat pada sebuah korpus (tidak memiliki frekuensi pada korpus) sehingga menghasilkan nilai probabilitas kalimat yang rendah sehingga daftar urutan kata rekomendasi menjadi tidak valid pada kasus tersebut.

4.3 Diskusi dan Pengembangan Metode

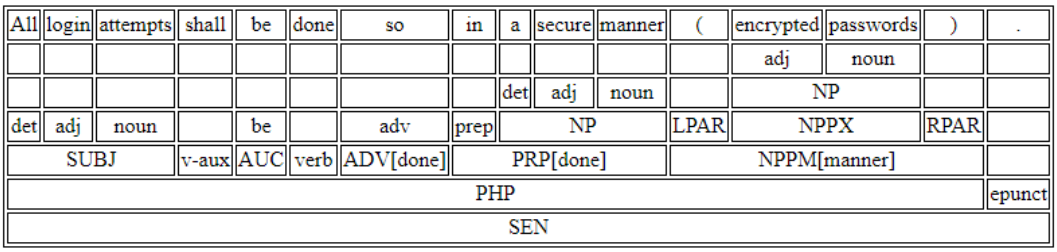
Berdasarkan analisis terhadap metode identifikasi kerancuan pernyataan kebutuhan yang digunakan pada penelitian ini ditemukan satu kasus dimana terdapat kekurangan dari metode identifikasi kerancuan berbasis aturan yang diusulkan yaitu metode ini tidak bisa melakukan analisis kerancuan pada tingkat frasa sebuah kalimat. Berikut adalah dua sampel data yang digunakan pada tahap pemrosesan awal dataset menggunakan metode identifikasi kerancuan berbasis aturan yang salah dalam mendeteksi kerancuan pada kebutuhan perangkat lunak.

Pernyataan kebutuhan :

1) All login attempts shall be done so in a secure manner (encrypted passwords).
2) Each cell is on a site (and a site can have many cells on it).

Hasil dari metode identifikasi kerancuan pada pernyataan kebutuhan pertama yaitu terdapat kata rancu "*secure*" pada pernyataan kebutuhan, dimana metode identifikasi kerancuan bekerja dengan mencocokan kata dan pola rancu pada kamus aturan kata rancu. Setiap bagian kata dalam pernyataan kebutuhan akan dicocokan dengan kata rancu yang terdapat pada kamus aturan kata rancu. Kata "*secure*" ada pada kamus aturan kata rancu sehingga kata "*secure*" dianggap sebagai kata rancu, sedangkan berdasarkan konteksnya kata "*secure*" tersebut seharusnya bukanlah sebuah kata rancu dikarenakan terdapat kata yang menerangkan kata "*secure*" setelah kata "*secure*" pada pernyataan kebutuhan tersebut, yaitu "*encrypted passwords*". Begitu juga dengan pernyataan kebutuhan yang kedua dimana metode mengidentifikasi kata "*it*" sebagai kata rancu, sedangkan berdasarkan konteks pernyataan kebutuhan tersebut, kata "*it*" sebenarnya bukanlah kata yang rancu karena sebelum kata "*it*" sudah terdapat kata yang menjelaskan kata "*it*" yaitu kata "*a site*".

Berdasarkan analisis terhadap dua pernyataan kebutuhan tersebut, penulis mencoba untuk memberikan sebuah ide pengembangan teknik identifikasi kerancuan menggunakan teknik berbasis aturan yang dikombinasikan dengan teknik depedensi *parsing*. Teknik depedensi *parsing* dapat melakukan analisis kalimat pada tingkat frasa/gramatikal. Sehingga teknik ini dapat meningkatkan peforma metode identifikasi kerancuan. Teknik depedensi *parsing* memetakan beberapa kata berdasarkan ketergantungannya dengan kata sesudah kata tersebut (*forward reference*) maupun kata yang sebelumnya (*backward reference*). Berikut adalah hasil parsing pada pernyataan kebutuhan yang pertama "*All login attemps shall be done so in a secure manner (encrypted passwords).*".



Gambar 4.14 Hasil parsing pada pernyataan kebutuhan "*All login attemps shall be done so in a secure manner (encrypted passwords).*"

(Halaman web *parsing* kalimat : *http://zzcad.com/cgi-bin/*)

Pada Gambar 4.14 dapat dilihat baris ketiga dari bawah menunjukkan struktur kalimat, dimana bagian kata "(*encrypted passwords*)" menjelaskan kata sebelumnya yaitu "*secure manner*", sehingga kata "*secure*" disini sebenarnya tidak menimbulkan banyak interpretasi dari pembaca dan tidak rancu.

Berdasakan hasil analisis terhadap metode rekomendasi perbaikan kata rancu yang diusulkan, penulis juga menemukan keterbatasan metode yang diusulkan yaitu hasil urutan rekomendasi yang kadang-kadang ditemukan tidak sesuai dengan konteks pernyataan kebutuhan. Hal ini disebabkan oleh cara kerja pengurutan kata rekomendasi dilakukan dengan cara menghitung nilai probabilitas kalimat (*perplexity*) berdasarkan model bahasa *bigram* dan *trigram* yang memanfaatkan sebuah korpus besar dalam memberikan rekomendasi. Nilai probabilitas kalimat sangat bergantung pada frekuensi *bigram* dan *trigram* kata pada sebuah korpus, semakin sering pasangan *bigram* dan *trigram* kata muncul pada sebuah korpus maka nilai probabilitas akan semakin besar untuk direkomendasikan sebagai pilihan kata rekomendasi. Hal ini tentunya tidak adil jika kata yang direkomendasikan tidak relevan dengan konteks pernyataan kebutuhan akan tetapi memiliki frekuensi yang sering pada sebuah korpus.

Untuk lebih jelasnya, berikut adalah suatu kasus dimana metode rekomendasi memberikan hasil urutan kata rekomendasi yang kurang sesuai sebagai pilihan kata rekomendasi.

Tabel 4.7 Hasil kata rekomendasi metode usulan pada data pengujian nomor 36

36	Pernyataan kebutuhan	Therefore, the changes must be planned and executed in a short time, usually during the middle of the night.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	Therefore_RB ,_, the_DT changes_NNS must_MD be_VB planned_VBN and_CC executed_VBN in_IN a_DT short_JJ time_NN ,_, usually_RB during_IN the_DT middle_NN of_IN the_DT night_NN ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) short_JJ → time_NN (aturan 43)
	Kata rancu (kata yang berwarna merah)	Therefore, the changes must be planned and executed in a short[1] time, usually during the middle of the night.
	Hasil keluaran metode rekomendasi	1) [short (smaller than 200 words), short (smaller than a quarter A4 page), at most one second]

Hasil keluaran urutan kata metode rekomendasi pada Tabel 4.7 untuk kata rancu "*short*" yaitu kata "*short* (*smaller than 200 words*)" pada urutan pertama, kata "*short* (*smaller than a quarter A4 page*)" di urutan kedua, dan kata "*at most one second*" pada urutan ketiga. Jika dilihat dari konteks pernyataan kebutuhan "*Therefore, the changes must be planned and executed in a short time, usually during the middle of the night.*" maka lebih sesuai jika kata "*short*" diganti dengan kata "*at most one second*" pada urutan pertama, sehingga hasil rekomendasi bisa menjadi lebih relevan berdasarkan konteksnya.

Tabel 4.8 Hasil kata rekomendasi metode usulan pada data pengujian nomor 32

32	Pernyataan kebutuhan	This is complicated because each cell has literally hundreds of parameters, A short summary:
	Hasil penandaan tiap bagian kata (*POS Tagging*)	This_DT is_VBZ complicated_VBN because_IN each_DT cell_NN has_VBZ literally_RB hundreds_NNS of_IN parameters_NNS ._. A_DT short_JJ summary_NN :_:
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) This_DT → is_VBZ (aturan 129) 2) each_DT → cell_NN (aturan 57) 3) short_JJ → summary_NN (aturan 43)
	Kata rancu (kata yang berwarna merah)	This[1] is complicated because each[2] cell has literally hundreds of parameters, A short[3] summary:
	Hasil keluaran metode rekomendasi	1) [-, configure LABEL networks by hand, REVISE_SENTENCE, LABEL] 2) [at least in 80% of, at least 80% of related cells, at least 80 business objects, -, a, one, 100%, LABEL, 100% of, two, three, four, an] 3) [at most one second, short (smaller than 200 words), short (smaller than a quarter A4 page)]

Sedangkan hasil keluaran metode rekomendasi untuk pernyataan kebutuhan "*This is complicated because each cell has literally hundreds of parameters, A short summary:*" pada kata rancu "*short*" yaitu kata "*at most one second*" di urutan pertama, kata "*short* (*smaller than 200 words*)" pada urutan kedua dan kata "*short* (*smaller than a quarter A4 page*)" di urutan ketiga, hal ini tentunya tidak sesuai dengan konteks urutan kata rekomendasi yang diharapkan.

Dalam upaya meningkatkan peforma metode usulan rekomendasi berdasarkan permasalahan yang telah dipaparkan penulis juga mencoba untuk memberikan satu ide pengembangan metode usulan rekomendasi yaitu menggunakan teknik berbasis aturan yang dikombinasikan dengan teknik kesamaan kontekstual (*contextual similarity*). Sehingga usulan kata rekomendasi bisa menjadi lebih relevan terhadap konteks kalimat pernyataan kebutuhan. Misalnya untuk pernyataan kebutuhan "*Therefore, the changes must be planned and executed in a short time, usually during the middle of the night.*" jika dilihat dari konteks kata rancu "*short*" maka konteks nya adalah menunjukkan keterangan waktu pada kata setelahnya. Sehingga solusi yang diharapkan untuk menggantikan kata rancu "*short*" ini seharusnya juga tidak jauh dari konteks keterangan waktu yaitu "*at most one second*".

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan hasil uji coba dan evaluasi yang telah dilakukan terhadap metode rekomendasi perbaikan pernyataan kebutuhan perangkat lunak yang rancu menggunakan teknik berbasis aturan dengan model bahasa *n-gram* maka dapat disimpulkan bahwa metode rekomendasi yang diusulkan dapat digunakan untuk memberikan rekomendasi perbaikan pernyataan kebutuhan perangkat lunak yang rancu dengan menyediakan pilihan kata rekomendasi yang lebih baik dibandingkan dengan metode rekomendasi menggunakan teknik statistik berbasis frekuensi *n-gram*. Hal ini dibuktikan dengan hasil uji coba metode rekomendasi yang diusulkan memiliki tingkat proporsi kesepakatan yang lebih baik dibandingkan dengan metode rekomendasi menggunakan teknik statistik berbasis frekuensi *n-gram*.

Hasil skenario uji coba menggunakan teknik berbasis aturan dan model bahasa *trigram* memiliki nilai indeks Kappa tertinggi, yaitu sebesar 0.5263, dengan proporsi kesepakatan sedang. Hal ini menunjukkan bahwa metode rekomendasi yang diusulkan memiliki peforma yang cukup baik dalam memberikan rekomendasi perbaikan pernyataan kebutuhan perangkat lunak yang rancu.

5.2 Saran

Berdasakan hasil pengujian yang telah dilakukan terhadap metode rekomendasi perbaikan kata rancu yang diusulkan, penulis menemukan kekurangan dari metode yang diusulkan yaitu hasil urutan rekomendasi yang kadang-kadang ditemukan tidak relevan dengan konteks pernyataan kebutuhan. Dalam upaya meningkatkan peforma metode rekomendasi yang diusulkan berdasarkan permasalahan tersebut penulis mencoba untuk memberikan satu ide pengembangan metode usulan rekomendasi yaitu menggunakan teknik berbasis aturan yang dikombinasikan dengan teknik kesamaan kontekstual (*contextual similarity*). Sehingga usulan kata yang direkomendasi bisa menjadi lebih relevan terhadap konteks kalimat pernyataan kebutuhan.

{Halaman ini sengaja dikosongkan}

DAFTAR PUSTAKA

Armitage, S. (1996) "RMACS software requirements specification," hal. 1–7. doi: 10.2172/331653.

ATHANASELIS, T. *et al.* (2011) "a Corpus Based Technique for Repairing Ill-Formed Sentences With Word Order Errors Using Co-Occurrences of N-Grams," *International Journal on Artificial Intelligence Tools*, 20(3), hal. 401–424. doi: 10.1142/S0218213011000218.

Berry, D. M. dan Kamsties, E. (2003) *From Contract Drafting to Software Specification: Linguistic Sources of Ambiguity*, *Automated Software Engineering, 1997 …*.

Bussel, D. van (2009) *Detecting ambiguity in requirements specifications*. Tilburg University. Tersedia pada: http://ilk.uvt.nl/downloads/pub/papers/hait/vanbussel2009.pdf.

Cohen, J. (1984) "This Week's Citation Classic'_____," 1(1), hal. 1984.

Gwet, K. (2002) "Kappa Statistic is not satisfactory for assessing the extent of agreement between raters," *Statistical Methods For Inter-Rater Reliability Assessmen*, (1), hal. 1–5. Tersedia pada: http://www.agreestat.com/research_papers/kappa_statistic_is_not_satisfactory.pdf.

Henrich, V. dan Reuter, T. (2009) *LISGrammarChecker: Language Independent Statistical Grammar Checking*. Hochschule Darmstadt & Reykjavík University. Tersedia pada: papers2://publication/uuid/19D55F4B-E422-4630-BF78-2D8030B5D866.

Hussain, H. I. (2007) *Using text classification to automate ambiguity detection in SRS documents*. Concordia University. doi: 9780494344422.

Jurafsky, D. dan Martin, J. (2017) "Speech and language processing," hal. 499. Tersedia pada: http://www.cs.colorado.edu/~martin/SLP/Updates/1.pdf.

Kamsties, E., Berry, D. M. dan Paech, B. (2001) "Detecting Ambiguities in Requirements Documents Using Inspections," *Engineering*, (July), hal. 1–12. Tersedia pada: http://publica.fraunhofer.de/documents/N-6838.html%5Cnhttp://www.cas.mcmaster.ca/wise/wise01/KamstiesBerryP

aech.pdf%5Cnhttp://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.145.6497&rep=rep1&type=pdf.

Kiyavitskaya, N. *et al.* (2008) "Requirements for tools for ambiguity identification and measurement in natural language requirements specifications," 1, hal. 207–239. doi: 10.1007/s00766-008-0063-7.

Lami, G. *et al.* (2005) "An automatic tool for the analysis of natural language requirements," *International Journal of Computer Systems Science & Engineering*, 20(1), hal. 53–62. Tersedia pada: http://matrix.isti.cnr.it/FMT/WEBPAPER/QuARS_JCSSE.pdf.

Muliawan, I. W. dan Siahaan, D. O. (2011) "Analisis ambiguitas kebutuhan perangkat lunak berdasarkan acuan."

Naber, D. (2003) "A Rule-Based Style and Grammar Checker," hal. 1–77. Tersedia pada: http://www.google.com/search?client=safari&rls=10_7_4&q=A+Rule+Based+Style+and+Grammar+Checker&ie=UTF-8&oe=UTF-8%5Cnpapers2://publication/uuid/B4D2196A-B23C-4174-9D9E-05D1E097D597.

Singh, S. P. *et al.* (2016) "Frequency based spell checking and rule based grammar checking," *International Conference on Electrical, Electronics, and Optimization Techniques, ICEEOT 2016*, hal. 4435–4439. doi: 10.1109/ICEEOT.2016.7755557.

Tjong, S. F. (2008) *Avoiding ambiguitiy in requirements specifications, Faculty of Engineering & Computer Science*.

Wu, J., Chang, J. dan Chang, S. J. (2013) "Correcting Serial Grammatical Errors based on N-grams and Syntax," *International Journal of Computational Linguistics & Chinese Language Processing, Volume 18, Number 4, December 2013-Special Issue on Selected Papers from ROCLING XXV*, 18(4), hal. 31–44.

LAMPIRAN

Tabel 20 hasil rekomendasi perbaikan pernyataan kebutuhan yang rancu oleh ahli :

1	Pernyataan kebutuhan	As the main incoming channel for cases is frequently the telephone, the system must have sub-second response times and be intuitive to use.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	As_IN the_DT main_JJ incoming_JJ channel_NN for_IN cases_NNS is_VBZ frequently_RB the_DT telephone_NN ,_, the_DT system_NN must_MD have_VB sub-second_JJ response_NN times_NNS and_CC be_VB intuitive_JJ to_TO use_VB ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 3) incoming_JJ → channel_NN (aturan 40) 4) frequently_RB → the_DT (aturan 29)
	Kata rancu (kata yang berwarna merah)	As the main incoming[1] channel for cases is frequently[2] the telephone, the system must have sub-second response times and be intuitive to use.
	Kandidat kata rekomendasi	3) -, formal, single 4) -
2	Pernyataan kebutuhan	Prioritise the cases based on specific rules (including Artificial Intelligence engines that deal with complex rules).
	Hasil penandaan tiap bagian kata (*POS Tagging*)	Prioritise_VB the_DT cases_NNS based_VBN on_IN specific_JJ rules_NNS -LRB-_-LRB- including_VBG Artificial_NNP Intelligence_NNP engines_NNS that_WDT deal_VBP with_IN complex_JJ rules_NNS -RRB-_-RRB- ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 3) specific_JJ → rules_NNS (aturan 30) 4) complex_JJ → rules_NNS (aturan 31)
	Kata rancu (kata yang berwarna merah)	Prioritise the cases based on specific[1] rules (including Artificial Intelligence engines that deal with complex[2] rules).
	Kandidat kata rekomendasi	3) -, two, three, four 4) -, two, three four
3	Pernyataan kebutuhan	The design problem for Design-fest® is to create an Object Oriented framework for use in developing Case Management applications quickly, and customizable to the particular work practices of the company purchasing the framework.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	The_DT design_NN problem_NN for_IN Design-fest_JJ ®_NN is_VBZ to_TO create_VB an_DT Object_NNP Oriented_NNP framework_NN for_IN use_NN in_IN developing_VBG Case_NNP Management_NNP applications_NNS quickly_RB ,_, and_CC

		customizable_JJ to_TO the_DT particular_JJ work_NN practices_NNS of_IN the_DT company_NN purchasing_VBG the_DT framework_NN ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) quickly_RB → . (aturan 32) 2) particular_JJ → work_NN (aturan 34)
	Kata rancu (kata yang berwarna merah)	The design problem for Design-fest® is to create an Object Oriented framework for use in developing Case Management applications quickly[1], and customizable to the particular[2] work practices of the company purchasing the framework.
	Kandidat kata rekomendasi	1) -, less than one hour 2) -, two, three
4	Pernyataan kebutuhan	A case may involve a multiple requests requiring approval, and security may be needed to restrict who can make these decisions, depending on the organization building the application using this framework.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	A_DT case_NN may_MD involve_VB a_DT multiple_JJ requests_NNS requiring_VBG approval_NN ,_, and_CC security_NN may_MD be_VB needed_VBN to_TO restrict_VB who_WP can_MD make_VB these_DT decisions_NNS ,_, depending_VBG on_IN the_DT organization_NN building_VBG the_DT application_NN using_VBG this_DT framework_NN ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) multiple_JJ → requests_NNS (aturan 35) 2) who_WP → can_MD (aturan 173) 3) these_DT → decisions_NNS (aturan 164) 4) this_DT → framework_NN (aturan 129)
	Kata rancu (kata yang berwarna merah)	A case may involve a multiple[1] requests requiring approval, and security may be needed to restrict who[2] can make these[3] decisions, depending on the organization building the application using this[4] framework.
	Kandidat kata rekomendasi	1) single, one, two, twenty, 10, 100 2) user, actor, worker, manager, LABEL 3) approval, LABEL 4) LABEL
5	Pernyataan kebutuhan	The manager then determines the next appropriate step (close, assign to another case worker for the next processing step, put aside until a bring-forward is activated, etc.).
	Hasil penandaan tiap bagian kata (*POS Tagging*)	The_DT manager_NN then_RB determines_VBZ the_DT next_JJ appropriate_JJ step_NN -LRB-_-LRB- close_RB ,_, assign_VB to_TO another_DT case_NN worker_NN for_IN the_DT next_JJ processing_NN step_NN ,_, put_VBD aside_RB until_IN a_DT bring-forward_NN is_VBZ activated_VBN ,_, etc._FW -RRB-_-RRB- ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu :

		1) appropriate_JJ → step_NN (aturan 74) 2) next_JJ → processing_NN (aturan 40) 3) etc._FW → - (aturan 53)
	Kata rancu (kata yang berwarna merah)	The manager then determines the next appropriate[1] step (close, assign to another case worker for the next[2] processing step, put aside until a bring-forward is activated, etc.[3]).
	Kandidat kata rekomendasi	1) - 2) LABEL 3) and LABEL
6	Pernyataan kebutuhan	A case worker selects cases from a pool to perform a specific function, and then moves that case to the next appropriate step (close, put in another pool, put aside until a bring-forward is activated, etc.).
	Hasil penandaan tiap bagian kata (*POS Tagging*)	A_DT case_NN worker_NN selects_VBZ cases_NNS from_IN a_DT pool_NN to_TO perform_VB a_DT specific_JJ function_NN ,_, and_CC then_RB moves_VBZ that_IN case_NN to_TO the_DT next_JJ appropriate_JJ step_NN -LRB-_-LRB- close_NN ,_, put_VBN in_IN another_DT pool_NN ,_, put_VBD aside_RB until_IN a_DT bring-forward_NN is_VBZ activated_VBN ,_, etc._FW -RRB-_-RRB- ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) specific_JJ → function_NN (aturan 34) 2) that_IN → case_NN (aturan 159) 3) appropriate_JJ → step_NN (aturan 74) 4) etc._FW → - (aturan 53)
	Kata rancu (kata yang berwarna merah)	A case worker selects cases from a pool to perform a specific[1] function, and then moves that[2] case to the next appropriate[3] step (close, put in another pool, put aside until a bring-forward is activated, etc.[4]).
	Kandidat kata rekomendasi	1) -, LABEL 2) LABEL 3) -, LABEL 4) and LABEL
7	Pernyataan kebutuhan	Finally, work-flow support also means providing the ability to Approve or Reject particular request regarding a case.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	Finally_RB ,_, work-flow_NN support_NN also_RB means_VBZ providing_VBG the_DT ability_NN to_TO Approve_VB or_CC Reject_VB particular_JJ request_NN regarding_VBG a_DT case_NN ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) Finally_RB → , (aturan 33) 2) also_RB → means_VBZ (aturan 64) 3) particular_JJ → request_NN (aturan 34)
	Kata rancu (kata yang berwarna merah)	Finally[1], work-flow support also[2] means providing the ability to Approve or Reject particular[3] request regarding

		a case.
	Kandidat kata rekomendasi	1) - 2) - 3) -, LABEL
8	Pernyataan kebutuhan	A case may involve a single requests requiring approval, and security may be needed to restrict who can make these decisions, depending on the organization building the application using this framework.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	A_DT case_NN may_MD involve_VB a_DT single_JJ requests_NNS requiring_VBG approval_NN ,_, and_CC security_NN may_MD be_VB needed_VBN to_TO restrict_VB who_WP can_MD make_VB these_DT decisions_NNS ,_, depending_VBG on_IN the_DT organization_NN building_VBG the_DT application_NN using_VBG this_DT framework_NN ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) single_JJ → requests_NNS (aturan 12) 2) who_WP → can_MD (aturan 173) 3) these_DT → decisions_NNS (aturan 164) 4) this_DT → framework_NN (aturan 129)
	Kata rancu (kata yang berwarna merah)	A case may involve a single[1] requests requiring approval, and security may be needed to restrict who[2] can make these[3] decisions, depending on the organization building the application using this[4] framework.
	Kandidat kata rekomendasi	1) one, two, three 2) user, actor, worker, manager, LABEL 3) LABEL 4) LABEL
9	Pernyataan kebutuhan	Changing to a particular status may cause an event to be created automatically, such as creating a bring-forward entry due at some future date.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	Changing_VBG to_TO a_DT particular_JJ status_NN may_MD cause_VB an_DT event_NN to_TO be_VB created_VBN automatically_RB ,_, such_JJ as_IN creating_VBG a_DT bring-forward_JJ entry_NN due_JJ at_IN some_DT future_JJ date_NN ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) particular_JJ → status_NN (aturan 34) 2) some_DT → future_JJ (aturan 10) 3) future_JJ → date_NN (aturan 40)
	Kata rancu (kata yang berwarna merah)	Changing to a particular[1] status may cause an event to be created automatically, such as creating a bring-forward entry due at some[2] future[3] date.
	Kandidat kata rekomendasi	1) LABEL 2) two, three, four 3) -
10	Pernyataan kebutuhan	Changing to a particular status may also automatically close a bring-forward (whether it is due or not).

	Hasil penandaan tiap bagian kata (*POS Tagging*)	Changing_VBG to_TO a_DT particular_JJ status_NN may_MD also_RB automatically_RB close_VB a_DT bring-forward_NN -LRB-_-LRB- whether_IN it_PRP is_VBZ due_JJ or_CC not_RB -RRB-_-RRB- ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) particular_JJ → status_NN (aturan 34) 2) also_RB → automatically_RB (aturan 64) 3) it_PRP → is_VBZ (aturan 51)
	Kata rancu (kata yang berwarna merah)	Changing to a particular[1] status may also[2] automatically close a bring-forward (whether it[3] is due or not).
	Kandidat kata rekomendasi	1) LABEL 2) - 3) Status, bring-forward, LABEL
11	Pernyataan kebutuhan	Bring-Forward entries are reminders to a case worker that it is time to perform a certain.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	Bring-Forward_JJ entries_NNS are_VBP reminders_NNS to_TO a_DT case_NN worker_NN that_IN it_PRP is_VBZ time_NN to_TO perform_VB a_DT certain_JJ ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) it_PRP → is_VBZ (aturan 51) 2) certain_JJ → ._. (aturan 59)
	Kata rancu (kata yang berwarna merah)	Bring-Forward entries are reminders to a case worker that it[1] is time to perform a certain[2].
	Kandidat kata rekomendasi	1) -, REVISED_SENTENCE 2) LABEL
12	Pernyataan kebutuhan	If case worker are geographically dispersed, then you may assume that the cases are managed by geographic regions.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	If_IN case_NN worker_NN are_VBP geographically_RB dispersed_VBN ,_, then_RB you_PRP may_MD assume_VB that_IN the_DT cases_NNS are_VBP managed_VBN by_IN geographic_JJ regions_NNS ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) you_PRP → may_MD (aturan 179) 2) that_IN → the_DT (aturan 159) 3) geographic_JJ → regions_NNS (aturan 36)
	Kata rancu (kata yang berwarna merah)	If case worker are geographically dispersed, then you[1] may assume that[2] the cases are managed by geographic[3] regions.
	Kandidat kata rekomendasi	1) user, actor, manager, worker, LABEL 2) - 3) geographical
13	Pernyataan kebutuhan	The production is planned at factory level and is represented by lots assign to the FMS (a lot is an administrative entity indicating a group of identical pieces to be manufactured together). Several lots of different types of pieces can be in production at the same time in a

		given FMS.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	The_DT production_NN is_VBZ planned_VBN at_IN factory_NN level_NN and_CC is_VBZ represented_VBN by_IN lots_NNS assign_VBP to_TO the_DT FMS_NN -LRB-_-LRB- a_DT lot_NN is_VBZ an_DT administrative_JJ entity_NN indicating_VBG a_DT group_NN of_IN identical_JJ pieces_NNS to_TO be_VB manufactured_VBN together_RB -RRB-_-RRB- ._. Several_JJ lots_NNS of_IN different_JJ types_NNS of_IN pieces_NNS can_MD be_VB in_IN production_NN at_IN the_DT same_JJ time_NN in_IN a_DT given_VBN FMS_NN ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) Several_JJ→ lots_NNS (aturan 12) 2) different_JJ → types_NNS (aturan 12) 3) same_JJ → time_NN (aturan 25)
	Kata rancu (kata yang berwarna merah)	The production is planned at factory level and is represented by lots assign to the FMS (a lot is an administrative entity indicating a group of identical pieces to be manufactured together). Several[1] lots of different[2] types of pieces can be in production at the same[3] time in a given FMS.
	Kandidat kata rekomendasi	1) two, three, four 2) two, three, four 3) a, within a second
14	Pernyataan kebutuhan	Machines are assigned to a waiting piece (if any) as soon as they become idle.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	Machines_NNS are_VBP assigned_VBN to_TO a_DT waiting_VBG piece_NN -LRB-_-LRB- if_IN any_DT -RRB-_-RRB- as_RB soon_RB as_IN they_PRP become_VBP idle_JJ ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) any_DT → RRB-_-RRB- (aturan 10) 2) they_PRP → become_VBP (aturan 165)
	Kata rancu (kata yang berwarna merah)	Machines are assigned to a waiting piece (if any[1]) as soon as they[2] become idle.
	Kandidat kata rekomendasi	1) at least one 2) the machines, the LABEL
15	Pernyataan kebutuhan	For our purposes (as is the case in a small FMS), the time needed to serve a transport request is small relative to the time needed for a machine operation.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	For_IN our_PRP$ purposes_NNS -LRB-_-LRB- as_RB is_VBZ the_DT case_NN in_IN a_DT small_JJ FMS_NN -RRB-_-RRB- ,_, the_DT time_NN needed_VBN to_TO serve_VB a_DT transport_NN request_NN is_VBZ small_JJ relative_JJ to_TO the_DT time_NN needed_VBN for_IN a_DT machine_NN operation_NN ._.

	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) our_PRP$ → purposes_NNS (aturan 152) 2) small_JJ → FMS_NN (aturan 37) 3) small_JJ → relative_JJ (aturan 37)
	Kata rancu (kata yang berwarna merah)	For our[1] purposes (as is the case in a small[2] FMS), the time needed to serve a transport request is small[3] relative to the time needed for a machine operation.
	Kandidat kata rekomendasi	1) LABEL 2) at most two, at most three, at most four 3) at most two second, at most three second, at most four second
16	Pernyataan kebutuhan	Two turning machines, M1 and M2, and a drilling machine, M3, each of which can work on a single piece at a time.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	Two_CD turning_VBG machines_NNS ,_, M1_NN and_CC M2_NN ,_, and_CC a_DT drilling_NN machine_NN ,_, M3_NN ,_, each_DT of_IN which_WDT can_MD work_VB on_IN a_DT single_JJ piece_NN at_IN a_DT time_NN ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) each_DT → of_IN (aturan 57) 2) which_WDT → can_MD (aturan 171) 3) single_JJ → piece_NN (aturan 13)
	Kata rancu (kata yang berwarna merah)	Two turning machines, M1 and M2, and a drilling machine, M3, each[1] of which[2] can work on a single[3] piece at a time.
	Kandidat kata rekomendasi	1) one 2) ‘M1, M2, or M3’ 3) one
17	Pernyataan kebutuhan	If the piece is finished, it is transported to Store_out; otherwise, the Dispatcher checks if there is an idle machine performing the next operation on the piece.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	If_IN the_DT piece_NN is_VBZ finished_VBN ,_, it_PRP is_VBZ transported_VBN to_TO Store_out_NNP ;_: otherwise_RB ,_, the_DT Dispatcher_NNP checks_NNS if_IN there_EX is_VBZ an_DT idle_JJ machine_NN performing_VBG the_DT next_JJ operation_NN on_IN the_DT piece_NN ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) it_PRP → is_VBZ (aturan 51) 2) otherwise_RB → ,_, (aturan 67) 3) next_JJ → operation_NN (aturan 40)
	Kata rancu (kata yang berwarna merah)	If the piece is finished, it[1] is transported to Store_out; otherwise[2], the Dispatcher checks if there is an idle machine performing the next[3] operation on the piece.
	Kandidat kata rekomendasi	1) the piece, the LABEL 2) if the pieace is not finished, else 3) LABEL

18	Pernyataan kebutuhan	When a piece is evacuated from drilling machine M3, the machine becomes idle and it is ready to perform an operation on another piece.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	When_WRB a_DT piece_NN is_VBZ evacuated_VBN from_IN drilling_VBG machine_NN M3_NN ,_, the_DT machine_NN becomes_VBZ idle_JJ and_CC it_PRP is_VBZ ready_JJ to_TO perform_VB an_DT operation_NN on_IN another_DT piece_NN ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) it_PRP → is_VBZ (aturan 51) 2) ready_JJ → to_TO (aturan 38)
	Kata rancu (kata yang berwarna merah)	When a piece is evacuated from drilling machine M3, the machine becomes idle and it[1] is ready[2] to perform an operation on another piece.
	Kandidat kata rekomendasi	1) the piece, the LABEL 2) able, capable
19	Pernyataan kebutuhan	When the machine enters the IDLE state, the dispatcher is informed and checks whether a piece (in the store for raw pieces Store_in or in the temporary in process pieces Store_tmp) is waiting for the next operation on the drilling machine M3.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	When_WRB the_DT machine_NN enters_VBZ the_DT IDLE_JJ state_NN ,_, the_DT dispatcher_NN is_VBZ informed_VBN and_CC checks_NNS whether_IN a_DT piece_NN -LRB-_-LRB- in_IN the_DT store_NN for_IN raw_JJ pieces_NNS Store_in_NN or_CC in_IN the_DT temporary_JJ in_IN process_NN pieces_NNS Store_tmp_NN -RRB-_-RRB- is_VBZ waiting_VBG for_IN the_DT next_JJ operation_NN on_IN the_DT drilling_NN machine_NN M3_NN ._.
	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) temporary_JJ → in_IN (aturan 89) 2) next_JJ → operation_NN N (aturan 40)
	Kata rancu (kata yang berwarna merah)	When the machine enters the IDLE state, the dispatcher is informed and checks whether a piece (in the store for raw pieces Store_in or in the temporary[1] in process pieces Store_tmp) is waiting for the next[2] operation on the drilling machine M3.
	Kandidat kata rekomendasi	1) LABEL 2) LABEL
20	Pernyataan kebutuhan	The specification can be adjusted for different levels of complexity applying one or more of the following constraints.
	Hasil penandaan tiap bagian kata (*POS Tagging*)	The_DT specification_NN can_MD be_VB adjusted_VBN for_IN different_JJ levels_NNS of_IN complexity_NN applying_VBG one_CD or_CC more_JJR of_IN the_DT following_VBG constraints_NNS ._.

	Deskripsi Kerancuan	Kata dan pola yang rancu : 1) different_JJ → levels_NNS (aturan 12)
	Kata rancu (kata yang berwarna merah)	The specification can be adjusted for different[1] levels of complexity applying one or more of the following constraints.
	Kandidat kata rekomendasi	1) one to ten, a to z