Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) DOI: 10.25126/jtiik.201851632
Vol. 5, No. 1, Maret 2018, hlm. 121-127 p-ISSN: 2355-7699
Akreditasi RISTEKDIKTI, No 51/E/KPT/2017 e-ISSN: 2528-6579

PERHITUNGAN NILAI KOHESI CLASS DENGAN PENDEKATAN SEMANTIK DENGAN MEMPERTIMBANGKAN ARTEFAK DESAIN

Bayu Priyambadha[1], Fajar Pradana[2]

[1]Jurusan Teknik Informatika, Fakultas Ilmu Kompauter, Universitas Brawijaya, Malang
[2] Jurusan Sistem Informasi, Fakultas Ilmu Komputer, Universitas Brawijaya, Malang
Email: [1] bayu_priyambadha@ub.ac.id, [2]fajar.p@ub.ac.id

(Naskah masuk: 24 Januari 2018, diterima untuk diterbitkan: 8 Maret 2018)

Abstrak

Rekayasa perangkat lunak bertujuan memberikan sebuah cara atau metode untuk membangun sebuah sistem perangkat lunak yang berkualitas. Kualitas perangkat lunak yang dikembangkan tidak hanya bertumpu pada satu tahap saja, melainkan kualitas harus dijaga pada setiap tahapan sehingga perangkat lunak yang dihasilkan dapat mencapai kualitas yang baik. Salah satu proses untuk mewujudkan hasil perangkat lunak yang berkualitas dapat dilakukan pada fase perancangan sistem. Kohesi adalah salah satu indikator untuk menilai kualitas dari sebuah hasil perancangan. Perhitungan nilai kohesi dilakukan dengan melihat keterkaitan antara atribut dan metode yang ada di dalam sebuah klas. Sebuah metode diasumsikan mempunyai hubungan yang erat apabila tipe parameter memiliki kesamaan dengan tipe atribut yang dimiliki oleh klas tersebut. Kesamaan tipe parameter dan atribut tidak selalu menandakan bahwa atribut tersebut dikelola di dalam metode. Penelitian ini berupaya untuk menggali sebuah informasi yang dapat meningkatkan tingkat kepastian dari keterkaitan antara metode dan atribut dalam klas. Kesamaan penamaan tidak hanya dilihat dari kesamaan penulisan tetapi dilihat dari kesamaan makna (semantik). Selain itu, juga akan dipertimbangkan artefak desain berupa gambaran algoritma untuk melakukan perhitungan kohesi. Tahapan penulaian dimulai dari studi literatur, pengumpulan data, perancangan algoritma dan sistem, perhitungan kohesi dengan pendekatan semantik, analisis hasil dan yang terakhir adalah penarikan kesimpulan. Setelah melakukan perhitungan kohesi hasil mengalami peningkatan, sehingga perlu mempertimbangkan algorima artefak. Nilai koefisien Kappa yang meningkat dari 0.001519 ke 0.347587.

Kata kunci: *Rekayasa Perangkat Lunak, Kualitas Perangkat Lunak, Kualitas Desain, Kohesi, Metrik Desain*

CALCULATION OF CLASS COHESION USING THE SEMANTIC APPROACH WITH CONSIDERING DESIGN ARTIFACT

Abstract

The software engineering provides a way or metode *to build a quality software system. Software quality is developed not only rely on one stage, but the quality of all stages must be guaranteed. One of the important phases to results of qualified software is design phase. With good design, the qualified software can be created. Cohesion is one of the indicators to assess the quality of a design. Calculation of cohesion values is done by looking at the relationship between attributes and* metodes *in a class. A* metode *is assumed to have a close relationship with attribute if it has the same as type as the attribute's type. But, the similar type of parameters and attributes does not necessarily signify these attributes is used in the* metode*. This research tries to obtain an information that can increase the level of certainty of the relationship between* metodes *and attributes in the class. The naming equation is not only seen from the similarity of syntax but also seen from the similarity of meaning (semantics). In addition, it will also consider design artifacts to perform cohesion calculations. After performing cohesion calculations the results have increased, so it is necessary to consider the algorima artifact. The value of the Kappa coefficient is increased from 0,001519 to 0,347587.*

Keywords: *Software Engineering, Software Quality, Design Quality, Cohesion, Design Metric*

1. PENDAHULUAN

Penjaminan kualitas perangkat lunak yang dikembangkan tidak hanya bertumpu pada satu tahap saja, melainkan kualitas harus dijaga kualitasnya pada setiap tahapan, agar perangkat lunak yang dihasilkan dapat mencapai kualitas yang baik (Al Dallal, 2007; Al Dallal dan Briand,2010). Salah satu proses untuk mewujudkan hasil perangkat lunak yang berkualitas dapat dilakukan pada fase perancangan sistem. Fase perancangan menghasilkan diskripsi tentang struktur perangkat lunak, model data dan struktur data, antarmuka antara komponen sistem, dan algoritma yang digunakan (Sommerville, 2011). Kualitas perancangan mempengaruhi hasil akhir perangkat lunak yang dihasilkan. Kohesi adalah salah satu indikator untuk menilai kualitas dari sebuah hasil

perancangan (Al Dallal, 2007; Al Dallal dan Briand, 2010; Chowdhury dan Zulkernine, 2011).

Kohesi adalah keterkaitan antara elemen di dalam sebuah komponen (Al Dallal, 2007). Semakin tinggi nilai kohesi dalam sebuah komponen, maka perancangan semakin baik (Chen,dkk., 2002; Pressman, 2015). Tingginya tingkat kohesi pada sebuah komponen menunjukkan bahwa tingkat ketekaitan antara elemen tinggi. Sebagai contoh dalam pengembangan perangkat lunak dengan pendekatan berorientasi obyek, terdapat klas yang merupakan komponen, elemen yang ada di dalam klas antara lain adalah attribute dan metode (Al Dallal, 2007). Kedua elemen dikatakan berkaitan apabila sebuah atribut dikelola pada satu atau lebih metode yang ada di dalam klas tersebut. Semakin banyak atribut yang dikelola sendiri oleh metode di dalam klas yang sama, semakin tinggi pula tingkat kemandirian klas tersebut. Artinya, klas tersebut memiliki sumber daya sendiri untuk dikelola tanpa bergantung dengan sumber daya klas yang lain. Hal tersebut dapat meningkatkan kemudahan dalam mengelola klas yang ada di dalam sistem. Perubahan pada satu klas tidak akan mempengaruhi klas yang lain. Sehingga, pemeliharaan terhadap sistem dapat dilakukan dengan mudah dan fokus pada masalah.

Proses perhitungan nilai kohesi pada sebuah klas akan sangat berguna dalam menjaga kualitas perancangan (Al Dallal, 2007; Al Dallal dan Briand, 2010). Perhitungan tersebut dilakukan pada fase perancangan. Perhitungan kohesi pada tahap perancangan bertujuan untuk memberikan informasi tentang kualitas perancangan dari nilai kohesi sedini mungkin. Proses pemeliharaan kualitas kode pada tahap implementasi memerlukan biaya dan upaya jauh lebih besar dibandingkan dilakukan pada fase perancangan (Al Dallal, 2007). Oleh karena itu, apabila kualitas perancangan sistem dapat diketahui sedini mungkin, maka dapat dimungkinkan ada penghematan terhadap biaya dan upaya dari pengembang untuk melakukan pemeliharaan.

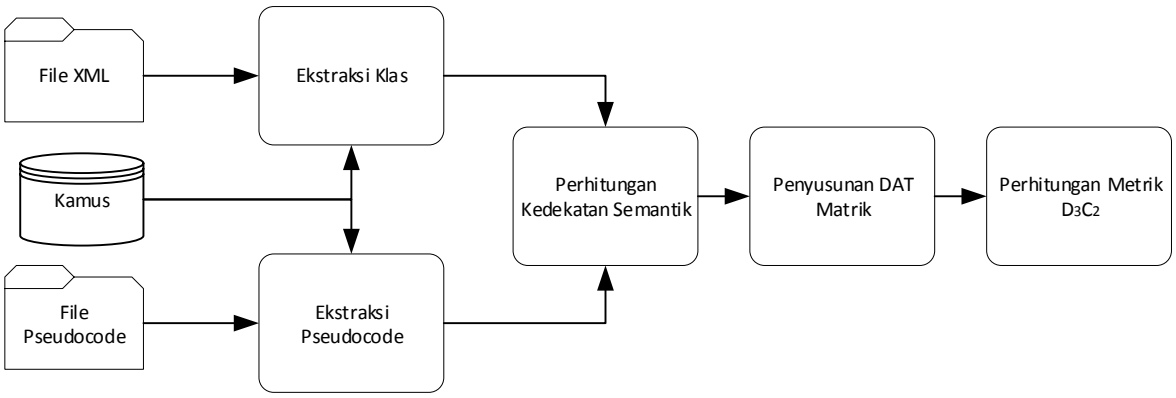
Informasi dan makna adalah dua hal yang sangat penting dalam kehidupan. Sesuatu dapat dikatakan bermakna apabila informasi dapat diinterpretasikan dari sesuatu tersebut (Menant, 2010). Informasi adalah bentukan data yang dikemas sedemikian rupa. Dengan demikian, data, informasi dan makna adalah hal-hal yang saling berkaitan. Konsep Piercean yang dituliskan oleh Menant (2010), membahas tentang sebuah konsep tanda atau informasi. Teori tersebut menyatakan bahwa terdapat tiga komponen dalam konsep Piercean, yaitu objek pemikiran, tanda/ informasi yang merepresentasikan objek, dan penafsir (*interpretant*). Semakin banyak informasi yang dapat dipertimbangkan dalam berbagai bentuk maka akan dapat pula mempengaruhi hasil interpretasi.

Perhitungan kohesi yang dilakukan pada tahap perancangan memiliki tingkat kesulitan tersendiri. Pada tahap ini, informasi yang dapat didapatkan sangatlah terbatas, pada fase ini bentukan sistem masih digambarkan dalam bentuk yang abstrak. Perhitungan nilai kohesi sebuah klas yang dilakukan oleh Al Dallal (2007) berdasarkan diagram klas yang telah dihasilkan pada tahap perancangan. Metrik yang diusulkan adalah *Distance Design-based Direct Class Cohession* (D_3C_2). Dasar perhitungan adalah kesamaan tipe data yang dimiliki oleh atribut klas dan parameter masukan pada metode klas. Dari hasil ini, terdapat ketidak-pastian dikelolanya atribut pada sebuah metode klas. Kondisi ini telah diperbaiki yang menambahkan aspek semantik pada metode perhitungan metrik D_3C_2. Hal yang dapat dikorelasikan antara atribut dan metode adalah penamaan atribut dan metode. Kesamaan makna antara atribut dan metode menunjukkan dan meningkatkan nilai kepastian sebuah atribut dikelola oleh metode klas.

Telah diketahui bahwa artefak yang dihasilkan pada fase perancangan tidak hanya klas diagram saja. Artefak desain lain yang dihasilkan yang bertujuan untuk memperjelas definisi klas adalah desain algoritma. Desain algoritma dapat dituangkan dalam bentuk pseudocode ataupun flowchart. Peran desain algoritma dalam tahap perancangan adalah untuk mendefinisikan alur proses yang terjadi pada setiap metode klas. Kemudian, hasil desain ini akan diimplementasikan pada tahap selanjutnya untuk diubah menjadi kode. Berdasarkan hubungan makna (semantik) dan informasi yang dituliskan oleh Menant (2010), maka penelitian tentang perhitungan metrik D_3C_2 berdasarkan pendekatan semantik akan dikembangkan. Informasi yang dipertimbangkan dalam perhitungan metrik D_3C_2 diperbanyak. Semakin banyak informasi yang dipertimbangkan diasumsikan tingkat kepastian sebuah atribut akan dikelola pada sebuah metode klas akan semakin tinggi. Dalam penelitian ini, gambaran algoritma dari hasil rancangan akan dipertimbangkan. Gambaran algoritma akan dijadikan salah satu faktor yang mempengaruhi nilai metrik D_3C_2. Perhitungan kesamaan makna tidak hanya berdasarkan kesamaan tipe data, dan arti nama atribut dan metode saja, melainkan informasi yang terkait dengan atribut yang dikemukakan di dalam gambaran algoritma juga dipertimbangkan.

2. METODE

Terdapat beberapa tahapan yang dilakukan untuk merealisasikan penelitian perhitungan nilai kohesi klas dengan pendekatan semantik. Tahapan tersebut akan dilakukan secara berurutan yang dimulai dari studi literatur, pengumpulan data, perancangan algoritma dan sistem, perhitungan kohesi dengan pendekatan semantik, analisis hasil dan yang terakhir adalah penarikan kesimpulan. Gambaran urutan alur penelitian tergambar pada Gambar 1.



Gambar 1. Alur Perhitungan Metrik D_3C_2

2.1. Ekstraksi Klas

Ekstraksi klas dilakukan untuk mendapatkan informasi tentang klas. Dalam penelitian ini informasi yang diperlukan dalam sebuah klas adalah nama klas, nama metode (beserta parameternya), dan nama atribut klas. Untuk informasi atribut dan parameter metode disertai dengan informasi tipe datanya. Proses ekstraksi ini dilakukan pada gambaran diagram klas. Gambaran diagram klas disimpan dalam sebuah file .xml. Dari file .xml ini proses ekstraksi akan dilakukan.

Setelah informasi tersebut didapatkan, maka proses selanjutnya adalah melakukan pemecahan (split). Proses pemecahan dilakukan pada nama metode dan nama atribut. Pemecahan ini memiliki tujuan untuk memecahkan nama metode atau atribut per kata. Sehingga akan lebih mudah untuk dibandingkan.

2.2. Ekstraksi *Pseudocode*

Proses ekstraksi tidak hanya dilakukan pada diagram klas saja, gambaran pseudocode juga dipertimbangkan. Dari sebuah pseudocode yang menggambarkan algoritma sebuah metode akan diambil beberapa informasi, seperti nama variable yang digunakan dalam pseudocode tersebut. Dari nama variable tersebut, seperti halnya ekstraksi klas, akan dipecah (split) tiap kata (tokenisasi).

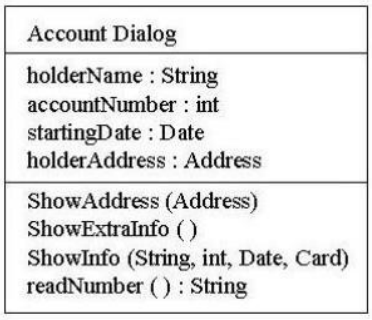
Kata-kata hasil pemecahan dari pseudocode akan dibandingkan dengan kamus untuk diseleksi mana kata yang memiliki arti. Kemudian, akan dibandingkan dengan nama atribut pada sebuah klas. Jika kata dalam pseudocode memiliki kesamaan makna dengan nama atribut dalam klas, kata tersebut akan dimasukkan ke dalam kata-kata yang berasosiasi dengan metode untuk dipertimbangkan.

2.3. Perhitungan Kedekatan Semantik

Setiap kata hasil proses ektraksi klas dan ekstraksi pseudocode akan dibandingkan satu per satu untuk dilakukan analisis secara semantik. Nilai kedekatan semantik setiap pasang kata disimpan, kemudian nilai ini akan menjadi pertimbangan dalam penyusunan DAT Matrik pada proses selanjutnya.

2.4. Penyusunan DAT Matrik

DAT Matrik adalah hal yang sangat penting dalam proses perhitungan metric D_3C_2. Matrik ini menjelaskan keterkaitan secara semantik antara metode dan atribut. Metode diwakili oleh nama metode, parameter metode, dan *pseudocode*. Perbandingan makna antara nama atribut dan metode akan menentukan keterkaitan antara kedua hal tersebut yang tergambar pada DAT Matrik.



Gambar 2. Contoh Klas

Matrik DAT membandingkan metode-metode yang ada pada klas dengan type dari atribut dalam klas yang unik (tidak ada redundansi tipe atribut). Metode yang mengandung (memiliki parameter bertipe sama atau return value sama) tipe data dari atribut maka akan memiliki nilai 1. Namun, dalam pendekatan semantik, perlu difikirkan bentuk matrik DAT yang sesuai dan dapat digunakan untuk melakukan perhitungan dengan pendekatan semantik. Dalam kasus diatas (Gambar 2), bentuk matrik DAT yang akan digunakan dalam penelitian ini tergambar dalam Gambar 3.

	holderName: String	accountNumber: int	stringDate: Date	holderAddress: Address
ShowAddress	0	0	0	1
ShowExtraInfo	0	0	0	0
ShowInfo	1	1	1	1
readNumber	1	0	0	0

Gambar 3. DAT Matrik

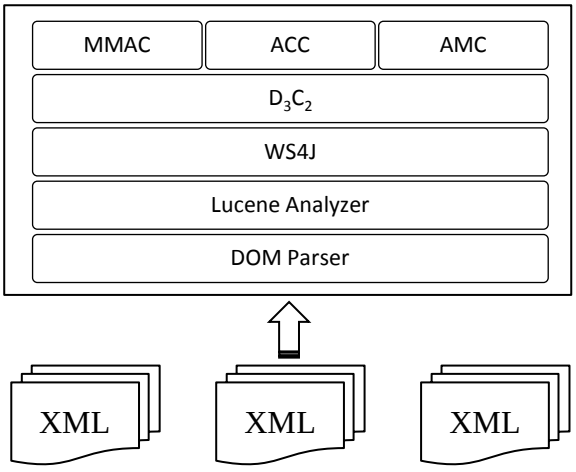
2.5. Perhitungan Metrik D_3C_2

Setelah DAT Matrik selesai disusun, maka proses selanjutnya adalah melakukan perhitungan metrik D_3C_2. Perhitungan ini menggunakan rumus metrik yang diusulkan oleh Al Dalal. Hasil perhitungan diperlukan untuk proses analisis yang menjadi dasar proses penalikan kesimpulan dari penelitian ini. Terdapat kemungkinan DAT matrik yang disusun berbeda dengan matrik yang disusun oleh Al Dalal dalam proses perhitungan metrik D_3C_2. Proses ini tidak dapat didefinisikan apabila suatu kelas program tidak memiliki metodes dan attributes. Metrik D_3C_2 digunakan melakukan pembobotan nilai untuk menghasilkan penjumlahan akhir dari metode MMAC, AAC, dan AMC (Al Dallal, 2007).
Berikut merupakan rumus dari metrik D_3C_2:

$$D_3C_2 = \begin{cases} 0 & \text{if } k = 0 \text{ and } l = 1, \\ 1 & \text{if } k = 0 \text{ and } l = 0, \\ \frac{k(k-l)MMAC+l(l-1)AAC+2lkAMC}{k(k-l)+l(l-1)+2lk} & otherwise. \end{cases} \quad (1)$$

3. PERANCANGAN SISTEM

Sistem perhitungan metrik D_3C_2 dengan pendekatan semantik dibangun dengan menggunakan bahasa pemrograma Java. Beberapa library digunakan untuk membangun sistem tersebut, antara lain, WS4J dan Lucene Analyzer. Gambaran arsitektur sistem perhitungan metrik D_3C_2 dengan pendekatan semantik dijelaskan pada Gambar 4.



Gambar 4. Arsitektur Sistem Perhitungan Metrik D_3C_2

Sistem perhitungan metrik D_3C_2 memiliki kebutuhan input berupa file xml. File-file xml didapatkan dari konversi dari source code menggunakan aplikasi Visual Paradigm. Ekstraksi informasi yang ada pada file xml yang berkaitan dengan nama klas, nama atribut, nama metode dan nama type data dilakukan dengan melakukan parsing data menggunakan DOM Parser. Data tersebut akan dipecah berdasarkan kata dan suku kata yang kemudian dilakukan analisis makna dengan menggunakan WS4J (Wu Palmer). Pemecahan kata atau suku kata dengan proses tokenisasi menggunakan Lucene Analyzer. Kedekatan makna akan menjadi pertimbangan dalam pembuatan matrik DAT. Matrik DAT dengan pendekatan semantic akan menjadi dasar dalam perhitungan metrik MMAC, ACC dan AMC yang kemudian akan bahan perhitungan D_3C_2.

4. UJI COBA SISTEM

Uji coba pada sistem dilakukan dengan menjalankan sistem untuk melakukan perhitungan pada kasus riil. Kasus yang diangkat adalah kasus riil yang ada pada repositori kode pada website sourceforge.net.

4.1. Data Uji Coba

Data uji yang digunakan adalah digram klas yang didapatkan dari sebuah kode sumber pada repositori. Kendala yang ada dalam mendapatkan diagram klas ini adalah tidak adanya dokumen pengembangan sistem yang dibagikan secara bebas. Oleh karena itu, dalam penelitian ini digunakan proses reverse engineering untuk membangkitkan sebuah diagram klas dari sebuah sistem aplikasi open source. Data tersebut akan digunakan pada proses uji coba sistem. Proses uji coba dibagi menjadi dua, yaitu uji coba awal untuk meyakinkan keterkaitan makna antara nama atribut, metode dan parameter. Percobaan kedua adalah percobaan sistem secara keseluruhan dengan menghitung secara langsung metrik D_3C_2 berdasarkan pendekatan semantik.

Data yang dipilih untuk uji coba ini adalah kode sumber dari aplikasi jDraw versi 1.1.5 yang ada pada website www.sourceforge.net. Penelitian ini bekerja pada fase perancangan, oleh karena itu, data uji yang dibutuhkan adalah representasi perancangan sistem aplikasi jDraw. Namun data tersebut tidak tersedia sehingga perlu ada proses untuk melakukan konversi dari sebuah kode sumber menjadi salah satu diagram produk dari fase perancangan. Diagram yang dipilih adalah diagram klas. Proses konversi dari kode sumber menjadi diagram klas dilakukan menggunakan aplikasi Visual Paradigm versi 8. Aplikasi tersebut sering digunakan untuk proses perancangan sistem oleh para pengembang sistem aplikasi. Hasil proses konversi dieksport dalam bentuk file .xml agar lebih standar. Klas-klas yang akan digunakan pada percobaan ini dijelaskan pada Tabel 1 berikut ini.

Tabel 1. Daftar Klas untuk Uji Coba

No	Klas
1	ColourEntry
2	ChangingData
3	Clip
4	ColourEntry
5	ColourSettings
6	DataChangeListener

7	DataObject
8	Update
9	Frame
10	MinMax
11	FrameSettings
12	Palette
13	ColourList
14	UsageComparator
15	Picture
16	Pixel
17	ChangeEvent
18	EventConstants

Pseudocode sebagai artefak desain yang lain yang juga akan dipertimbangkan dalam penelitian ini didapatkan dengan cara reverse engineering. Bahan yang digunakan untuk melakukan reverse engineering adalah kode program. Kode program dikonversi menjadi pseudocode dengan menggunakan alat bantu online. Alat bantu online yang digunakan untuk melakukan konversi kode menjadi pseudocode adalah https://pseudogen-2016.appspot.com/converter.html.

4.2. Skenario Uji Coba

Uji coba dilakukan dengan dua tahap, yaitu, uji coba dengan tidak mempertimbankan pseudocode dan mempertimbangkan pseudocode. Kedua uji coba dilakukan perbandingan dengan hasil yang didapatkan dari seorang pakar untuk mengetahui tingkat kecocokan (*agreement*) antara pakar dan sistem. Pakar yang dimaksud adalah seorang yang memahami tentang bahasa pemrograman Java. Dalam penelitian ini, dosen mata kuliah pemrograman Java dipilih untuk menjadi pakar yang melakukan analisis terhadap kode program yang ada sesuai dengan rancangan yang dijadikan studi kasus.

Uji coba pertama dilakukan dengan melakukan ekstraksi data atau informasi dari file XML. Data yang diambil adalah, nama klas, atribut dan metode beserta parameter dan tipe datanya. Kemudian, tahap selanjutnya dilakukan pemecahan label nama atribut dan metode dari sebuah klas. Hasil dari pemecahan label nama berbentuk kumpulan kata yang akan dibandingkan antara kata-kata dari atribut dan metode untuk mengetahui kedekatan makna setiap pasangan kata. Matrik DAT dibentuk berdasarkan atas atribut dan metode yang diambil dari xml. Setelah matrik DAT terbentuk, perhitungan metrik D_3C_2 dilakukan yang kemudian hasilnya dibandingkan dengan hasil dari pakar.

Uji coba kedua dilakukan dengan proses awal yang sama dengan uji coba pertama. Sebuah pembeda antara uji coba pertama dan kedua terletak pada ekstraksi data dari pseudocode. Ekstraksi data dari pseudocode dilakukan hanya pada uji coba kedua. Proses ekstraksi pseudocode dilakukan setelah proses ekstraksi atribut dan metode klas. Kemudian proses dilanjutkan seperti halnya uji coba pertama.

Hasil dari uji coba pertama dan kedua akan dibandingkan. Proses perbandingan ini membandingkan antara hasil perhitungan koefisien Kappa yang dilakukan pada uji coba pertama dan kedua. Hasil perbandingan ini akan dianalisis pada bagian selanjutnya untuk mengetahui perbedaan kedua uji coba.

4.3. Hasil Uji Coba

Hasil perhitungan koefisien Kappa dilakukan berdasarkan hasil dari dua skenario uji coba. Hasil perhitungan koefisien Kappa untuk perhitungan dengan tidak mempertimbangkan *pseudocode* dijelaskan pada Tabel 2. Po adalah nilai kesepakatan relatif antara penilai, Pc adalah probabilitas dari nilai kesepatakan, Y adalah jumlah jawaban Ya, dan T adalah jumlah jawabana Tidak.

Tabel 2. Kappa Tanpa *Pseudocode*

KOHENS KAPPA			
	Pakar		
Sistem	**Y**	**N**	**Total**
Y	14	122	136
N	180	1600	1780
Total	194	1722	1916
Po	0,84238		
Pc	0,84214		
Kappa	**0,001519**	Rendah (less than chance agreement)	

Hasil pada tabel 2 didapatkan dengan cara melakukan perhitungan jumlah data yang true-positif (sistem dan pakar menyatakan bersifat kohesif), true-negatif (pakar menyatakan kohesif tapi sistem tidak), false-positif (sistem menyatakan kohesif tapi pakar tidak) dan false-negatif (pakar dan sistem sama-sama menyatakan tidak kohesif). Proses perhitungan dilakukan dengan cara membandingkan antara pakar dan sistem. Kemudian, dihitung dengan rumus perhitungan koefisien Kappa berdasarkan tabel 2. Pada hasil tersebut terlihat bahwa nilai Kappa adalah 0,001519. Nilai tersebut dapat diinterpretasikan sebagai rendah. Nilai cukup memiliki arti bahwa, tingkat kesepakatan (kesamaan hasil) antara sistem dan pakar adalah rendah dengan nilai 0,001519.

Hasil tersebut akan dibandingkan dengan hasil pada skenario ke dua. Pada skenario kedua koefisien Kappa dihitung dengan mempertimbangkan keberadaan pseudocode. Kata-kata yang ada dalam pseudocode akan memperkaya informasi yang dimiliki oleh sebuah metode. Jika kata-kata dalam pseudocode memiliki kedekatan dengan beberapa atribut pada klas tersebut, maka antara atribut dan metode memilki keterkaitan. Tabel 3 menjelaskan hasil dari perhitungan koefisien Kappa dengan mempertimbangkan *pseudocode.*

Tabel 3. Kappa dengan Mempertimbangkan Pseudocode

KOHENS KAPPA			
	Pakar		
Sistem	**Y**	**N**	**Total**

Y	92	151	243
N	102	1571	1673
Total	194	1722	1916
Po	0,867954		
Pc	0,797604		
Kappa	**0,347587**	Cukup (fair agreement)	

Pada hasil yang tercantum pada Tabel 3, koefisien Kappa yang didapatkan ketika pseudocode menjadi salah satu pertimbangan adalah 0,347587. Nilai tersebut dapat diinterpretasikan bahwa tingkat kesepakatan antara sistem dan pakar adalah cuku (fair agreement) dengan nilai 0,347587.

Skenari pertama dan kedua menghasilkan nilai yang berbeda. Dengan mempertimbangkan pseudocode, nilai koefisien Kappa terlihat terdapat selisih yang cukup besar. Nilai koefisien Kappa dengan pseudocode memiliki nilai yang lebih tinggi daripada tidak mempertimbangkan pseudocode.

5. PEMBAHASAN HASIL

Hasil perbandingan koefisien Kappa menunjukkan adanya peningkatan performa jika sistem mempertimbangkan pseudocode. Nilai kesepakatan antara sistem dan pakar meningkat. Hal ini menunjukkan semakin banyak data yang dihasilkan sama antara sistem dan pakar. Jumlah data true-positif meningkat dari 14 ke 92. Namun, pada data false-negatif, mengalami penurunan jumlah sebanyak 29 data. Di sisi lain true-negatif bertambah 29 dari 122 menjadi 151. Dengan kata lain, pendekatan semantic dengan mempertimbangkan pseudocode dapat meningkatkan 78 data. Namun di sisi lain, terdapat data yang mengalami perbedaan (awalnya sama) sebanyak 29 data.

Pada klas ColourEntry terdapat atribut yang bernama index dan metode yang bernama decreaseIndex(). Perhitungan dengan mengabaikan pseudocode menghasilkan nilai kedekatan semantik sebanyak 0,47058. Ambang batas yang digunakan pada proses ini adalah 0,5. Hasil perhitungan tersebut berada di bawah ambang batas. Oleh karena itu, makna antara atribut dan metode tersebut dinyatakan tidak sama.

Pada proses perhitungan kedekatan makna dengan mempertimbangkan pseudocode. Terdapat informasi yang dapat memperkaya informasi yang dimiliki oleh metode decreaseIndex(). Sehingga, hasil perhitungan dengan mempertimbangkan pseudocode adalah 0,5555. Berdasarkan nilai ambang batas yang digunakan, maka dengan pseudocode pasangan atribut dan metode tersebut memiliki makna yang sama. Hasil tersebut menunjukkan bahwa pseudocode dapat meningkatkan kepastian apakah sebuah atribut dikelola dalam sebuah metode. Jika dibandingkan dengan hasil analisis pakar, hasil analisis dari pakar mengatakan bahwa pasangan atribut dan metode tersebut adalah memiliki keterkaitan sehingga kasus ini dapat menambah data true-positif dalam perhitungan koefisien Kappa.

Di sisi lain, masih pada klas ColourEntry, terdapat pasangan antara atribut yang bernama "colour" dan metode bernama "hashString()". Hasil analisis dari pakar untuk pasangan ini adalah tidak memiliki keterkaitan. Pada perhitungan dengan mengabaikan pseudocode dihasilkan nilai sebesar 0,21428. Nilai tersebut dibawah ambang batas sehingga dapat diartikan sebagai tidak sama secara makna. Namun, pada perhitungan dengan mempertimbangkan pseudocode, pada proses ekstraksi kata dari pseudocode ditemukan sebuah kata berlabel "color". "color" memiliki kedekatan makna dengan "colour". Sehingga untuk perhitungan kedekatan makna antara "colour" dan "hashString()" menghasilkan nilai 0.68478. Jika dibandingkan dengan ambang batas, nilai tersebut berarda diatas ambang batas yang dapat diartikan kedua pasangan atribut dan metode tersebut dianggap sama secara makna. Kasus ini meningkatkan jumlah data true-negatif dalam perhitungan koefisien Kappa. Bertambahnya data true-negatif dalam perhitungan koefisien Kappa dapat menjadi factor menurunnya hasil koefisien. Diperlukan upaya lebih lanjut untuk menangani kasus seperti ini agar perhitungan dapat dilakukan dengan tingkat keakuratan yang semakin baik.

Di sisi lain, dengan memperhitungkan *pseudocode* sebagai artefak desain, proses perhitungan metrik D_3C_2 mengalami peningkatan nilai. Peningkatan nilai ini menunjukkan bahwa adanya peningkatan keakuratan dan kepastian bahwa sebuah atribut dikelola oleh metode.

6. KESIMPULAN

Proses ekstraksi algoritma yang tergambar dalam pseudocode dilakukan dengan pemecahan setiap kata dengan metode tokenisasi. Tokenisasi bertujuan untuk memecah sebuah kata yang merupakan gabungan dari dua atau lebih kata. Kemudian, kata-kata tersebut dibandingkan dengan kamus untuk mendeteksi kata-kata yang dihasilkan oleh tokenisasi memiliki makna atau tidak.

Dalam perhitungan metrik D_3C_2, terdapat dua unsur utama yang dibandingkan yaitu atribut dan metode. Dalam penelitian terdahulu, perhitungan metrik D_3C_2 dihitung dengan menggunakan sebuah matrik DAT. Dalam matrik tersebut, atribut dan metode yang memiliki keterkaitan diisikan nilai 1. Penanda keterkaitan antara atribut dan metode adalah kesamaan type. Dalam penelitian ini, terdapat modifikasi pada matrik DAT yang tidak hanya mengakomodir type data namun juga penamaan atribut dan metode, maupun pseudocode. Kata-kata penting yang ada didalam pseudocode akan menjadi pengaya makna untuk metode, dan kondisi tersebut akan sangat mempengaruhi kondisi matrik DAT dalam perhitungan metrik D_3C_2.

Dengan mempertimbangkan gambaran algoritma, hasil perhitungan metrik D_3C_2 mengalami peningkatan. Peningkatan tidak hanya dilihat dari

nilai perhitungan metrik D_3C_2 saja tetapi juga melihat peningkatan tingkat kesepakatan antara sistem dan pakar yang meningkat dari 0,001519 ke 0,347587.

DAFTAR PUSTAKA

CHEN, Z. YUMING ZHOU, AND BAOWEN XU 2002. 'A Novel Approach to Measuring Class Cohesion Based on Dependence Analysis', *International Conference on Software Maintenance*, pp. 377–384. doi: 10.1109/ICSM.2002.1167794.

CHOWDHURY, I. DAN ZULKERNINE, M. 2011. 'Using complexity, coupling, dan cohesion metrics as early indicators of vulnerabilities', *Journal of Systems Architecture*. Elsevier B.V., 57(3), pp. 294–313. doi: 10.1016/j.sysarc.2010.06.003.

AL DALLAL, J. 2007. 'A design-based cohesion metric for object-oriented classes', *International Journal of Computer, Electrical, Automation, Control and Information Engineering*, 1(3), pp. 195–200. Available at: http://139.141.170.133/drjehad/A design based cohesion.pdf.

AL DALLAL, J. AND BRIAND, L. C. 2010. 'An object-oriented high-level design-based class cohesion metric', *Information and Software Technology*. Elsevier B.V., 52(12), pp. 1346–1361. doi: 10.1016/j.infsof.2010.08.006.

MENANT, C. 2010. 'Introduction to a systemic theory of meaning', *5th EFFC*, pp. 1–9. Available at: http://www.researchgate.net/publication/28765331_Introduction_to_a_systemic_theory_of_meaning/file/f2faf4f6b208c59bad.pdf (Accessed: 5 April 2014).

PRESSMAN, R. S. 2015. *Software Engineering : A Practitioner's Approach*. Seventh Ed. McGraw-Hill.

SOMMERVILLE, I. 2011. *Software Engineering*. Addison-Wesley Professional. doi: 10.1111/j.1365-2362.2005.01463.x.