KAKAS BANTU PERHITUNGAN NILAI KOPLING MENGGUNAKAN METRIK COGNITIVE WEIGHTED COUPLING BETWEEN OBJECT (CWCBO)

Muhammad Ubaidillah[1)], Fajar Pradana[2)], Bayu Priyambadha[3)]

[1)]Universitas Brawijaya, Jl,Veteran Malang Jawa Timur (Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya)
[2)]Universitas Brawijaya, Jl,Veteran Malang Jawa Timur (Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya)
[3)]Universitas Brawijaya, Jl,Veteran Malang Jawa Timur (Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya)

e-mail: ubaidillah.m26@gmail.com

Abstract

Object Oriented Programming (OOP) is a programming concept that was built with a focus on some objects. OOP concepts are able to measuring the quality of software through a possible connection between multiple objects or coupling. A good software is a software that has a good design, one of its characteristics is having a low value of coupling. High coupling value could cause an increasingly complex and bad software design, and so will become to difficult to understand, especially when maintenance carried out. To improve the software's quality, it's necessary to control coupling's value in order to minimize the complexity of the software. However, the manually calculation of coupling's value on the sizeable amount of software will take an ample time and resources. Therefore, a tool to calculate coupling's value were made using Cognitive Weighted Coupling Between Object (CWCBO) metric. CWCBO metric is a coupling metric that based on the understanding weight to count the different types of coupling by various researchers. A tool was built with primary component from Java programming language, Spoon library to analyze source code and JFreeChart library to display charts. From testing results the system's accuracy on five Java program inputs obtained a 100% of accuracy. Testing is done by comparing the accuracy of the calculations manually and from system.

Keywords: OOP, Coupling, CWCBO, Source Code

Abstrak

Konsep Object Oriented Programming (OOP) merupakan pemrograman yang dibangun dengan berpusat pada beberapa objek. Dengan konsep OOP dapat dilakukan pengukuran kualitas perangkat lunak dengan melalui kemungkinan hubungan antar beberapa objek atau kopling. Perangkat lunak yang baik adalah perangkat lunak yang memiliki desain yang baik, salah satu ciri-cirinya adalah memiliki nilai kopling yang rendah. Nilai kopling yang tinggi dapat menyebabkan desain perangkat lunak yang semakin kompleks dan buruk, sehingga akan mengakibatkan sulit untuk dipahami, terutama ketika dilakukan maintenance. Untuk meningkatkan kualitas perangkat lunak perlu mengontrol nilai kopling agar dapat meminimalkan kompleksitas perangkat lunak. Namun perhitungan nilai kopling secara manual pada jumlah perangkat lunak yang banyak akan membutuhkan waktu dan sumber daya yang besar. Oleh karena itu dibuatlah kakas bantu perhitungan nilai kopling menggunakan metrik Cognitive Weighted Coupling Between Object (CWCBO). Metrik CWCBO merupakan metrik kopling yang didasarkan pada bobot pemahaman untuk menghitung perbedaan jenis kopling dari berbagai peneliti. Kakas bantu tersebut dibangun dengan bahasa pemrograman utama Java, library Spoon untuk menganalisis source code, dan library JFreeChart untuk menampilkan grafik. Dari hasil pengujian akurasi sistem pada 5 program inputan berbahasa Java didapatkan akurasi sebesar 100%. Pengujian akurasi dilakukan dengan membandingkan perhitungan secara manual dan dengan sistem.

Kata Kunci: OOP, Kopling, CWCBO, Source Code

1. PENDAHULUAN

Rekayasa Perangkat Lunak (RPL) merupakan suatu cabang ilmu profesi yang membahas tentang teknik-teknik dalam mengembangkan perangkat lunak, mulai dari perencanaan, pembuatan, pengujian hingga pemeliharaan. Rekayasa perangkat lunak merupakan tugas yang rumit dan kompleks (Aloysius & Arockiam, 2012). Struktur perangkat lunak yang kompleks bisa berdampak pada kualitas perangkat lunak yang buruk, karena semakin sulit untuk dipahami, pada proses *maintenance*. Sehingga banyak para pengembang berupaya menciptakan teknik-teknik untuk mempermudah dalam proses pengembangan dan

maintenance. Salah satu upayanya adalah menerapkan model-model pengembangan perangkat lunak, seperti pendekatan berorientasi objek dan pendekatan terstruktur.

Lebih dari dua dekade terakhir, penggunaan teknik berbasis objek lebih dominan dibandingkan dengan penggunaan teknik terstruktur untuk mengembangkan perangkat lunak (Misra & Akman, 2008). Perangkat lunak yang dibangun dengan menggunakan konsep *Object Oriented Programing* (OOP) akan dilakukan pemrograman yang berpusat pada beberapa objek. Dengan pemanfaatan konsep OOP dapat dilakukan pengukuran kualitas suatu perangkat lunak berdasarkan kemungkinan hubungan antar *atribute* dan *method* yang terdapat pada masing-masing *class* (Lestari, 2015). Proses ini dapat dilakukan sebagai alat dalam memutuskan kualitas suatu perangkat lunak pada tahap implementasi berdasarkan nilai kopling antar objek.

Perangkat lunak yang baik adalah perangkat lunak yang memiliki desain baik. Salah satu ciri-ciri desain yang baik adalah memiliki nilai kopling yang rendah. Nilai kopling mempunyai dampak yang negatif pada kualitas perangkat lunak (Yadav & Khan, 2009 dalam Aloysius & Arockiam, 2012). Apalagi kopling kelas yang tinggi dianggap sebagai desain yang buruk dan dapat menyebabkan kesulitan dalam memahami kelas (Stevens, et al., 1974). Perangkat lunak yang sulit dipahami juga meyebabkan kesulitan *developer* untuk melakukan *maintenance* pada sistem. Namun, jika perangkat lunak tersebut memiliki desain yang baik, maka akan lebih mudah untuk dipahami, sehingga mudah dilakukan pengelolaan. Untuk meningkatkan kualitas perangkat lunak harus mempertimbangkan nilai kopling untuk meminimalkan kompleksitas perangkat lunak (Yadav & Khan, 2009 dalam Aloysius & Arockiam, 2012).

Nilai kopling dapat menentukan tingkat kualitas dari desain perangkat lunak, maka diperlukan suatu perhitungan nilai kopling yang dapat menentukan kualitas perangkat lunak berdasarkan relasi-relasi antar objek dan juga dapat mendefinisikan tingkat kesulitan dalam memahami hubungan antar kelas pada suatu program. Oleh karena itu, dipilih lah perhitungan kualitas desain berorientasi objek menggunakan metrik *Cognitive Weighted Coupling Between Object* (CWCBO). Metrik tersebut dipilih karena menurut Aloysius dan Arockiam (2012), belum pernah ada metrik kopling yang didasarkan pada bobot pemahaman untuk menghitung perbedaan jenis kopling dari berbagai peneliti, sehingga metrik CWCBO yang dihitung berdasarkan bobot pemahaman diharapkan dapat mendefinisikan kopling pada berbagai tingkatan.

Adanya perhitungan tersebut dirasa masih kurang efektif dan efisien jika dilakukan perhitungan secara manual dengan jumlah perangkat lunak yang banyak. Selain membutuhkan waktu yang lama, proses tersebut juga membutuhkan sumber daya yang besar. Apalagi dalam menentukan nilai kopling tersebut dibutuhkan sumberdaya manusia yang benar-benar paham dengan model perhitungannya, sedangkan untuk mendapatkan sumberdaya tersebut tidaklah mudah.

Oleh karena itu, berdasarkan permasalahan yang telah dipaparkan, dibuat lah suatu program yang dapat melakukan otomatisasi perhitungan nilai kopling, sehingga diharapkan dapat membantu para pengembang dalam menentukan tingkat kualitas perangkat lunak secara efektif dan efisien. Perhitungan kualitas desain *Object-Oriented* ini menggunakan metrik kopling Cognitive *Weighted Coupling Between Object* (CWCBO) dan dibangun dengan menggunakan bahasa pemrograman Java. Dengan adanya sistem perhitungan kualitas ini nanti diharapkan dapat menjadi alternatif bagi *user* dalam mengembangkan perangkat lunak yang lebih baik. Terutama *source code* yang telah dibuat *user* akan menjadi lebih baik dan mudah untuk dipahami.

2. LANDASAN KEPUSTAKAAN

Landasan kepustakaan terdiri dari kajian pustaka atau literatur yang berhubungan dengan pembuatan Kakas Bantu Perhitungan Nilai Kopling Menggunakan Metrik *Cognitive Weighted Coupling Between Object* (CWCBO).

2.1 Kajian pustaka

Kajian pustaka pada penelitian ini membahas penelitian sebelumnya dengan ditulis oleh A. Aloysius dan L. Arockiam dengan judul jurnal "*Coupling Complexity Metric: A Cognitive Approach*". Penelitian tersebut membahas tentang rekayasa perangkat lunak dengan tujuan untuk mengembangkan teknik dan alat untuk meningkatkan kualitas perangkat yang stabil dan mudah dalam pemeliharaannya. Untuk menilai kualitas suatu perangkat lunak selama proses pengembangan dilakukan perhitungan nilai kopling secara otomatis dengan adanya alat bantu. Dari jurnal yang diusulkan Aloysius menjelaskan bahwa telah banyak metrik kopling untuk mengukur kopling antar objek atau kelas, namun tidak ada metrik kopling dengan bobot pemahaman untuk mengukur perbedaan jenis kopling oleh berbagai peneliti. Jadi Aloysius harapkan metrik kopling *Cognitive Weighted Coupling Between Object* (CWCBO) yang didasarkan pada bobot pemahaman dapat mendefinisikan kopling pada berbagai tingkatan.

Metrik CWCBO berisi berbagai macam kopling, yaitu *Control Coupling* (CC), *Global Data Coupling* (GDC), *Internal Data Coupling* (IDC), *Data Coupling* (DC) dan *Lexical Content Coupling* (LCC), dimana jenis-jenis kopling tersebut Aloysius tinjau dari usulan Edward Berard (Edward, 1993 dalam Aloysius & Arockiam, 2012). Bobot pemahaman pada jenis kopling tersebut dikalibrasikan melelui percobaan psikologis terhadap mahasiswa sarjana dan master (Aloysius & Arockiam, 2012). Adanya metrik tersebut diharapkan dapat mencerminkan kopleksitas dari sistem OO secara nyata. Dan hasil dari pengukuran tersebut akan mampu menunjukkan sebuah objek mempunyai keterkaitan yang sangat erat pada method di kelas lain (Aloysius &

Arockiam, 2012). Hal ini dapat menjadi sebuah alternatif yang lebih dalam mengukur kualitas suatu sistem berdasarkan perhitungan kopling antar objek secara lebih nyata. Prosedur dalam penelitian ini meggunakan perhitungan manual tanpa menggunakan tools yang mendukung pada proses kalkulasinya.

2.2 Metrik Coupling

Metrik kopling menunjukkan hubungan dan interaksi elemen-elemen antar objek pada suatu perangkat lunak. Semakin tinggi nilai kopling, maka hubungan antar objek atau modul pada suatu perangkat lunak semakin kuat dan semakin kompleks. Nilai kopling yang tinggi dapat berakibat pada semakin buruknya kualitas desain perangkat lunak, karena semakin banyak pertukaran pesan antar objek (Coad & Yourdon, 1991 dalam Misra & Akman, 2008). Selain itu desain perangkat lunak yang buruk juga dapat menyebabkan kesulitan untuk memahami kelas (Stevens, et al., 1974).

2.2.1 Coupling Between Object (CBO)

*Coupling between Objec*t (CBO) merupakan metrik perhitungan nilai kopling yang pertama kali diperkenalkan oleh Chidamber dan Kamerer (CK). Aloysius dan Arockiam (2012) berpendapat bahwa, metrik CBO hanya menghitung kopling luar dari kelas lain dengan cara menggabungkan dua kelas. Dan nilai yang diberikan untuk kopling ditetepkan dengan nilai 1. Sehingga mereka, mengusulkan penelitian dengan berbagai nilai-nilai kopling lain juga. Seperti yang diusulkan oleh Edward Berard (Edward, 1993 dalam Aloysius & Arockiam, 2012), ia mengusulkan berbagai macam kopling yang didefinisikan sebagai berikut:

a. *Control Coupling* (CC)

Melewati penanda kontrol antar modul, sehingga satu modul mengontrol pengurutan langkah-langkah dari modul lain. Control Coupling terjadi antar modul dimana ketika data yang lewat dapat mempengaruhi logika internal pada salah satu modul, misalnya penanda atau switch (Borysowich, 2007).

b. *Global Data Coupling* (GDC)

Dua atau lebih modul yang menggunakan struktur data global yang sama. Kondisi ini terjadi ketika pada objek/kelas parent terdapat suatu variabel global dimana variabel tersebut juga digunakan pada kelas atau objek yang lain.

c. *Internal Data Coupling* (IDC)

Suatu modul yang secara langsung memodifikasi data lokal dari modul yang lain. Pada sistem berorientasi objek, Internal Data Coupling dapat terjadi ketika suatu objek memanggil suatu *method* yang sama pada objek yang berbeda, dimana *method* yang dipanggil memiliki modifikasi yang berbeda terhadap variabel global yang sama di setiap objek.

d. *Data Coupling* (DC)

Output dari satu modul merupakan suatu inputan pada modul yang lainnya menggunakan daftar parameter untuk melewatkan *item* antar *routine.*

e. *Control Coupling* (CC)

Beberapa atau semua isi dari suatu modul yang termasuk dalam isi yang lainnya. Lexical Content Coupling pada sistem berorientasi objek dapat terjadi ketika dua atau lebih objek melakukan modifikasi yang berbeda pada variabel global yang sama.

2.2.1 Kalibrasi Penilaian Bobot

Dalam menentukan bobot pemahaman pada berbagai tipe kopling yang diusulkan oleh Edward Berard, Aloysius menyelenggarakan sebuah test pemahaman kepada 30 mahasiswa yang memiliki kemampuan cukup dalam menganalisa pemrograman berorientasi objek dan bahasa Java serta mendapatkan nilai 65% lebih pada setiap ujian semester. Test tersebut di tujukan untuk mencari waktu yang dibutuhkan dalam memahami kompleksitas dari program berorientasi objek dengan memperhatikan perbedaan macam kopling. Setiap kasus atau kategori disediakan dua program berorientasi objek yang berbeda, jadi total keseluruhan terdapat sepuluh macam kode program. Waktu yang diperoleh mahasiswa pada setiap kode program dirata-rata berdasarkan tiap-tiap kategori. Kategori waktu pamahaman dapat dilihat pada Tabel 2.1

Tabel 2.1 Kategori Waktu Pemahaman

Program	Rata-rata Waktu Pemahaman	Kategori	Rata-rata Waktu Pemahaman
1	40.7	LCC	40.18333
2	39.66667		
3	30.76667	DC	30.88333
4	31		
5	21.43333	IDC	22.21667
6	23		
7	10.8	GDC	11.13333
8	11.46667		

Sumber: (Aloysius & Arockiam, 2012)

2.2.3 *Cognitive Weighted Coupling Between Object* (CWCBO)

Metrik *Cognitive Weighted Coupling Between Object* (CWCBO) yang diusulkan oleh Aloysius merupakan perpaduan antara metrik yang diusulkan oleh Edward Berard dan penggunaan teori bobot berdasarkan waktu pemahaman pada setiap kategori dalam penelitian yang dilakukan oleh Aloysius. Metrik CWCBO dapat dihitung menggunakan persamaan sebagai berikut.

$$cwcbo = ((CC * WFCC) + (GDC * WFGDC) + (IDC * WFIDC) + (DC * WFDC) + (LCC * WFLCC))$$

Keterangan:
CC : Jumlah dari modul yang berisi *Control Coupling.*
GDC : Jumlah *Global Data Coupling*
IDC : Jumlah *Internal Data Coupling*
DC : Jumlah *Data Coupling*
LCC : Jumlah *Lexical Content Coupling*

Penilaian faktor bobot pada setiap jenis kopling didasarkan pada hasil kalibrasi yang dilakukan Aloysius, yaitu pengukuran rata-rata waktu pemahaman mahasiswa pada setiap kode program dengan kategori-kategori yang telah ditentukan. Pemberian nilai bobot pada setiap jenis kopling dapat dilihat pada Tabel 2.2.

Tabel 2.2 Faktor Bobot pada Setiap Kopling

No	Faktor Bobot	Bobot	Keterangan
1	WFCC	1	Faktor bobot *Control Coupling*
2	WFGDC	1	Faktor bobot *Global Data Coupling*
3	WFIDC	2	Faktor bobot *Internal Data Coupling*
4	WFDC	3	Faktor bobot *Data Coupling*
5	WFLCC	4	Faktor bobot *Lexical Data Coupling*

Sumber: (Aloysius & Arockiam, 2012)

2.3 Spoon

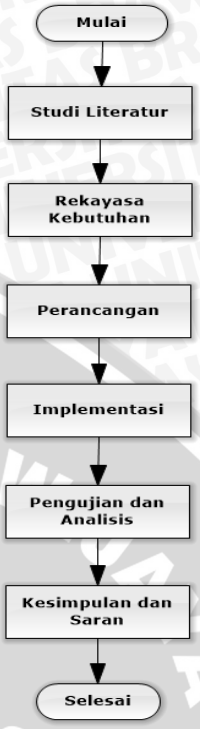
Spoon merupakan sebuah *library* yang dapat digunakan untuk menganalisa dan merubah kode program dengan bahasa Java. Spoon memungkinkan pengembang untuk menulis dengan cakupan besar dalam menganalisa *domain* yang spesifik dan merubahnya dengan mudah sesuai kebutuhan (Pawlak, et al., 2015).

2.4 JFreeChart

JFreeChart merupakan sebuah *library* gratis yang digunakan oleh para developer untuk menampilkan diagram pada aplikasinya. JFreeChart pertama kali dibuat oleh David Gilbert pada Februari 2000. Hingga saat ini JFreeChart menjadi *library* diagram untuk Java yang paling banyak digunakan oleh para developer. Hal itu dapat dilihat pada situs resmi JFreeChart, bahwa lebih dari 2,2 juta pengguna yang telah mengunduhnya (Gilbert, David, 2005).

3. METODOLOGI

Metodologi penelitian ini dilakukan dalam beberapa tahapan, yaitu Studi Literatur, Rekayasa Kebutuhan, Perancangan, Implementasi, Pengujian dan analisa, pengambilan kesimpulan dan saran serta penulisan laporan. Gambar 3.1 menunjukkan tahapan - tahapan dalam penelitian ini:



Gambar 3.1 Alur metodologi penelitian

4. REKAYASA KEBUTUHAN

Proses rekayasa kebutuhan pada sistem ini terdiri dari gambaran umum sistem, analisa kebutuhan fungsional dan analisa kebutuhan non fungsional.

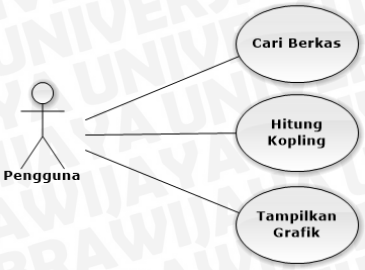
4.1 Gambaran Umum Sistem

Sistem yang dibangun dalam penelitian ini adalah sebuah perangkat lunak dengan judul "Kakas Bantu Perhitungan Nilai Kopling Menggunakan Metrik Cognitive *Weighted Coupling Between Object (CWCBO)"*. Tujuan utama dari sistem ini adalah untuk mengetahui masukan, keluaran dan efek berupa perhitungan kualitas sebuah *source code* berdasarkan nilai kopling antar objek. Dimana nilai kopling tersebut juga didasarkan pada metrik kopling dengan penilaian bobot berupa tingkat pemahaman terhadap berbagai jenis kopling. Pada tahap awal sistem mendapat masukan berupa sejumlah *source code* dari pengguna, kemudian sistem akan melakukan perhitungan kopling dengan metrik kopling CWCBO dan menghasilkan keluaran berupa nilai kopling. Selain itu sistem ini juga dapat menampilkan data berupa grafik perbandingan nilai kopling dari sejumlah kode program yang telah dihitung berdasarkan persamaan metrik CBO dan persamaan metrik CWCBO.

4.2 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional menghasilkan daftar kebutuhan fungsional sistem atau layanan-layanan yang dapat dilakukan oleh sistem. Daftar dari kebutuhan fungsional sistem ditunjukkan *use case diagram* pada

Gambar 4.1



Gambar 4.1 *Use Case Diagram*

4.3 Analisis Kebutuhan Non Fungsional

Kebutuhan non fungsional berisi daftar kebutuhan yang dapat dijadikan acuan dalam menentukan kualitas secara keseluruhan dari suatu sistem. Kebutuhan non fungsional menetukan batasan pada produk yang sedang dikembangkan serta dapat menentukan batasan-batasan eksternal yang harus dipenuhi oleh produk tersebut. Daftar kebutuhan non fungsioanl dapat dilihat pada Tabel 4.1

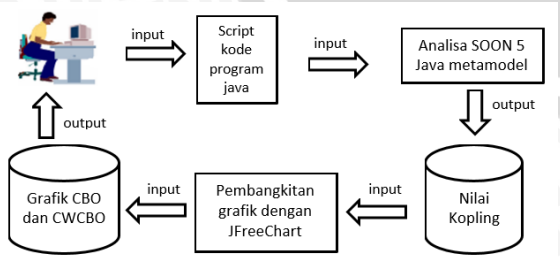
Tabel 4.1 Daftar Kebutuhan Non Fungsional

Kode Kebutuhan	Jenis Kebutuhan	Deskripsi
SRS_NF001	*Akurasi*	Sistem harus dapat melakukan perhitungan nilai kopling antar objek dalam source code kode program yang diuji sesuai dengan perhitungan manual minimal 90%.

5. PERANCANGAN DAN IMPLEMENTASI

5.1 Perancangan Arsitektur

Perancangan arsitektur digunakan untuk menggambarkan struktur keseluruhan sistem sehingga sistem dapat teroganisir (Sommerville, 2011). Perancangan arsitektur menggunakan notasi-notasi untuk menggambarkan langkah-langkah penelitan yang digunakan dalam pembangunan sistem ini. Arsitektur perangkat lunak dapat dilihat pada gambar 5.1 berikut:



Gambar 5.1 Arsitektur Sistem Perhitungan Nilai Kopling

5.2 Perancangan Alur Parse Source Code

Perancangan alur *parse source code* menjelaskan tentang alur bagaimana menganalisa sebuah inputan yang berupa *source code* program. Perancangan ini bertujuan untuk mendapatkan nilai dari lima tipe kopling yang akan menjadi parameter dalam perhitungan nilai kopling *CBO* dan *CWCBO*. Kelima kopling tersebut adalah *Control Coupling* (CC), *Global Data Coupling* (GDC), *Internal Data Coupling* (IDC), *Data Coupling* (DC) dan *Lexical Content Coupling* (LCC). Alur dalam mendapatkan nilai kelima kopling tersebut adalah sebagai berikut:

1. Mencari semua *class* pada file Java yang diinputkan.
2. Mencari *parentclass* beserta variabelnya.
3. Mencari *method* dari *class* yang merupakan class turunannya.
4. Menelusuri setiap *method-method* berdasarkan *statement-statement*. Jika terdapat variabel yang sama dengan variabel yang ada pada *parentclass* maka variabel tersebut merupakan GDC.
5. Mencari *statement* yang merupakan percabangan if else.
6. Menelusuri setiap parameter kondisi yang ada. Jika parameter kondisinya terdapat variabel yang sama dengan GDC yang dicari sebelumnya, maka variabel tersebut termasuk ke dalam DC.
7. Mencari *statement else* (tidak terdapat parameter kondisi) pada percabangan *if else*. Jika parameter kondisi pada *statement if else* sebelumnya terdapat DC yang dicari sebelumnya, maka *statement else* tersebut merupakan CC.
8. Menelusuri setiap *method-method* yang lain berdasarkan *statement-statement*. Jika terdapat variabel yang sama dengan GDC namun memiliki *class* yang berbeda, maka variabel tersebut merupakan LCC.
9. Mencari *statement* yang merupakan *statement* operasi. Jika *leftassigned* (variable yang berada di sebelah kiri simbol operator, seperti "+=") sama dengan GDC, maka variabel tersebut disimpan sementara pada suatu variabel penyimpanan, diasumsikan variabel penyimpanannya tempIDC.
10. Menelusuri setiap *method* berdasarkan *statement-statement* pada *class* yang lainnya. Mencari *statement* yang merupakan *statement* operasi. Jika *leftassigend* sama dengan tempIDC, variable tersebut disimpan sementara pada suatu variabel penyimpanan, diasumsikan variabel penyimpanannya dataIDC.
11. Menelusuri *parentclass*. Jika ada *method* pada *parentclass* yang berisi dataIDC, dimana dataIDC tersebut merupakan *leftassigned* dalam *statement* persamaan, maka method tersebut disimpan dalam variabel methodWithIDC.
12. Mencari *class* yang mengandung *main method*.
13. Menelusuri *main method*. Jika pada *main method* terdapat lebih dari satu *instance* objek dari *parentclass*, dan setiap objek memanggil methodWithIDC, maka variabel dalam dataIDC merupakan IDC.

5.3 Perancangan Algoritma

Perancangan algoritma menjelaskan secara detail algoritma method dari yang terdapat pada suatu *class* yang telah dimodelkan pada *class diagram.* Salah satu perancangan algoritma yang dibuat adalah algoritma *method* checkMethodParent yang ada dalam *class SPOON_metamodel.* Perancangan algoritma method checkMethodParent dapat dilihat pada Tabel 5.1 berikut ini.

Tabel 5.1 Algoritma *method checkMethodParent*

Algoritma *method checkMethodParent*
checkMethodParent(methodName, methodParent) { mengulang sejumlah methodParent variabel get = methodParent terpilih jika methodName = get return true return false

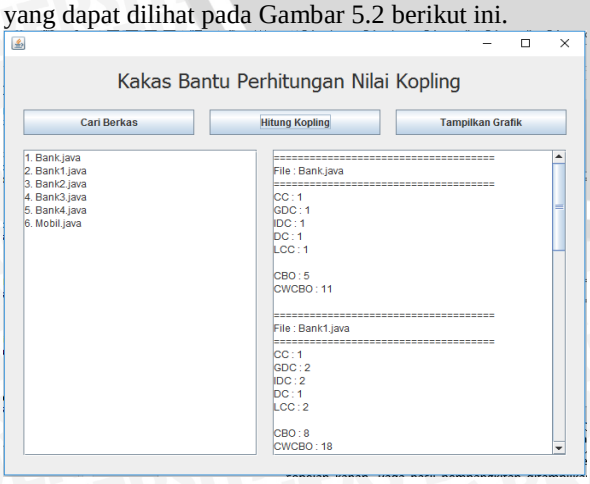
5.4 Batasan Implementasi

Batasan-batasan yang terdapat pada implementasi sistem kakas bantu perhitungan nilai kopling ini adalah sebagai berikut:

1. Data yang dimasukkan kedalam sistem untuk dibangkitkan kasus ujinya merupakan file berekstensi Java.
2. Dalam satu file yang diinputkan harus tediri lebih dari satu kelas.
3. Sistem hanya dapat melakukan perhitungan pada *source code* yang *object oriented* dan terdapat struktur hirarki.
4. Perhitungan nilai kopling didasarkan pada metrik CBO dan CWCBO, dimana pada masing metrik terdapat lima jenis tipe kopling, yaitu *Control Coupling, Global Data Coupling, Internal Data Coupling, Data Coupling* dan *Lexical Content Coupling.*

5.5 Implementasi Antarmuka

Implementasi antarmuka dilakukan berdasarkan pada hasil perancangan antarmuka yang telah dibuat sebelumnya. Salah satu antarmuka yang dihasilkan adalah antarmuka hasil perhitungan nilai kopling, yang dapat dilihat pada Gambar 5.2 berikut ini.



Gambar 5.2 Antarmuka hasil perhitungan nilai kopling

Daftar nama program yang dinputkan ditampilkan pada *text area* sebelah kiri. Sedangkan *text area* sebelah kanan untuk menampikan hasil perhitungan nilai kopling.

6. PENGUJIAN

Proses pengujian sistem dilakukan melalui empat tahapan yaitu, pengujian unit, pengujian integrasi, pengujian validasi dan pengujian akurasi.

6.1 Hasil dan Analisis Pengujian Validasi

Proses pengujian validasi dilakukan dengan melihat kesesuaian antara fungsi hasil kerja sistem dengan daftar kebutuhan sistem. Hasil pengujian validasi dapat dilihat pada Tabel 6.1 berikut ini.

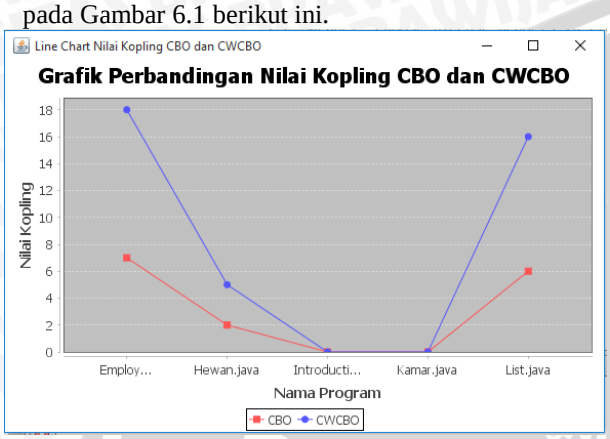
Tabel 6.1 Hasil Pengujian Fungsional

No	Kasus Uji	Hasil yang Didapatkan	Status
1.	Uji Cari Berkas	Sistem dapat menampilkan jendela bagi *user* untuk memasukkan file inputan. Sistem dapat menerima beberapa inputan yang berupa file berektensi java.	Valid
2.	Uji Cari Berkas Alternatif 1	Sistem dapat membatalkan menginputkan file ketika *user* menekan tombol *cancel*	Valid
3.	Uji Hitung Kopling	Sistem dapat menampilkan hasil perhitungan kopling dari beberapa inputan kode program	Valid
4.	Uji Hitung Kopling ALternatif 1	Sistem dapat menampilkan pesan “File kosong, masukkan file yang akan diuji!” ketika *user* menekan tombol Hitung Kopling tanpa memasukkan file inputan terlebih dahulu	Valid
5.	Uji Tampilkan Grafik	Sistem dapat menampilkan grafik perbandingan nilai kopling CBO dan CWCBO sesuai perhitungan kopling sebelumnya.	Valid
6.	Uji Tampilkan Grafik alternative 1	Sistem dapat menampilkan pesan “File kosong, masukkan file yang akan diuji!” ketika *user* menekan tombol Tampikan Grafik tanpa memasukkan file inputan atau melakukan perhitungan kopling terlebih dahulu.	Valid

Berdasarkan hasil pengujian fungsional pada Tabel 6.1, maka dapat disimpulkan bahwa sistem telah memenuhi semua kebutuhan fungsional.

6.2 Hasil dan Analisis Pengujian Akurasi

Pengujian akurasi merupakan proses pengujian yang membandingkan hasil perhitungan nilai kopling secara manual dan hasil perhitungan nilai kopling secara otomatis menggunakan sistem. Dalam proses pengujian akurasi menggunakan lima program dengan bahasa Java yang akan dijadikan sebagai inputan sistem. Program-program tersebut diambil dari sampel beberapa projek mata kuliah. Hasil perhitungan nilai kopling dengan sistem dapat dilihat pada Gambar 6.1 berikut ini.



Gambar 6.1 Grafik Perbandingan Nilai Kopling *CBO* dan *CWCBO*

Dari grafik perbandingan nilai kopling tersebut menunjukkan bahwa metrik kopling CWCBO mempunyai nilai yang relative lebih tinggi dibandingkan dengan metrik kopling CBO, hal ini didasari karena metrik CWCBO telah ditambahkan bobot *cognitive* (pemahaman) pada setiap jenis kopling yang menjadi parameternya. Setiap jenis kopling yang menjadi parameter metrik tersebut mempunyai bobot yang berbeda sesuai dengan tingkat kesulitan untuk dipahami. Sehingga 2 program yang berbeda dengan nilai CBO yang sama memungkinkan untuk menghasilkan nilai CWCBO yang berbeda, jika jumlah kopling pada masing-masing parameter berbeda.

Setelah dilakukan perhitungan kopling secara manual dan secara otomatis menggunakan system pada Gambar 6.1, maka dapat diperoleh hasil perbandingan nilainya yang dapat dilihat pada Tabel 6.2 sebagai berikut.

Tabel 6.2 Hasil Pengujian Akurasi

Nama Program	Secara Manual		Dengan Sistem		Hasil
	CBO	CW CBO	CBO	CW CBO	
Employee.java	7	18	7	18	Valid
Hewan.java	2	5	2	5	Valid
Introduction.java	0	0	0	0	Valid
Kamar.java	0	0	0	0	Valid
List.java	6	16	6	16	Valid

$$Akurasi = \frac{jumlah\ program}{jumlah\ valid} \times 100\%$$
$$= \frac{5}{5} \times 100\%$$
$$= 100\%$$

Dari kasus uji yang telah dilaksanakan sesuai dengan prosedur pengujian akurasi yaitu, membandingkan hasil perhitungan secara menual dan dengan sistem pada lima program diperoleh hasil akurasi sebesar 100%. Hal ini dapat disimpulkan bahwa pengujian akurasi yang dilakukan telah berhasil karena nilai akurasi lebih besar dari pada nilai batas minimalnya, yaitu 90%.

6.3 Pembahasan Hasil

Berdasarkan hasil perhitungan nilai kopling yang telah dilakukan pada lima program, seperti yang tertera pada Gambar 6.1, diketahui bahwa dua program menghasilkan nilai kopling CBO dan CWCBO = 0. Kedua program tersebut adalah Introduction.java dan Kamar.java. Hal tersebut dapat dikatakan bahwa kedua program dianggap mempunyai desain yang baik karena memiliki kopling rendah, sehingga mudah untuk dipahami. Sedangkan program dengan nama Employee.java memiliki nilai kopling tertinggi, yaitu CBO = 8 dan CWCBO = 18. Maka program Employee.java dapat dianggap mempunyai desain perangkat lunak yang buruk, sehingga akan lebih sulit untuk dipahami dibandingkan program Introduction.java dan Kamar.java.

Nilai kopling CWCBO relatif lebih tinggi dibandingkan dengan nilai kopling CBO. Hal ini dikarenakan pada perhitungan kopling CWCBO ditambahkan bobot pemahaman pada setiap kategorinya. Nilai bobot yang diberikan pada setiap kategori memiliki nilai yang berbeda sesuai dengan tingkat kesulitan ketegori untuk dipahami. Nilai bobot paling kecil adalah 1 dan paling besar adalah 4. Suatu program memungkinkan mempunyai nilai kopling CBO dan CWCBO yang sama, jika program tersebut hanya mengandung jenis-jenis kopling dengan bobot pemahaman yang paling rendah, yaitu Control Coupling dan Global Data Coupling. Kedunya memiliki bobot pemahaman dengan nilai 1. Namun suatu program akan memiliki nilai kopling CBO dan CWCBO dengan selisih yang semakin jauh jika program tersebut banyak mengandung kategori kopling yang memiliki bobot pemahaman tinggi, seperti Lexical Content Coupling dengan bobot pemahaman = 4.

7. KESIMPULAN DAN SARAN

7.1 Kesimpulan

Dalam penelitian pembangunan kakas bantu perhitungan nilai kopling menggunakan metrik *Cognitive Weighted Coupling Between Object* (CWCBO) yang dilakukan oleh peneliti, terdapat beberapa hal yang dapat disimpulkan yaitu:

1. Kakas bantu perhitungan nilai kopling ini mempunyai 3 fitur yaitu, cari berkas, hitung kopling dan tampilkan grafik. Sistem ini dibangun dengan menggunakan 4 *class*, yaitu *Controller*, *Main_UI*, *ResultCoupling* dan *SPOON_metamodel*. Sistem ini diimplementasikan menggunakan bahasa pemrograman utama Java, *library* Spoon dan *library* JFreeChart. Cara kerja sistem, pertama sistem akan menerima inputan berupa *source code* program berbahasa Java, kemudian sistem akan menganalisa *source code* dengan *library* Spoon untuk mendapatkan jumlah dari masing-masing tipe kopling yang menjadi parameter dari metrik CWCBO. Setelah itu sistem akan menampilkan grafik perbandingan nilai kopling CBO dan CWCBO dengan pemanfaatan *library* JFreeChart.
2. Pengujian akurasi pada sistem ini dilakukan dengan menguji tingkat akurasi antara perhitungan yang dilakukan secara manual dan perhitungan yang dilakukan dengan sistem. Dari pengujian pada lima program yang menjadi data ujinya menghasilkan nilai kopling CBO dan CWCBO sama semua, sehingga nilai akurasinya adalah 100%. Dengan rincian dua pogram memiliki nilai kopling 0 dan nilai kopling tertinggi adalah 18.

7.2 Saran

Kakasa bantu perhitungan nilai kopling ini masih belum sempurna, sehingga masih perlu dilakukan pengembangan, perbaikan dan penyempurnaan. Beberapa hal yang perlu dikembangkan yaitu:

1. Pada sistem kakas bantu ini perlu ditambahkan *threshold* atau batasan nilai untuk menentukan tingkat kualitas suatu nilai kopling antar objek. Sehingga dengan adanya *threshold* dapat menentukan apakah perangkat lunak dengan nilai kopling tertentu sudah baik atau tidak.
2. Pada kakas bantu ini juga perlu adanya pengembangan agar sistem dapat menghitung nilai kopling tidak hanya dari program yang memiliki struktur hirarki, melainkan segala jenis program berorientasi objek.

DAFTAR PUSTAKA

Aloysius, A., & Arockiam, L. (2012). Coupling Complexity Metric: A Cognitive Approach. I.J. Information Technology and Computer Science, 29-35.

Borysowich, C. (2007). Design Principles: Coupling (Data and otherwise). Dipetik 3 22, 2016, dari http://it.toolbox.com/blogs/enterprise-solutions/design-principles-coupling-data-and-otherwise-16061

Coad, P., & Yourdon, E. (1991). Object Oriented Analysis. Prentice-Hall, New Jersey, Edisi ke-2.

Edward, B. V. (1993). Essays on object-oriented (Vol. 1 ed.). Prentice-Hall: Berard Software.

Gilbert, David;. (2005). JFreeChart. Dipetik Juli 22, 2016, dari http://www.jfree.org/jfreechart/

Lestari, E. (2015). PERHITUNGAN KUALITAS DESAIN OBJECT-ORIENTED MENGGUNAKAN MATRIX SIMILARITY-BASED CLASS COHESION (SCC) PADA KELAS KOHESI BERBASIS WEB. Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya.

Misra, S., & Akman, K. I. (2008). Weighted Class Complexity: A Measure of Complexity for Object Oriented System. Department of Computer Engineering, Atilim University, Ankara, Turkey, 1689-1708.

Pawlak, R., Monperrus, M., Petitprez, N., Noguera, C., & Seinturier, L. (2015). Spoon: A Library for Implementing Analyses and Transformations of Java Source Code. Software: Practice and Experience, Wiley, 33.

Sommerville, I. (2011). Software Engineering. Ninth Edition. Boston: Pearson Education, Inc.,.

Stevens, W., Myers, G., & Constantine, L. (1974). Structured design. IBM Systems Journal, , 115-139.

Yadav, A., & Khan, R. A. (2009). Measuring Design Complexity – An Inherited Method Perspective. SIGSOFT Software Engineering Notes, 24 No.4,pp:, 1-5.

Yadav, A., & Khan, R. A. (2009, Maret 6-7). Complexity:A Reliability Factor. IEEE International Advance Computing Conference (IACC-2009), Patiala, India, hal. 2375-2378.