

## BAB III

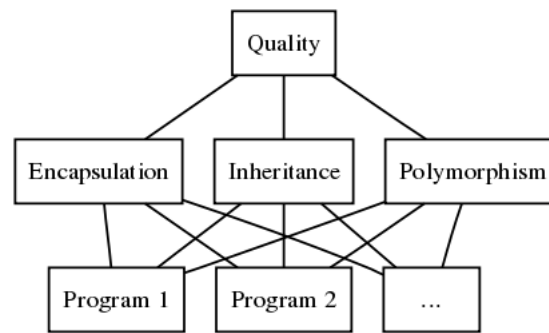
# METODOLOGI PENELITIAN

Salah satu tujuan dari pengukuran kualitas desain adalah untuk melihat apakah ada aspek desain yang masih dapat diperbaiki untuk meningkatkan kualitas desain. Setelah dilakukannya perubahan terhadap desain program, perlu dilakukan penilaian lebih lanjut untuk melihat apakah perubahan yang dilakukan tersebut meningkatkan kualitas desain atau malah menurunkan kualitas, dan juga perlu diketahui seberapa besar peningkatan kualitas yang didapatkan. Bab ini akan membahas mengenai proses penentuan peringkat kualitas desain dengan memanfaatkan *metric* MOOD dan AHP.

### 3.1 HIRARKI KEPUTUSAN AHP

MOOD menyediakan *metric* untuk melakukan pengukuran terhadap aspek desain sistem. Supaya hasil pengukuran MOOD ini dapat berguna, diperlukan suatu proses untuk menentukan apakah sistem yang diukur telah memenuhi suatu standar kualitas desain tertentu. [ABR96] memanfaatkan teknik perhitungan statistik (*Confidence Interval* dan *Standard Scores*) untuk menentukan rentang batasan nilai *metric* yang dianggap berkualitas baik dan kurang baik. [NUR07] mengimplementasi MOOD dan proses penentuan peringkat dengan perhitungan statistik ini untuk mengukur kualitas program Java.

Perhitungan statistik yang digunakan oleh [ABR96] dan [NUR07] mengasumsikan prioritas atau bobot nilai yang sama dari seluruh *metric* yang dikumpulkan datanya. Dari kumpulan *metric* MOOD yang digunakan (MHF, AHF, MIF, AIF, POF, COF), dapat dilihat bahwa hasil pengukuran dari suatu *metric* memiliki pengaruh terhadap *metric* yang lainnya. Hal ini diterjemahkan menjadi beberapa *metric* harusnya memiliki prioritas penilaian yang lebih tinggi dibandingkan dengan *metric* lainnya. Misalnya, MHF yang mengukur seberapa besar *encapsulation* operasi dari suatu kelas akan berdampak pada MIF, POF dan COF. Semakin tinggi MHF akan menyebabkan nilai MIF, POF, COF semakin rendah dikarenakan jumlah *method* yang diwariskan juga akan semakin sedikit, dan menyebabkan berkurangnya kemungkinan untuk melakukan *overriding* (untuk mendapatkan kemampuan *polymorphism* suatu kelas) dan semakin sedikit jumlah ketergantungan antar satu kelas dengan *interface* kelas lainnya (karena jumlah *interface*-nya juga semakin sedikit).

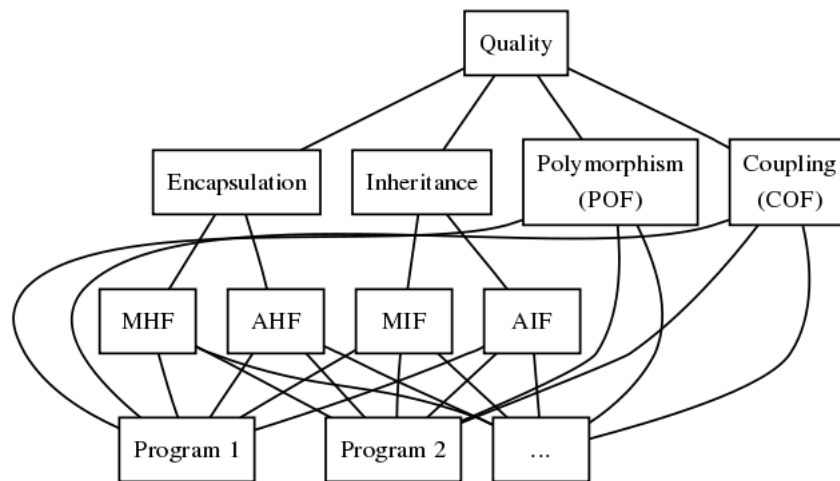


Gambar 3.1: Sistem AHP untuk *metric* berorientasi objek

AHP merupakan suatu sistem penilaian yang mengakomodasi bobot nilai yang berbeda dari masing-masing kriteria penilaian. Pada sistem penentuan peringkat kualitas desain dengan menggunakan AHP, masing-masing *metric* dapat dijadikan sebagai kriteria penilaian. Dan masing-masing program yang diukur dijadikan sebagai alternatif pilihan. Misalkan, kualitas desain sistem berorientasi objek ditentukan oleh seberapa banyak implementasi ketiga konsep objek *encapsulation*, *inheritance* dan *polymorphism* ke dalam sistem. Masing-masing konsep objek yang dinilai ini dijadikan sebagai kriteria penilaian di dalam AHP (Gambar 3.1).

Konfigurasi kriteria penilaian pada Gambar 3.1, memperlihatkan tiga buah kriteria penilaian yaitu *encapsulation*, *inheritance* dan *polymorphism*. Masing-masing kriteria akan diberikan bobot prioritas relatif terhadap kriteria lainnya (*encapsulation* dibandingkan dengan *inheritance* dan *polymorphism* dan *inheritance* dibandingkan dengan *polymorphism*). Dan masing-masing alternatif pilihan (Program 1, Program 2, dst) dapat diberikan nilai sesuai dengan kriteria yang telah ditentukan pada level di atasnya.

Fleksibilitas AHP memungkinkan penilai untuk mengubah konfigurasi kriteria penilaian sesuai dengan kebutuhan penilaian. Misalnya, hirarki kriteria AHP pada Gambar 3.1 dapat dibuat lebih detail lagi dengan menambahkan sub kriteria ke masing-masing kriteria yang telah ada. Dengan demikian, tidak tertutup kemungkinan untuk menambahkan kriteria baru ataupun sub kriteria baru untuk meningkatkan ketajaman penilaian kualitas desain. Misalnya dari kriteria yang ada pada Gambar 3.1, ditambahkan lagi sub kriteria berupa MHF dan AHF (sebagai sub kriteria *encapsulation*), AIF dan MIF (sebagai sub kriteria *inheritance*), POF untuk mengukur kriteria *polymorphism*



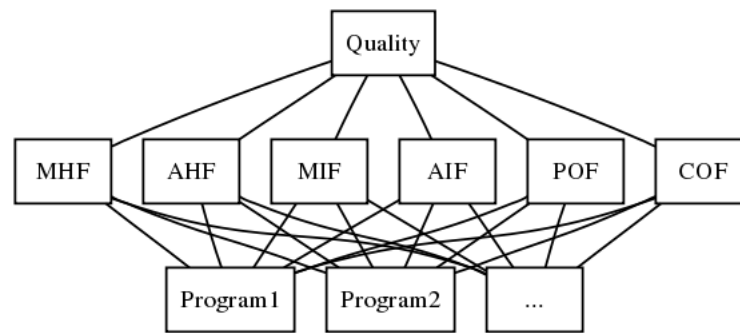
Gambar 3.2: Sistem AHP dengan sub kriteria

dan kriteria baru yaitu *Coupling* (diukur dengan COF). Bentuk sistem AHP yang baru ini dapat dilihat pada Gambar 3.2.

Konsekuensi meletakkan MHF, AHF, MIF dan AIF sebagai sub kriteria dari *encapsulation* dan *inheritance* adalah sub kriteria tersebut tidak dapat lagi diberikan nilai prioritas yang relatif terhadap sub kriteria dari kriteria yang berbeda (misalnya antara MHF dengan MIF) dan juga dengan kriteria yang berbeda (antara MHF dengan POF). Masing-masing sub kriteria hanya dapat memiliki bobot prioritas yang relatif terhadap sub kriteria yang berada pada tingkatan yang sama dan yang berasal dari kriteria yang sama juga. Apabila diinginkan penentuan prioritas yang lebih fleksibel, masing-masing sub kriteria dapat di-*upgrade* menjadi kriteria level pertama seperti pada Gambar 3.3. Dengan meletakkan masing-masing *metric* sebagai kriteria pada level yang sama, penilai dapat memberikan bobot prioritas yang relatif antara satu *metric* dengan *metric* lainnya.

### 3.2 GOOD MOOD DAN BAD MOOD

Proses pengartian hasil pengukuran *metric* merupakan proses yang penting dalam menentukan peringkat kualitas desain. [ABR96] dan [NUR07] memanfaatkan perhi-



Gambar 3.3: Struktur hirarki keputusan MOOD pada sistem AHP

tungan statistik dalam menentukan rentang nilai MOOD yang dianggap berkualitas bagus dengan MOOD yang dianggap kurang berkualitas. Rentang nilai ini pada [NUR07] disebut dengan *Confidence Interval Good Object Oriented Design* (CIGOOD). Program yang memiliki nilai MOOD yang berada di dalam rentang CIGOOD dikelompokkan ke dalam kelompok program yang memiliki kualitas desain yang baik, dan program yang berada di luar CIGOOD dikelompokkan pada kelompok program yang kualitas desainnya kurang memadai.

Program yang memiliki nilai *metric* di dalam rentang CIGOOD akan diposisikan pada peringkat kualitas yang lebih tinggi dibandingkan dengan program yang nilai *metric*-nya berada diluar rentang CIGOOD. Kemudian, masing-masing program akan diurutkan lagi berdasarkan nilai *metric*-nya. Untuk program yang nilai *metric*-nya berada di dalam rentang CIGOOD, penentuan peringkat ditentukan lagi berdasarkan *standard scores*. Dan untuk program yang berada diluar rentang nilai CIGOOD, penentuan peringkat dilakukan dengan memanfaatkan perhitungan derajat error.

Pada AHP, masing-masing program akan memiliki bobot penilaian tergantung dari hasil pengukuran dan bobot dari masing-masing *metric*. Untuk memisahkan antara program yang memenuhi suatu standar kualitas tertentu, hasil pengukuran *metric* akan diproses lebih lanjut. Penilai menentukan sendiri rentang nilai dari masing-masing *metric* yang dianggap memenuhi kualitas desain yang baik. *Metric* yang hasil pengukurannya berada di dalam rentang nilai yang telah ditentukan dianggap memiliki kualitas desain yang baik (yang disebut dengan Good MOOD). Dan sebaliknya apabila hasil pengukuran *metric* berada diluar rentang yang telah ditentukan, maka program yang diukur dianggap memiliki kualitas yang kurang baik pada aspek yang diukur (yang disebut dengan Bad MOOD).

Masing-masing nilai *metric* dari program yang telah diukur akan diperiksa apakah berada pada rentang nilai kualitas desain yang telah ditentukan oleh penilai. Apabila hasil pengukuran *metric* berada pada rentang nilai yang ditentukan, maka nilai *metric* akan ditambahkan sebanyak 1. Jika nilai *metric* berada diluar rentang nilai yang ditentukan, nilai *metric* ditambahkan dengan 0.

$$good\_mood(score, lower, upper) = \begin{cases} 1 & \text{if } lower \leq score \leq upper \\ 0 & \text{otherwise} \end{cases}$$

Nilai 0 dan 1 dari perhitungan fungsi *good\_mood* berdasarkan dari nilai *metric* yang dihasilkan. Tujuan dari penggunaan fungsi *good\_mood* ini adalah untuk memberikan nilai tambah bagi program yang termasuk pada Good MOOD dan penalti kepada program yang termasuk pada Bad MOOD. Kemudian, hasil penjumlahan dari *metric* dan fungsi *good\_mood* dibagi dengan total nilai tertinggi yang dimungkinkan (dalam contoh ini 2) untuk mendapatkan nilai *metric* yang akan dimasukkan ke dalam sistem penilaian AHP.

$$\begin{aligned} MHF &= \frac{MHF + good\_mood(MHF, lower, upper)}{2} & AIF &= \frac{AIF + good\_mood(AIF, lower, upper)}{2} \\ AHF &= \frac{AHF + good\_mood(AHF, lower, upper)}{2} & POF &= \frac{POF + good\_mood(POF, lower, upper)}{2} \\ MIF &= \frac{MIF + good\_mood(MIF, lower, upper)}{2} & COF &= \frac{COF + good\_mood(COF, lower, upper)}{2} \end{aligned}$$

Berikut ini contoh perhitungan MOOD yang telah diproses dengan beban kualitas. Contoh disini mengambil hasil pengukuran MOOD dari sampel program Enterprise Resource Planning Adempiere. Rentang nilai Good MOOD yang digunakan pada contoh ini dapat dilihat pada Tabel 3.1. Dan hasil pengukuran MOOD sebelum diproses terlihat pada Tabel 3.2. Proses penyaringan hasil pengukuran *metric* MOOD dapat dilihat pada Gambar 3.4.

Program yang mengimplementasikan konsep objek dengan baik adalah program yang memiliki *interface* yang cukup, tidak terlalu banyak dan juga tidak terlalu sedi-

Tabel 3.1: Rentang nilai Good MOOD

<i>Metric</i>	<i>Lower Limit</i>	<i>Upper Limit</i>
MHF	0.080	0.250
AHF	0.800	1.000
MIF	0.200	0.800
AIF	0.000	0.480
POF	0.000	0.100
COF	0.000	0.120

Tabel 3.2: Pengukuran MOOD ERP Adempiere

	MHF	AHF	MIF	AIF	POF	COF
Adempiere	0.146	0.371	0.884	0.721	0.017	0.001

kit (MHF). Meng-*encapsulate* atribut/data objek sehingga aksesnya dapat dikendalikan (AHF). Me-*reuse* kode program yang sudah diimplementasikan sebelumnya, tetapi tetap menjaga supaya hirarki pewarisan tidak terlalu dalam (MIF dan AIF). Meningkatkan fleksibilitas program melalui *polymorphism* tanpa mengorbankan kejelasan proses eksekusi program (POF) dan mengurangi tingkat dependensi antar objek/kelas (COF).

### 3.3 KONTRUKSI TOOL PENGUKURAN DAN PENENTUAN PERINGKAT DESAIN

Proses pengukuran kualitas desain program menghabiskan banyak waktu dan usaha yang cukup besar. Oleh karena itu, proses pengukuran desain biasanya dilakukan dengan memanfaatkan bantuan *tool* pengukuran. Pada sub bab ini, akan dibahas mengenai konstruksi *tool* yang digunakan untuk mengukur dan menentukan peringkat desain program dengan menggunakan MOOD sebagai *metric* pengukuran dan AHP sebagai *tool* untuk menentukan peringkat kualitas program.

*Tool* pengukuran dan penentuan kualitas desain yang dikembangkan pada tesis ini mengukur program yang ditulis dengan menggunakan bahasa pemrograman Java. Input yang dimasukkan dalam pengukuran adalah *byte code* program Java yang telah dipaketkan dalam bentuk paket JAR. *Tool* ini diimplementasikan dengan menggunakan bahasa Java. *Development Kit* yang digunakan adalah *Java Development Kit* versi 1.6.0.

$$\begin{aligned}
 MHF &= \frac{0.146 + good\_mood(0.146, 0.08, 0.25)}{2} = \frac{0.146 + 1}{2} = 0.573 \\
 AHF &= \frac{0.371 + good\_mood(0.371, 0.8, 1.0)}{2} = \frac{0.371 + 0}{2} = 0.185 \\
 MIF &= \frac{0.884 + good\_mood(0.884, 0.2, 0.8)}{2} = \frac{0.884 + 0}{2} = 0.442 \\
 AIF &= \frac{0.721 + good\_mood(0.721, 0.0, 0.48)}{2} = \frac{0.721 + 0}{2} = 0.360 \\
 POF &= \frac{0.017 + good\_mood(0.017, 0.0, 0.1)}{2} = \frac{0.017 + 1}{2} = 0.509 \\
 COF &= \frac{0.001 + good\_mood(0.001, 0.0, 0.12)}{2} = \frac{0.001 + 1}{2} = 0.500
 \end{aligned}$$

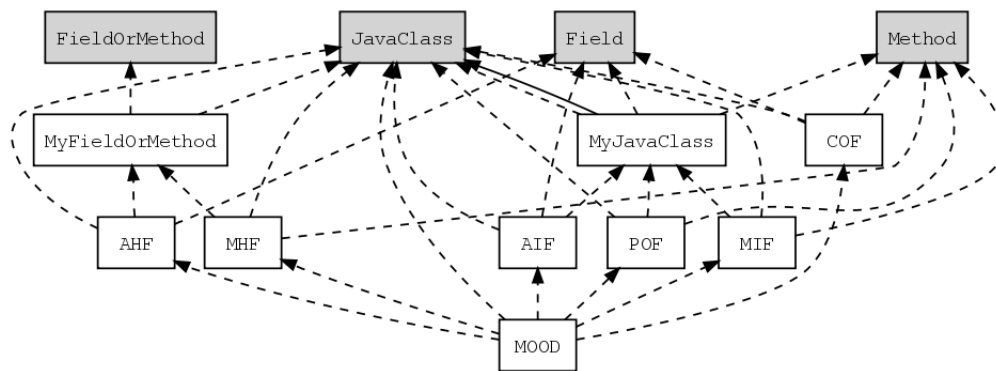
Gambar 3.4: Hasil pengukuran metric MOOD

Dalam pengembangan, *tool* ini dipecah ke dalam beberapa paket yaitu paket “jmood” sebagai implementasi *metric* MOOD, “jahp” sebagai implementasi proses pengambilan keputusan AHP. Aplikasi yang dikembangkan memiliki antar muka yang berjalan pada lingkungan grafik. Antar muka berbasis grafik ini diimplementasikan pada paket “moag”. Paket “jmood” dibangun dengan memanfaatkan paket *Byte Code Engineering Library* (BCEL), sebuah produk dari proyek *open source* Jakarta Apache Project. Dan paket “jahp” dibangun dengan memanfaatkan paket *Jama*, sebuah fungsi pustaka untuk operasi aljabar linear untuk Java.

#### **Paket jmood**

Paket jmood yang dikembangkan pada tesis ini merupakan kumpulan dari beberapa kelas yang merupakan implementasi dari *metric* MOOD. Kelas yang mengimplementasikan *metric* MOOD adalah kelas MHF, AHF, MIF, AIF, POF dan COF. Kelas-kelas pada paket ini secara ekstensif menggunakan paket BCEL dalam menganalisa *byte code* program Java yang akan diukur. Gambar 3.5 menampilkan hubungan antar kelas yang membentuk jmood.

Kelas MOOD merupakan kelas utama dari paket jmood ini. Input yang diharapkan adalah nama *file* paket jar sistem yang hendak diukur menggunakan MOOD. Kelas



Gambar 3.5: *Class Diagram* jmood

yang berada pada gambar kotak berwarna abu-abu pada Gambar 3.5 merupakan kelas dari paket BCEL. Garis panah putus-putus menggambarkan hubungan “*has-a*” antara kelas (Contoh MOOD “*has-a*” AIF). Sedangkan panah dengan garis solid menggambarkan hubungan “*kind-of*” antara kelas (Contoh MyJavaClass “*kind-of*” JavaClass).

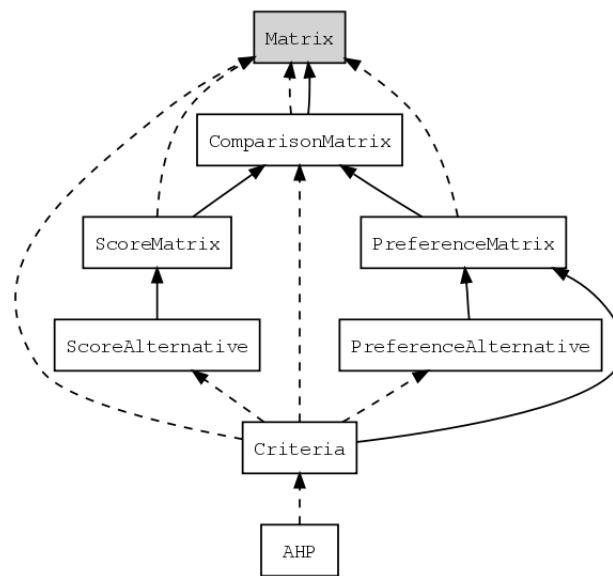
#### **Paket jahp**

Paket jahp merupakan implementasi proses pengambilan keputusan AHP. Proses pada AHP banyak melibatkan operasi perhitungan matriks dan aljabar linear, terutama untuk perhitungan *eigenvector*. Untuk melaksanakan tugasnya, paket jahp memanfaatkan paket *Jama*. Gambar 3.6 menampilkan hubungan antar kelas yang membentuk pake jahp.

#### **Paket moag**

Paket moag merupakan antar muka antara pengguna dengan sistem penilaian dan menentukan peringkat kualitas desain. Paket ini berisikan aplikasi yang merupakan tampilan utama program penentuan peringkat kualitas yang berjalan pada lingkungan grafik. Gambar 3.7 menampilkan hubungan antar kelas dari pake moag.





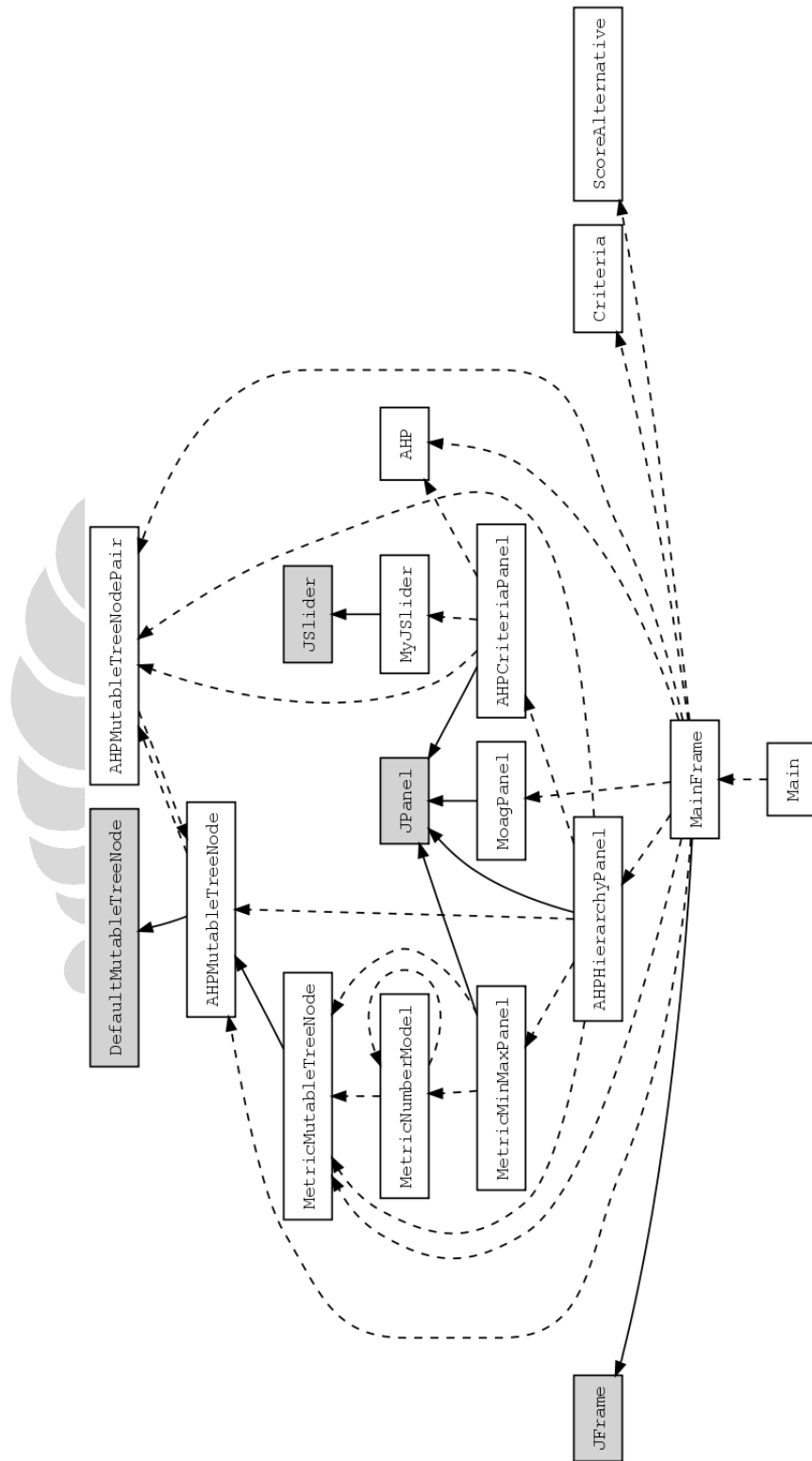
Gambar 3.6: *Class Diagram* jahp

### 3.4 PENGGUNAAN TOOL PENGUKURAN DAN PENENTUAN PERINGKAT DESAIN

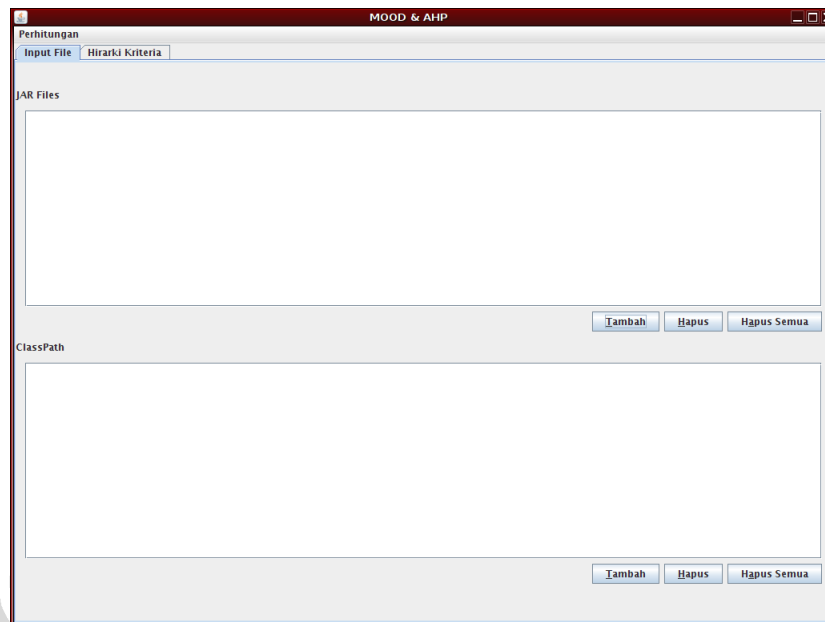
Aplikasi *tool* pengukuran dan penentuan peringkat desain yang dikembangkan pada tesis ini berjalan pada lingkungan grafik. Persyaratan yang harus dipenuhi untuk menjalankan aplikasi ini adalah tersedianya *java virtual machine* dengan versi 1.6 (JRE 1.6) untuk mengeksekusi program. Gambar 3.8 menampilkan tampilan program ketika pertama kali dijalankan.

Ada 4 tahapan yang harus dilalui oleh pengguna untuk menggunakan aplikasi ini. Pada langkah pertama, pengguna harus memasukkan *file* program Java yang akan diukur beserta dengan paket yang menjadi pelengkap dari *file* program Java yang akan diukur. Input *file* JAR (paket program yang akan diukur) dan ClassPath (paket pelengkap yang merupakan dependensi dari program yang diukur) dimasukkan oleh pengguna melalui tampilan seperti pada Gambar 3.8.

Langkah kedua adalah menentukan hirarki keputusan dalam pengukuran kualitas desain. Dalam menentukan hirarki keputusan, pengguna dapat menambahkan komponen-komponen yang menjadi penilaian dalam pengukuran kualitas desain. Setiap komponen penilaian dapat dipecah menjadi beberapa sub komponen penilaian. Persyaratan yang harus dipenuhi pada saat penentuan komponen penilaian ini adalah seluruh kom-



Gambar 3.7: Class Diagram moag



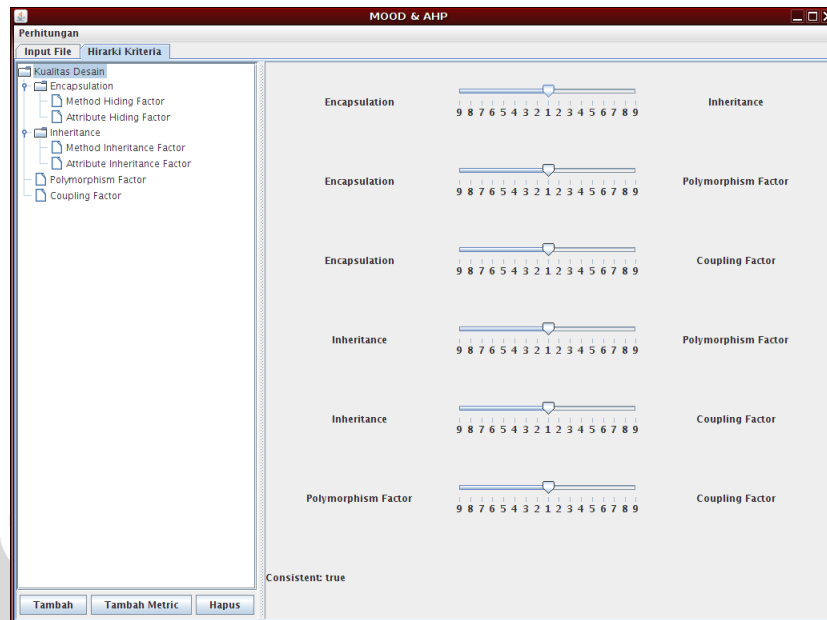
Gambar 3.8: Tampilan awal *tool* pengukuran dan penentuan peringkat desain

ponen penilaian yang tidak memiliki sub komponen penilaian harus berupa *Metric* Pengukuran Desain.

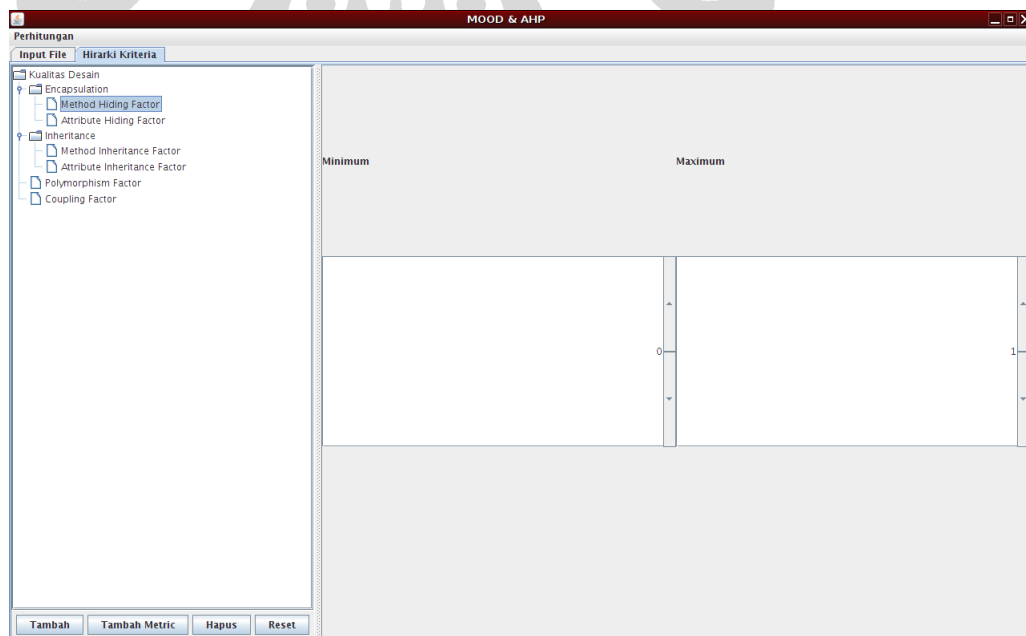
Aplikasi yang dikembangkan pada tesis ini mengikut sertakan sejumlah *metric* pengukuran yang dapat digunakan dalam penentuan komponen penilaian. *Metric* pengukuran yang lainnya dapat dikembangkan dan ditambahkan ke dalam aplikasi ini. Gambar 3.9 menampilkan tampilan pada saat penentuan komponen penilaian.

Penentuan perbandingan prioritas antara komponen penilaian juga ditentukan pada lokasi yang sama seperti yang terlihat pada Gambar 3.9. Apabila pengguna memilih komponen penilaian yang memiliki sub komponen penilaian, pada panel disebelah kanan akan ditampilkan *slider* untuk menentukan prioritas sub komponen penilaian dibandingkan dengan sub komponen penilaian lainnya. Besar nilai bobot prioritas penilaian berkisar antara 1 sampai dengan 9, dimana nilai 1 menunjukkan bahwa komponen penilaian memiliki bobot yang sama besar dengan komponen penilaian lainnya.

Gambar 3.10 menampilkan pilihan konfigurasi untuk mengubah rentang nilai yang termasuk pada Good MOOD. Untuk melakukan konfigurasi terhadap rentang nilai Good MOOD pada masing-masing *Metric* Pengukuran, pengguna harus memiliki komponen penilaian yang merupakan *Metric* Pengukuran pada hirarki penilaian. Rentang nilai yang dapat diisikan pada konfigurasi Good MOOD ini adalah berkisar antara 0.0 sampai dengan 1.0. Dengan ketentuan nilai minimum  $\leq$  maximum.



Gambar 3.9: Tampilan Penentuan Hirarki Kriteria



Gambar 3.10: Tampilan penentuan rentang nilai Good MOOD

The screenshot shows the output of the MOOD & AHP tool. It displays a comparison matrix, a criteria priority vector, MOOD scores for a specific program, and AHP weighted scores.

Criteria Comparison Matrix						
	Method Hiding Factor	Attribute Hiding Factor	Method Inheritance Factor	Attribute Inheritance Factor	Polywrphism Factor	
Method Hiding Factor	1,000	8,000	4,000	6,000	9,000	
Attribute Hiding Factor	0,125	1,000	6,000	3,000	5,000	
Method Inheritance Factor	0,250	0,167	1,000	1,000	1,000	
Attribute Inheritance Factor	0,157	0,333	1,000	1,000	2,000	
Polywrphism Factor	0,111	0,200	1,000	0,500	1,000	
Coupling Factor	0,111	0,111	0,500	0,250	0,500	
Consistent:	true					

Criteria Priority Vector	
Method Hiding Factor	0,564
Attribute Hiding Factor	0,215
Method Inheritance Factor	0,065
Attribute Inheritance Factor	0,079
Polywrphism Factor	0,048
Coupling Factor	0,029

MOOD Scores					
	Method Hiding Factor	Attribute Hiding Factor	Method Inheritance Factor	Attribute Inheritance Factor	Polywrphism Factor
/home/david/Adenpiere/11b/Adenpiere.jar	0,573	0,185	0,442	0,360	0,5

AHP Weighted Scores	
/home/david/Adenpiere/11b/Adenpiere.jar	1,000

Gambar 3.11: Tampilan Output *tool* pengukuran dan penentuan kualitas desain

AHP memberikan kebebasan untuk menentukan bobot komponen penilaian yang digunakan secara relatif dibandingkan dengan komponen penilaian lainnya. Untuk menjaga konsistensi pengaturan yang dilakukan oleh pengguna, pada tampilan konfigurasi juga ditampilkan apakah konfigurasi yang ditentukan konsisten atau tidak. Apabila konfigurasi bobot komponen penilaian tidak konsisten (Consistency: false), pengguna tidak dapat melanjutkan ke proses pengukuran.

Setelah menetapkan seluruh konfigurasi, pengguna dapat melakukan pengukuran terhadap kualitas desain dari program yang di-input berdasarkan pada hirarki penilaian yang telah ditentukan. Perintah pengukuran ini diakses melalui menu Pengukuran ▸ Ukur Kualitas Desain. Setelah selesai melakukan proses pengukuran, akan ditampilkan sebuah tab baru yang menampakan hasil output dari pengukuran yang telah dilakukan, tampilan output terlihat seperti pada Gambar 3.11.