PENERAPAN *DOMINANCE-BASED ROUGH SET APPROACH* DALAM PEMERINGKATAN KUALITAS DESAIN *SOFTWARE* BERORIENTASI OBJEK

Ardhi Tomiarfi[1] dan Petrus Mursanto[2]

[1]Magister Teknologi Informasi, Universitas Indonesia, Jl. Salemba Raya 4, Jakarta, 12000, Indonesia
[2]Enterprise Computing Lab – Fakultas Ilmu Komputer, Universitas Indonesia, Kampus Baru UI Depok, Jawa Barat, 16424 Indonesia

E-mail: tomiarfi@cs.ui.ac.id

Abstrak

Penerapan *Dominance-based Rough Set Approach* (DRSA) untuk menentukan peringkat kualitas sejumlah aplikasi berbasis objek telah dipelajari. DRSA diterapkan untuk memformulasikan satu set klasifikasi dimana kualitas *software* aplikasi digolongkan. Klasifikasi dibentuk berdasarkan hasil pengukuran *Metrics of Object Oriented Design* (MOOD) terhadap properti *software* berbasis Java. Pemanfaatan DRSA merupakan metode alternatif untuk menginterpretasikan nilai-nilai *metrics* dan menentukan sebuah nilai kuantitatif yang merepresentasikan kualitasnya relatif terhadap *software* yang lain. Studi eksperimental menunjukkan bahwa akurasi hasil DRSA bergantung pada jumlah *sample* aplikasi yang dijadikan referensi dalam menurunkan satu set definisi aturan untuk mengklasifikasikan properti *software*. *Sample* aplikasi dalam eksperimen diambil dari *library* Java yang telah teruji *modularity*, *usability* dan *maintainability*-nya. Secara umum, hasil eksperimen menyimpulkan bahwa DRSA dapat diterapkan untuk menentukan peringkat kualitas relatif dari sejumlah aplikasi dalam populasi *sample*.

Kata Kunci: *metric for object oriented software, analytic hierarchy process, DRSA*

Abstract

Application of Dominance-based Rough Set Approach (DRSA) to rank the quality of a number of object-based applications have been studied. DRSA applied to formulate a set of software quality classification in which the application is classified. Classification established by the results of measurements Metrics of Object Oriented Design (MOOD) against the property of Java-based software. Utilization of DRSA is an alternative method for interpreting the values of metrics and determine a quantitative value that represents the quality relative to other software. Experimental studies indicate that the accuracy of the DRSA depends on the number of sample applications that are used as reference in the definition set down one rule for classifying software properties. Sample applications in the experiment was taken from a proven library Java modularity, usability and maintainability of his. In general, the experimental results concluded that the DRSA can be applied to rank the relative quality of a number of applications in the sample population.

Keywords: *metric for object oriented software, analytic hierarchy process, DRSA*

1. Pendahuluan

Pengukuran merupakan aspek fundamental dalam *engineering*, termasuk *software engineering*. Evaluasi yang objektif dapat dicapai dengan melakukan pengukuran berdasarkan satu set *metrics* yang standar. Dalam paradigma berorientasi objek, sejumlah set *metrics* telah diperkenalkan untuk mengukur derajat penerapan properti desain berbasis objek dalam *software* aplikasi. Set *metrics* ini salah satunya adalah *Metrics for Object Oriented Design* (MOOD) yang diperkenalkan oleh Fernando Brito e Abreu [1].

Dalam penelitiannya, Abreu tidak secara spesifik memberikan suatu nilai ukuran yang dapat menentukan baik atau buruknya hasil perhitungan nilai keluaran dari masing-masing *metrics* yang ada dalam MOOD. Dominasi dari tiap-tiap *metrics* terhadap *metrics* lainnya juga tidak dijelaskan pada penelitian Abreu. Hal ini menyebabkan apa yang menjadi ukuran baik atau tidaknya suatu perangkat lunak berorientasi objek menjadi belum terdefinisi dengan jelas. Banyak parameter-parameter yang mempengaruhi kualitas perangkat lunak, termasuk juga salah satunya adalah preferensi kebutuhan terhadap perangkat lunak tersebut.

Des sudah mengimplementasikan AHP sebagai salah satu dari metode *Multi Criteria Decision Analysis* (MCDA) untuk memecahkan masalah pengukuran tersebut. Dalam penelitiannya, Des [2] menggunakan AHP untuk menentukan peringkat kualitas dari sekumpulan aplikasi berorientasi objek. Selain AHP, perlu dikaji metode lain yang dapat digunakan untuk menentukan peringkat kualitas aplikasi berorientasi objek. *Dominance-based Rough Set Approach* (DRSA) merupakan salah satu metode MCDA yang dapat digunakan untuk menentukan peringkat berdasarkan kriteria-kriteria dari sampel-sampel yang dijadikan referensi untuk penentuan peringkat.

Tujuan dari penelitian ini adalah mengujicobakan penerapan DRSA untuk menentukan peringkat kualitas aplikasi aplikasi berorientasi objek. Selanjutnya dilakukan evaluasi serta penyesuaian terhadap parameter dan langkah-langkah penerapan DRSA sehingga dapat digunakan untuk menjawab pertanyaan: "Bagaimana menentukan peringkat kualitas dari sejumlah aplikasi berorientasi objek relatif satu terhadap yang lainnya?".

Ada dua konsep utama yang memotivasi kami untuk melakukan studi ini. Pertama, OO *metrics* yang belum ditentukan bagaimana menurunkannya menjadi sebuah nilai tunggal dan kedua, DRSA sebagai sebuah pendekatan yang mungkin dapat diterapkan untuk memainkan peran tersebut.

Sejak populernya desain berbasis objek dalam pengembangan *software*, *metrics* desain berbasis objek pun menjadi bagian esensial dalam lingkungan *software*. Telah banyak riset yang ditujukan untuk memformulasikan himpunan *metrics* berbasia objek yang digunakan untuk mengukur kualitas desain berbasis objek. *Metrics* untuk OO difokuskan pada pengukuran yang diterapkan ke karakteristik *class* dan desain [3].

Sejumlah *metrics* berbasis objek yang penting telah dikembangkan dalam literatur. Contohnya, *metrics* yang diusulkan oleh Abreu [4], C.K *metrics* [5], Li and Henry [6] MOOD *metrics*, Lorenz and Kidd [7] *metrics* dsb. Penelitian ini fokus pada pemanfaatan MOOD yang kemudian dianalisa dengan menggunakan DRSA untuk menentukan peringkat kualitas dari aplikasi-aplikasi tersebut. Seperti telah digambarkan dalam [8], *metrics* tersebut antara lain *Attribute Hiding Factor* (AHF), *Method Hiding Factor* (MHF), *Attribute Inheritance Factor* (AIF), *Method Inheritance Factor* (MIF), *Polymorphism Factor* (POF), dan *Coupling Factor* (COF).

Untuk merepresentasikan properti OO [9], *metrics* dapat dikelompokkan dalam properti *encapsulation* (AHF dan MHF), *inheritance* (AIF dan MIF), *polymorphism* (POF), dan *message passing* (COF). *Metrics* ini dan hirarkinya telah dievaluasi menggunakan *Analytical Hierarchy Process* (AHP) dalam [10]. Untuk maksud yang sama, penelitian ini membagikan pengalaman dalam penerapan metode DRSA.

DRSA merupakan perluasan dari teori *Rough Set* pada MCDA. DRSA diperkenalkan oleh Greco, Matarazzo dan Slowinski [11]. Perubahan dasar DRSA dari teori *rough set* klasik terletak pada adanya substitusi terhadap relasi yang tidak bisa dibedakan dengan relasi yang mendominasi. Hal ini memungkinkan DRSA untuk menangani ketidak-konsistenan yang biasa terjadi terhadap suatu kriteria pengukuran dan pengelompokan.

Dikatakan bahwa x mendominasi y dengan $P \subseteq C$, dinotasikan dengan $xD_p y$, jika x lebih baik dari y dalam semua kriteria dari P, $x \geq_q y, \forall_q \epsilon P$. Untuk tiap $P \subseteq C$, hubungan dominasi D_p adalah reflektif dan transitif, artinya sebelumnya telah diurutkan sebagian. Jika $P \subseteq C$ dan $x \epsilon U$, maka

$$D_p^+(x) = \{y \in U : yD_p x\}$$
$$D_p^-(x) = \{y \in U : xD_p y\}$$

masing-masing secara berurut menyatakan *P-dominating set* (kumpulan yang mendominasi *P*) dan *P-dominated set* (kumpulan yang didominasi oleh *P*).

Gagasan utama dari filosofi *rough set* adalah memperkirakan sebuah pengetahuan berdasarkan pengetahuan yang sudah ada. Pada DRSA, pengetahuan yang diperkirakan adalah sebuah kumpulan dari gabungan antara perkiraan atas dan bawah dari *decision classes*, sedangkan pengetahuan yang digunakan untuk melakukan perkiraan adalah kumpulan *P-dominating* dan *P-dominated*.

Dengan dasar perkiraan yang didapatkan dari relasi dominasi, dimungkinkan untuk menurunkan sebuah deskripsi umum dari informasi preferensi yang terdapat di dalam sebuah tabel keputusan yang berisikan kumpulan *decision rules* [12]. *Decision rules* adalah ekspresi dalam bentuk jika [kondisi] maka [akibat], yang menggambarkan bentuk ketergantungan antara kriteria kondisi dan keputusan. Prosedur untuk menghasilkan *decision rules* dari sebuah tabel keputusan menggunakan prinsip pembelajaran induktif. Kita dapat membedakan tiga tipe aturan: pasti, mungkin dan kira-kira. Aturan pasti dihasilkan dari perkiraan bawah dari gabungan *class-class*; aturan mungkin dihasilkan dari perkiraan atas dari gabungan

class-class dan aturan kira-kira dihasilkan dari daerah batas.

Aturan pasti memiliki bentuk sebagai berikut:
Jika $f(x,q_1) \geq r_1$ dan $f(x,q_1) \geq r_2$ dan ... $f(x,q_p) \geq r_p$, maka $x \in CL_t^{\geq}$; Jika $f(x,q_1) \leq r_1$ dan $f(x,q_1) \leq r_2$ dan ... $f(x,q_p) \leq r_p$, maka $x \in CL_t^{\leq}$;

Aturan mungkin memiliki bentuk sebagai berikut:
Jika $f(x,q_1) \geq r_1$ dan $f(x,q_2) \geq r_2$ dan ... $f(x,q_p) \geq r_p$, maka mungkin merupakan anggota $CL_t^{\geq}$; Jika $f(x,q_1) \leq r_1$ dan $f(x,q_2) \leq r_2$ dan ... $f(x,q_p) \leq r_p$, maka mungkin merupakan anggota $CL_t^{\leq}$.

Aturan kira-kira memiliki bentuk sebagai berikut:
Jika $f(x,q_1) \geq r_1$ dan $f(x,q_2) \geq r_2$ dan ... $f(x,q_k) \geq r_k$ dan $f(x,q_{k+1}) \geq r_{k+1}$ dan $f(x,q_{k+2}) \leq r_{k+2}$ dan ... $f(x,q_p) \leq r_p$, maka $x \in CL_s \cup CL_{s+1} \cup CL_t$.

Aturan pasti, mungkin dan kira-kira merepresentasikan pengetahuan pasti, mungkin dan ambigu yang didapatkan dari tabel keputusan.

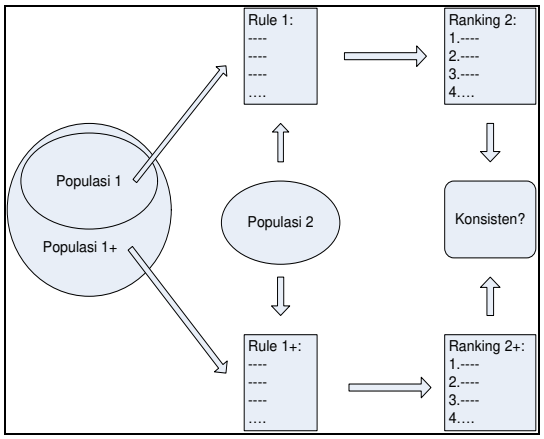
Tiap *decision rule* harus minimal. Karena sebuah *decision rule* merupakan suatu bentuk implikasi, dengan *decision rule* yang minimal kita dapat memahami bahwa tidak ada implikasi lain yang memiliki hubungan dengan kelemahan yang sama (dengan kata lain, tidak ada aturan yang menggunakan *subset* dari kondisi dasar dan/atau kondisi dasar yang lebih lemah) dan sebuah hasil yang kekuatannnya setidaknya sama kuat (dengan kata lain, aturan yang menempatkan objek ke dalam *union* yang sama atau *sub-union* dari *class*).

Sebuah kumpulan *decision rules* dianggap lengkap jika aturan-aturan tersebut dapat mencakup semua objek dari tabel keputusan sedemikian rupa hingga objek-objek yang konsisten di re-klasifikasikan ke *class* awal dan objek-objek yang inkonsisten diklasifikasikan ke dalam kumpulan *class-class* yang merujuk kepada inkonsistensi tersebut. Kita menyebutkan minimal untuk tiap kumpulan *decision rules* yang lengkap dan tidak *redundant*, artinya pengeluaran satu *rules* dari kumpulan ini akan menyebabkan kumpulan ini tidak lengkap. Satu dari tiga strategi induksi dapat digunakan untuk mendapatkan *decision rules* [13]. Pertama, pembentukan deskripsi yang minimal, artinya pembentukan kumpulan aturan yang minimal. Kedua, pembentukan deskripsi yang ekshaustif, artinya pembentukan seluruh aturan untuk sebuah matriks data. Ketiga, pembentukan deskripsi yang memiliki karakteristik, artinya pembentukan sebuah kumpulan aturan yang tiap aturannya mencakup relatif banyak objek, akan tetapi tidak seluruhnya harus objek dari tabel keputusan.

2. Metodologi

Studi dilakukan melalui serangkaian eksperimen terhadap sejumlah aplikasi berbasis Java sebagai *sample*. Populasi yang dipilih sebagai program *sample* dalam riset ini adalah paket *library* dari Sun Java Platform Edition. *Library* ini merupakan paket yang paling sering digunakan untuk mengembangkan apliasi berbasis Java. Paket tersebut didaftar dalam tabel VIII.

JMOOD Calculator telah dikembangkan oleh Rahman [14] berdasarkan hasil kerja Christariny [15] dan Nurmaya [16]. Kalkulator tersebut digunakan untuk mengumpulkan nilai-nilai MOOD dari *sample* aplikasi. Nilai *metrics* dalam MOOD kemudian diproses menggunakan pendekatan DRSA untuk menentukan peringkat kualitas relatifnya.



Gambar 1. Alur pengujian.

Metode pengujian menggunakan DRSA seperti diilustrasikan dalam gambar 1 dilakukan dengan empat tahapan. Pertama, tentukan himpunan aplikasi sebagai Populasi 1 sebagai referensi untuk merumuskan *Rule* 1: java.io, java.lang, java.math, java.net, java.text and java.util. Kedua, tambahkan beberapa aplikasi baru ke Populasi 1, himpunan yang baru disebut Populasi 1+ sebagai referensi untuk merumuskan *Rule* 1+. Dua librari tambahan adalah java.awt dan java.beans. Ketiga, uji himpunan baru aplikasi dalam Populasi 2 terhadap *Rule* yang dihasilkan oleh Populasi 1 terhadap *Rule* 1 maupun *Rule* 1+. Populasi 2 terdiri dari 4 *library*: java.nio, java.security, java.sql dan javax.swing. Keempat, Hasil pengujian dari Populasi 2 dibandingkan untuk memeriksa konsistensi peringkat kualitas yang dihitung terhadap *Rule* 1 dan *Rule* 1+.

3. Hasil dan Pembahasan

MOOD dari setiap *library* dalam Populasi 1 diukur menggunakan JMOOD Calculator. Sebagaimana disarankan oleh Abreu dalam [17], berdasarkan nilai rata-rata, *standard deviation*, maksimum dan minimum dari MOOD, dibuat aturan untuk menentukan kelas *good*, *medium* dan *sufficient*. Aturan tersebut ditampilkan pada tabel I.

Aturan dalam tabel I dipetakan ke hasil pengukuran MOOD pada *Library* 1. Langkah tersebut menghasilkan klasifikasi sebagaimana ditunjukkan dalam tabel III tanpa dua baris terakhir. Dengan tambahan dua *library*, klasifikasi *Library* 1+ ditambilkan oleh tabel II. *Library* 1+ memiliki dua *package* tambahan yaitu *package* jawa.awt and java.beans ke himpunan *Library* 1. Hasil lebih lengkap disajikan oleh tabel III.

Penerapan rentang klasifikasi dalam tabel I untuk *Library* 2 menghasilkan klasifikasi dalam tabel IV, sedangkan penerapan rentang klasifikasi dalam tabel II untuk *Library* 2 menghasilkan klasifikasi dalam tabel V.

Penerapan AHP untuk DRSA: Penentuan peringkat menggunakan DRSA memerlukan *d* untuk setiap aplikasi yang dipakai sebagai referensi dalam pembentukan aturan, mereka adalah *Library* 1 dan *Library* 1+. Nilai *d* kemudian digunakan untuk perbandingan dalam tabel *Pair-wise Comparison* dari DRSA. Oleh karena tidak adanya nilai standar yang merepresentasikan *sample library* dalam populasi, maka digunakan hasil AHP [18] untuk tujuan *d*. Skema AHP menggunakan vektor prioritas dari kriteria MOOD sebagaimana diterapkan oleh Des [2]. Vektor prioritas dapat dilihat dalam tabel VI.

TABEL I
RENTANG NILAI KLASIFIKASI MOOD UNTUK *LIBRARY* 1

Metric	Good	Medium	Sufficient
AHF	x > 0.864	0.642 ≤ x ≤ 0.864	x < 0.642
MHF	0.2898 ≤ x ≤ 0.440	x < 0.290	x > 0.440
AIF	x > 0.302	0.066 ≤ x ≤ 0.302	x < 0.066
MIF	x > 0.650	0.392 ≤ x ≤ 0.650	x < 0.392
POF	x > 0.168	0.074 ≤ x ≤ 0.168	x < 0.074
COF	0 < x ≤ 0.128	x > 0.128	-

TABEL II
RENTANG NILAI KLASIFIKASI UNTUK *LIBRARY* 1+

Metrik	Good	Medium	Sufficient
AHF	x > 0,884	0,617 ≤ x ≤ 0,884	x < 0,617
MHF	0,267 ≤ x ≤ 0,429	x < 0,267	x > 0,429
AIF	x > 0,402	0,077 ≤ x ≤ 0,402	x < 0,077
MIF	x > 0,709	0,429 ≤ x ≤ 0,709	x < 0,429
POF	x > 0,157	0,076 ≤ x ≤ 0,157	x < 0,076
COF	0 < x ≤ 0,108	x > 0,108	-

TABEL III
PEMETAAN NILAI *LIBRARY* 1+

java.*	AHF	MHF	AIF	MIF	POF	COF
io (1)	M	G	M	M	M	G
lang (2)	S	G	M	M	S	G
math (3)	M	S	S	S	M	M
net (4)	M	G	M	M	M	G
text (5)	M	G	M	M	M	G
util (6)	G	G	M	M	G	G
awt (7)	S	G	G	M	M	G
beans (8)	G	G	M	G	M	G

G = Good; M = Medium; S = Sufficient

TABEL VI
Pemetaan nilai kelompok paket *Library* 2 terhadap rentang kelompok paket *Library* 1

java.*	AHF	MHF	AIF	MIF	POF	COF
nio (1')	M	M	G	G	M	G
security (2')	G	M	S	G	M	G
sql (3')	S	M	M	M	G	G
swing (4')	M	G	G	G	S	G

G = Good; M = Medium; S = Sufficient

TABEL V
PEMETAAN NILAI KELOMPOK PAKET *LIBRARY* 2 TERHADAP RENTANG KELOMPOK PAKET *LIBRARY* 1+

java.*	AHF	MHF	AIF	MIF	POF	COF
nio (1')	M	M	G	G	M	G
security (2')	G	M	S	M	S	G
sql (3')	S	M	S	M	G	G
swing (4')	M	G	G	G	S	G

G = Good; M = Medium; S = Sufficient

TABEL VI
VEKTOR PRIORITAS KRITERIA

Komponen	Bobot
AHF	0,537
MHF	0,199
AIF	0,04
MIF	0,127
POF	0,063
COF	0,035

Nilai kualitas berdasarkan perhitungan AHP yang dihasilkan oleh perkalian pengukuran MOOD dengan bobot dalam vektor prioritas dapat terlihat di tabel VII a dan b secara berurutan.

TABEL VII
AHP COMPUTATION FOR (A) *LIBRARY* 1 AND (B) *LIBRARY* 1+

Library	(d) AHP	
	(a)	(b)
java.io (1)	0,572	0,572
java.lang (2)	0,442	0,442
java.math (3)	0,531	0,531
java.net (4)	0,577	0,577
java.text (5)	0,581	0,581
java.util (6)	0,654	0,654
java.awt (7)	-	0,466
java.beans (8)	-	0,676

Dalam penggunaan *Library* 1, DRSA harus mendefinisikan sebuah referensi sebagaimana terlihat di tabel VIII.

TABEL VIII
KELOMPOK *LIBRARY* 1 SEBAGAI *REFERENCE SET*

Lib java.*	*AHF*	*MHF*	*AIF*	*MIF*	*POF*	*COF*	*d(AHP)*
io (1)	M	G	M	M	M	G	0.5724
lang (2)	S	G	M	M	S	G	0.4424
math (3)	M	S	S	S	M	M	0.5313
net (4)	M	G	M	M	M	G	0.5769
text (5)	M	G	M	M	M	G	0.5811
util (6)	G	G	G	M	M	G	0.6538

TABEL IX
HASIL PERHITUNGAN PERINGKAT *LIBRARY* 2 BERDASARKAN RENTANG NILAI *LIBRARY* 1

Library	Strength (S)		Weakness (W)		NFS	Peringkat
	(+, f)	(-, f)	(+, a)	(+-, a)	(S-W)	
nio (1')	1	1	1	1	0	2 atau 3
security (2')	3	3	0	0	6	1
sql (3')	0	0	3	3	-6	4
swing (4')	1	1	1	1	0	2 atau 3

TABEL X
HASIL PERHITUNGAN PERINGKAT *LIBRARY* 2 TERHADAP ATURAN DARI *LIBRARY* 1+

Library	Strength(S)		Weakness(W)		NFS	Peringkat
	(+, f)	(-, f)	(+, a)	(+-, a)	(S-W)	
nio (1')	1	1	2	2	-2	3
security (2')	3	3	0	0	6	1
sql (3')	0	0	3	3	-6	4
swing (4')	2	2	1	1	2	2

Tabel *pair-wise comparison* dibangun sebagai landasan untuk membuat beberapa aturan. Pertama, $CORE_{PCT}$ = {AHF, AIF, POF}. Kedua, RED_{PCT} = {AHF, MHF, AIF, POF}, {AHF, AIF, MIF, POF}, {AHF, AIF, POF, COF}. Ketiga, pilah *Pair-wise Comparison Tabel* yang dihasilkan menjadi 2 bagian besar yaitu $\underline{C}$ *(S) dan* $\underline{C}$ *(*S^c*).* Kualitas aproksimasi dari S dan S^c berdasarkan kriteria dari set C adalah 0,67. Keempat, $D_{\geq}$ - *decision rule*s dan $D_{\leq}$ - *decision rule*s, jika $xP_1^{\geq 1}y$, maka xSy, (1,2), (3,2), (4,2), (5,2), (6,1), (6,2), (6,3), (6,4), (6,5); jika $xP_3^{\geq 1}y$ dan $xP_4^{\geq 1}y$, maka xSy, (1,3), (4,3), (5,3), (6,3); jika $xP_1^{\leq -1}y$, maka xS^cy, (1,6), (2,1), (2,3), (2,4), (2,5), (2,6), (3,6), (4,6), (5,6); jika $xP_3^{\leq -1}y$ dan $xP_4^{\leq -1}y$, maka xS^cy, (3,1), (3,4), (3,5), (3,6); xP_i^0y (dan yP_i^0x) x sama dengan y untuk AHF; xP_i^1y (dan $yP_i^{-1}x$) x lebih baik dari y untuk AHF; xP_i^2y (dan $yP_i^{-2}x$) x jauh lebih baik dari y untuk AHF; xSy jika $AHP(x) \geq AHP(y)$; xS^cy jika $AHP(x) < AHP(y)$.

Dengan perolehan seperti di tabel IV, dapat dihitung *Net Flow Score* (NFS). Menggunakan rumus

$$NFS = ((+,f) + (-,f)) - ((+,a) + (-,a))$$

dapat diturunkan peringkat akhir sebagaimana disajikan dalam tabel IX. Penurunan rumus NFS dijelaskan dalam gambar 2. Menerapkan prosedur yang sama untuk *Library* 2 dengan *Library* 1+ sebagai referensi menghasilkan peringkat seperti ditampilkan dalam Tabel 10.

4. Kesimpulan

Telah ditunjukkan bahwa DRSA sebagai salah satu pendekatan untuk pengambilan keputusan berbasis *multi-criteria*, dapat dimanfaatkan untuk menentukan kualitas aplikasi relatif terhadap sejumlah aplikasi. Studi dilakukan terhadap sejumlah *sample* aplikasi, yaitu Sun Java Platform Edition *libraries*. Dibandingkan dengan AHP, pengukuran yang dihasilkan oleh DRSA tidak dapat dipisahkan dari *sample* yang menjadi referensi. Penggunaan populasi *sample* yang berbeda sebagai referensi dapat mengakibatkan perbedaan peringkat kualitas terhadap *sample* populasi yang diuji. Beberapa kendalam ditemukan. Kendala utama adalah tidak adanya rentang nilai kualitas baku untuk setiap *metrics* dalam MOOD. Ini mengakibatkan beberapa asumsi yang dibuat untuk penentuan rentang nilai kualitas berdasarkan riset sebelumnya.

Riset lanjutan yang perlu dilakukan untuk fokus pada penggunaan DRSA untuk aplikasi-aplikasi berbasis objek. Dalam riset ini, pemeriksaan telah dilakukan terhadap *sample* dari Sun Java Platform Edition *libraries*. Riset berikutnya dapat diarahkan untuk menentukan kualitas dari aplikasi nyata seperti ERP, CRM, SCM atau sejumlah aplikasi berbasis objek dengan menggunakan lebih banyak populasi sehingga menghasilkan *rule* yang lebih baik. Perlu juga dilakukan riset lanjutan untuk memeriksa perbandingan komputasi yang dihasilkan oleh AHP yang dipakai sebagai *d* dengan hasil

komputasi oleh DRSA. Maka, korelasi antara AHP dan DRSA dapat ditentukan.

Riset lebih lanjut dapat juga fokus pada penerapan metode DRSA untuk membangun sebuah sistem pembelajaran dengan mengubah *sample* populasi sehingga *decision rule* yang optimal dapat diturunkan khususnya untuk aplikasi-aplikasi berbasis objek. Untuk memudahkan penggunaan DRSA dalam menentukan peringkat kualitas, sebuah *tool* perlu dikembangkan. Rentang nilai dari MOOD perlu dikaji lebih lanjut untuk menentukan nilai-nilai standar MOOD yang merepresentasikan desain berbasis objek yang dianggap berkualitas.

Untuk memudahkan penggunaan DRSA dalam penentuan peringkat, riset sebaiknya diarahkan untuk mengembangkan *software tool*. Sampai saat ini, belum ada riset yang berusaha mendefinisikan rentang baku kualitas dari setiap *metric* dalam MOOD. Oleh karenanya, riset selanjutnya perlu mengkaji kemungkinan penentuan nilai rentang ideal dari MOOD sesuai klasifikasi yang ditentukan. Selain itu, riset lanjutan perlu menerapkan DRSA dengan kategori menggunakan *metric* dalam MOOD2 yang memiliki lebih banyak jumlah *metric* untuk menghasilkan aturan keputusan yang lebih baik.

Referensi

[1] F. Brito e Abreu, Talk on "Design Metrics for Object-Oriented Software Systems" *In ECOOP'95 Quantitative Methods Workshop*, Aarhus, 1995,

[2] D. Dulianto, *Penerapan AHP dan Software Metric dalam Menentukan Kualitas Disain Software Berorientasi Objek*, Ph. D Thesis, Magister of Information Technology, Faculty of Computer Science, 2008.

[3] M. Sarker, *An Overview of Object Oriented Design Metrics*, Master Thesis, Dept. of Computer Science, Umeä University, Sweden, 2005.

[4] F.B. Abreu, Carapuca, & Regoerio, "Candidate Metrics for Object Oriented Software within a Taxonomy Framework," *Journal of Systems Software 26*, vol. 26, pp. 87-96, 1994.

[5] Chidamber, Shyam, Kemerer, & F. Chris, A Metrics Suite for Object-Oriented Design, M.I.T. Sloan School of Management E53-315, Cambridge, 1993.

[6] W. Li, S. Henry, "Maintenance Metrics for the Object-Oriented Paradigm" *In Proceedings: IEEE-CS International Software Metrics Symposium, Baltimore*, 52-60, 1993.

[7] M. Lorenz & J. Kidd, *Object Oriented Software Metrics*, Prentice Hall, New Jersey, 1994.

[8] F.B. Abreu, "The MOOD Metrics Set" *In Proceeding ECOOP'95 Workshop on Metrics*, pp. 476-493, 1995.

[9] F.B. Abreu, R. Esteves, & M. Goulau, "The Design of Eiffel Programs: Quantitative Evaluation usign The MOOD Metrics" *In Proceeding of TOOLS*, USA, 1996.

[10] P. Mursanto, P, E. Hermawan, & W. Jatmiko, "Measuring Relative Quality of Object Oriented Software Design using Analytic Hierarchy Process" *In Proceeding Indonesia-Japan Joint Scientific Symposium*, pp.89-94, 2008.

[11] S. Greco, B. Matarazzo, & R. Slowinski, *Multicriteria Classification by Dominance Based Rough Set Approach, Handbook of Data Mining and Knowledge Disecovery*, Oxford Univ. Press, New York, 2002.

[12] S. Greco, B. Matarazzo, & R. Słowiński, "Rough sets theory for multicriteria decision analysis, " European Journal of Operational Research, vol. 129, pp. 1-47, 2001.

[13] J. Stefanowski, In: Skowron, Rough Set Knowledge Discovering, *On Rough Set based Approach to Induction of Decision Rules*, Physica Verlag, Heidelberg, pp. 500-529, 1998.

[14] Rahman & Zaki, *The Use of MOOD for Defining Quality of OO Applications*, Ph.D Thesis, Magister of Information Technology, universitas Indonesia, 2009.

[15] Christariny, *Metrics Calculator untuk Sistem Berorientasi Objek*, Final Project, Fasilkom UI, 2004.

[16] Nurmaya, *Metoda Pengukuran Desain Berorientasi Objek Berbasis Metrics for Object Oriented Design (MOOD)*, Final Project, Fasilkom UI, 2007.

[17] F.B. Abreu, H. Zuse, H.A. Sharoui, W.L. Melo, "Quantitative Approaches in Object-Oriented Software Engineering" *ECOOP Workshops*, 1999.

[18] T.L. Saaty, Fundamentals of Decision Making and Priority Theory, RWS Publications, Pittsburgh, 2006.