Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) DOI: 10.25126/jtiik.201961289
Vol. 6, No. 3, Juni 2019, hlm. 229-234 p-ISSN: 2355-7699
Akreditasi KEMENRISTEKDIKTI, No. 30/E/KPT/2018 e-ISSN: 2528-6579

KUANTIFIKASI PENGARUH UNDERSTANDABILITY DAN MAINTAINABILITY PADA EVOLUSI PERANGKAT LUNAK

Mochammad Adhy [1], Bayu Priyambadha [2], Fajar Pradana [3]

[123]Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1] adhymch@gmail.com, [2] bayu_priyambadha@ub.ac.id, [3]fajar.p@ub.ac.id

(Naskah masuk: 26 November 2018, diterima untuk diterbitkan: 13 Februari 2019)

Abstrak

Understandability dipercaya sebagai salah satu faktor yang mempengaruhi proses *maintenance*. Hal ini dikarenakan dalam praktiknya tidak selalu tim pengembang yang sama yang melakukan perbaikan kesalahan pada perangkat lunak. Jika pengembang sebelumnya tidak ada maka pengembang yang baru atau staff *maintenance* perlu untuk memahami sistemnya terlebih dahulu. Sebagai contoh, dalam sebuah percobaan mengenai inspeksi kode, 60% dari isu yang dilaporakan oleh *reviewer* profesional pada *maintenance* terkait dengan *understandability*. Berdasarkan realita tersebut munculah motivasi untuk melakukan penelitian mengukur seberapa besar keterkaitan *understandability* dengan *maintainability* pada evolusi perangkat lunak. Penelitian ini menggunakan pendekatan statistika yaitu *spearman's rank correlation* untuk menganalisis tingkat keterkaitan antara *understandability* dengan *maintainability*. Berdasarkan percobaan yang dilakukan pada tiga macam perangkat lunak, ditemukan bahwa nilai rata-rata keterkaitan *understandability* terhadap *maintainability* pada proses evolusi perangkat lunak sebesar 0,95 yang menjelaskan bahwa korelasi kedua variabel tersebut sangatlah kuat.

Kata kunci: *evolusi perangkat lunak, kualitas perangkat lunak, object-oriented, understandability, maintainability*

QUANTIFICATION OF UNDERSTANDABILITY IMPACT ON MAINTAINABILITY IN SOFTWARE EVOLUTION PROCESS

Abstract

Understandability is believed to be one of the factors that affect maintenance process. This is because in practice it is not always the same development team is tasked to makes improvements to the software. If the previous developer does not exist then a new developer or maintenance staff needs to learn the system first. For example, in the experiment about code inspection, 60% of the issues reported by professional reviewers on maintenance related to understandability. Based on these realities, emerged a motivation to conduct a research related to the measurement of correlation between understandability and maintenance on software evolution. This research uses a statistical approach that is spearman's rank correlation to analyze the level of linkage between understandability and maintainability. From the conducted experiment on three types of software in software evolution process shows that spearman's rank correlation of 0,95 which means understandability has a very strong correlation with maintainability.

Keywords: *software evolution, software quality, object-oriented, understandability, maintainability*

1. PENDAHULUAN

Parameter kualitas *understandability* menjelaskan mengenai seberapa tinggi tingkat kemudahan dalam memahami sebuah modul perangkat lunak yang dikembangkan (Nazir, et al., 2010). *Understandability* dipercaya sebagai salah satu faktor yang mempengaruhi proses *maintenance* (Sommerville, 2011). *Maintenance* perangkat lunak adalah proses umum dalam mengubah sistem setelah sistem tersebut diluncurkan. Perubahan yang dibuat pada perangkat lunak dapat berupa perubahan sederhana seperti memperbaiki kesalahan pada *coding*, perubahan untuk memperbaiki kesalahan pada perancangan, memperbaiki kesalahan pada spesifikasi atau menyediakan kebutuhan baru, perubahan pada implementasi dengan memodifikasi komponen sistem yang sudah ada dan yang seperlunya dan/atau dengan menambahkan komponen baru pada sistem (Sommerville, 2011). Dalam melakukan *maintenance*, 80% jatah waktu digunakan untuk memahami sebuah perancangan atau modul perangkat lunak dan dokumen terkait (Izadkhah & Hooshyar, 2017). Hal ini dikarenakan

dalam praktiknya tidak selalu tim pengembang yang sama yang melakukan perbaikan kesalahan pada perangkat lunak. Jika pengembang sebelumnya tidak ada maka pengembang yang baru atau staff *maintenance* perlu untuk memahami sistemnya terlebih dahulu. Jika sistem sulit dipahami, perubahan yang akan dilakukan bisa saja mengakibatkan kesalahan yang serius dan rentetan perubahan. Hal tersebut dapat menghabiskan banyak biaya dan waktu. Sebagai contoh, dalam sebuah percobaan mengenai inspeksi kode, 60% dari isu yang dilaporakan oleh *reviewer* profesional pada *maintenance* terkait dengan *understandability* (Uchida & Shima, 2004). Penelitian lain menjelaskan bahwa *maintainability* dipengaruhi oleh sub-karakteristik seperti *understandability*, *analyzability*, dan *modifiability* yang diukur berdasarkan kompleksitas yang ada pada UML *class diagram* (Genero & Mario, 2001).

Pada tingkat perancangan UML *class diagram understandability* dipengaruhi oleh beberapa karakteristik *Object-Oriented* (OO) seperti *cohesion*, *coupling*, dan *inheritance*. Dimana berdasarkan penelitian yang dilakukan Nazir hasilnya adalah bahwa faktor seperti *cohesion* memberikan dampak positif terhadap *understandability* ketika nilainya bertambah sedangkan *coupling* dan *inheritance* memberikan dampak negatif terhadap *understandability* jika nilainya bertambah. Sehingga dari penelitian tersebut diketahui bahwa metrik *understandability* bergantung pada *inheritance*, *coupling*, dan *cohesion* (Nazir, et al., 2010). Berdasarkan realita tersebut memunculkan motivasi untuk melakukan penelitian dalam rangka mengobservasi dan membuktikan secara statistik seberapa besar keterkaitan *understandability* dengan *maintainability* pada evolusi perangkat lunak. Penelitian ini bertujuan untuk memberikan gambaran baku berdasarkan data angka mengenai keterkaitan antara *understandability* dengan *maintainability*.

Penelitian ini menggunakan pendekatan statistika yaitu *spearman's rank correlation* untuk menganalisis tingkat korelasi antara *understandability* dengan *maintainability*. Pendekatan *spearman's rank correlation* dipilih karena kedua variable yang akan dibandingkan diambil secara acak dan bersifat monoatomik. Penelitian ini menggunakan data set dari beberapa proyek perangkat lunak yang diambil dari repositori yang ada pada GitHub. Sedangkan untuk mengukur *understandability* menggunakan *tools* SUMIT dan mengukur *maintainability* menggunakan *tools* yaitu SonarQube.

Penelitian ini dibagi kedalam enam bagian, yaitu pendahuluan yang mengemukakan latar belakang penelitian. Bagian kedua membahas metrik yang berkaitan dengan *understandability* dan *maintainability*. Bagian ketiga membahas mengenai cara pengukuran korelasi menggunakan pendekatan statistika. Bagian keempat membahas mengenai metodologi penelitian. Bagian kelima mengenai *pre-processing* data yang digunakan. Bagian keenam adalah pembahasan dari hasil penelitian dan bagian ketujuh adalah kesimpulan dari penelitian yang dilakukan.

2. *UNDERSTANDABILITY METRIC* DAN *MAINTAIABILITY METRIC*

Dalam perhitungan *understandability* terdapat beberapa metrik yang dapat digunakan antara lain seperti *probabilistic model* yang mengukur *understandability* berdasarkan jumlah pegawai dan jumlah percobaan yang dilakukan untuk merekonstruksi sistem perangkan lunak hingga benar (Uchida & Shima, 2004), metrik selanjutnya yaitu *class complexity metric* (CCM) yang mengukur understandability berdasarkan kompleksitas (Rajnish, 2014). Berikutnya metrik *understandability* berdasarkan hirarki *inheritance*. Metrik ini menghitung *understandability* berdasarkan pewarisan dari *super class* ke *sub class* (Kumar & Prasad, 2015). Metrik lainya yaitu menghitung *understandability* menggunakan model regresi linear multi-variabel atau lebih dikenal dengan *multivariate understandability metric*. Metrik ini melibatkan *inheritance*, *coupling*, dan *cohesion* sebagai variabel saling bebas dan *understandability* sebagai variabel terikat (Nazir, 2010).

Dalam pengembangan perangkat lunak untuk memprediksi besar usaha dalam melakukan *maintenance* terdapat beberapa cara. Salah satu caranya yaitu memprediksi *maintainability* berdasarkan *techincal dept* dari proyek yang dikerjakan (Lehman, 1996). *Technical dept* sendiri berkaitan dengan kompleksitas, dokumentasi, *clone*, cacat, ukuran komponen, penambahan fungsionalitas baru, kebergantungan, dan pengetahuan yang kurang (Bogner, et al., 2018). Adapun cara lainnya yaitu dengan menggunakan *fuzzy prototypical knowledge discovery* (Genero, et al., 2001).

3. *SPEARMAN'S RANK CORRELATION*

Koefisien *spearman's rank correlation* merupakan pengukuran statistik yang diperkenalkan oleh Charles Spearman pada tahun 1904 yang digunakan untuk mencari tahu seberapa kuat hubungan monoatomik diantara dua buah data kuantitatif yang dibandingkan. *Spearman's rank correlation* dinotasikan sebagai r_s dan dimodelkan untuk memiliki batasan nilai berupa $-1 \leq r_s \leq 1$. Nilai dari *spearman's rank correlation* diinterpretasikan jika nilai r_s semakin mendekati 1 maka hubungan monoatomiknya pasangan data itu semakin kuat (Zar, 2005).

Berikut ini merupakan perhitungan *spearman's rank correlation* yang ditunjukkan pada persamaan 1 (Zar, 2005):

$$r_s = 1 - \frac{6\sum d^2}{n(n^2-1)} \quad (1)$$

Dimana,

r_s : Koefisien *spearman's rank correlation*
d : Selisih nilai ranking dari data yang dibandingkan
n : Jumlah data

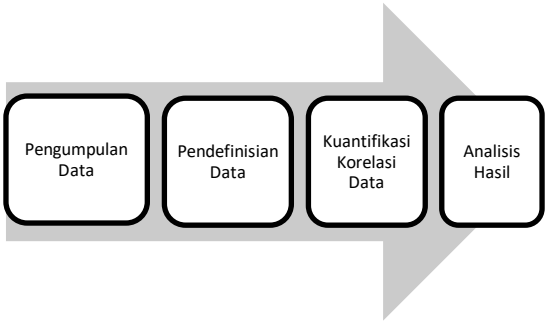
Pada Tabel 1 dapat dilihat tingkat keterkaitan dari *spearman's rank correlation* yaitu sebagai berikut:

Tabel 1 Nilai Tingkatan Spearman's Rank Correlation

Nilai	Predikat
.00-.19	Sangat lemah
.20-.39	Lemah
.40-.59	Sedang
.60-.79	Kuat
.80-1.0	Sangat Kuat

4. METODOLOGI PENELITIAN

Penelitian ini dilakukan kedalam empat buah tahapan yang berurutan yaitu dimulai dari pengumpulan data, pengukuran *understandability* dan *maintainability* data, kuantifikasi korelasi data, dan terakhir adalah analisis hasil.



Gambar 1 Metodelogi Penelitian

4.1 Pengumpulan Data

Data pada penelitian ini diambil dari proyek perangkat lunak pada repositori GitHub yaitu DockerFile-Maven, Jib, dan Hawk. Setiap proyek diambil masing-masing lima buah versi termutakhir untuk dijadikan sampel. Ketiga proyek perangkat lunak tersebut ditulis dalam bahasa pemrograman Java.

4.2 Pendefinisian Data

Pendefinisian data dilakukan untuk mendefinisikan nilai *understandability* dan *maintainability* setiap data proyek perangkat lunak yang telah berhasil dikumpulkan. Dalam mendefinisikan nilai *understandability* digunakan alat bantu easyUML untuk mengasilkan *class diagram* dari kode sumber proyek yang kemudian akan direkonstruksi ulang menggunakan Visual Paradigm. Berkas *class diagram* yang telah terekonstruksi kemudian disimpan kedalam dokumen .xml untuk dihitung nilai *understandability*-nya menggunakan alat bantu SUMIT.

Sedangkan untuk mendefinisikan nilai *maintainability* setiap data proyek digunakan alat bantu SonarQube. Nilai yang dicari menggunakan SonarQube adalah *rating* dan *grade* dari *maintainability* proyek perangkat lunak. Perlu digaris bawahi dalam perhitungan ini baik *understandabaility* maupun *maintainability*, jika nilai semakin naik maka usaha yang dibutuhkan semakin tinggi (Genero & Mario, 2001).

4.3 Kuantifikasi Korelasi Data

Setelah nilai *understandability* dan *maintainability* dari setiap data proyek perangkat lunak telah terdefinisi, korelasi keterkaitan *understandability* dan *maintainability* kemudian diukur menggunakan *spearman's rank correlation*. *Spearman's man rank correlation* digunakan karena data yang dibandingkan tersebar secara *monoatomic*.

Data yang diproses pada tahap ini adalah nilai *maintainability* dan nilai *understandability* dari setiap versi proyek perangkat lunak yang diranking terlebih dahulu, setelah itu dikaitkan satu sama lain dan dihitung besar korelasinya.

4.4 Analisis Hasil

Analisis hasil dilakukan dengan melihat hasil korelasi keterkaitan antara *understandability* dengan *maintainability*. Penggambaran data hasil pemrosesan dan perhitungan akan ditampilkan dalam bentuk tabel dan grafik. Analisis dilakukan dengan melihat data tabel dan grafik yang sudah dibuat, kemudian ditarik sebuah kesimpulan. Setiap tabel dan grafik direpresentasikan sesuai dengan obyek yang diteliti. Pada penelitian ini digunakan 3 obyek teliti berupa proyek perangkat lunak yaitu DockerFile-Maven, Jib, dan Hawk.

5. STUDI KASUS

Dalam penelitian ini dilakukan studi kasus berdasarkan data yang diperoleh dari beberapa repositori perangkat lunak pada GitHub. Perangkat lunak yang diteliti diantaranya ada tiga macam yaitu DockerFile-Maven, Jib, dan Hawk. Masing-masing perangkat lunak diambil kode sumber lima versi yang paling mutakhir.

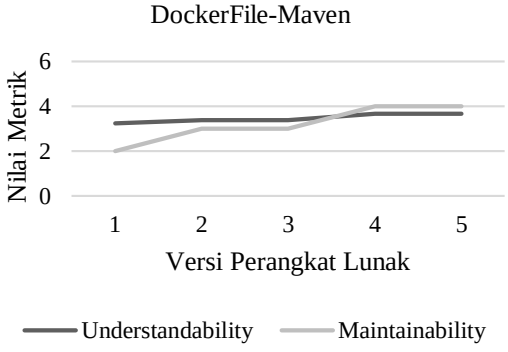
DockerFile-Maven yaitu sebuah ekstensi Maven untuk membantu mengintegrasikan Docker dengan Maven secara lancar. Lima versi termutakhir yang diambil yaitu versi 1.2, 1.3, 1.3.5, 1.4, dan 1.4.3. Jib adalah bagian dari Google Container Tools yang berfungsi untuk *container-building* pada lingkungan pemrograman Java yang sedang berada dalam tahap pengerjaan. Dalam penelitian ini diambil lima versi termutakhir Jib yaitu versi 0.9.3, 0.9.4, 0.9.5, 0.9.6, dan 0.9.7. Data yang terakhir diteliti yaitu Hawk. Hawk merupakan tempat penyimpanan yang aman dan sederhana untuk perangkat Andorid. Lima versi

termutakhir Hawk yang diambil yaitu versi 1.20, 1.21, 1.22, 1.23, dan 2.0. Keseluruhan data yang diambil ditulis menggunakan bahasa pemrograman Java dan dikembangkan menggunakan pendekatan berbasis objek. Untuk DockerFile-Maven dibangun pada proyek Maven. Hawk dibangun pada proyek Gradle. Sedangkan Jib dibangun pada Maven dan Gradle, akan tetapi pada penelitian ini difokuskan pada Jib yang dibangun pada Maven.

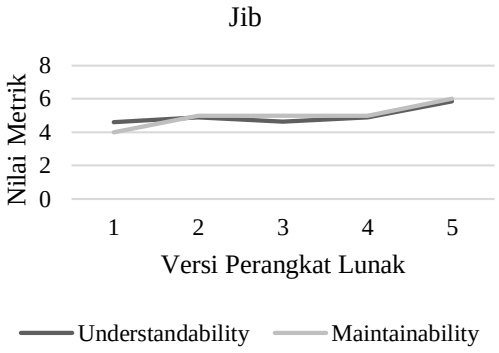
Dengan menggunakan alat bantu SUMIT diperoleh nilai *understandability* berdasarkan *multivariate understandability metric* dari setiap versi masing-masing perangkat lunak yang diteliti. Sedangkan alat bantu lainnya yaitu SonarQube digunakan untuk memperoleh nilai dan *grade maintainability* berdasarkan *technical dept*. Setelah kedua nilai variabel yang akan dikaitkan diperoleh, maka dilakukan proses *ranking* hasil pada nilai *understandability* masing-masing perangkat lunak dan *ranking* hasil pada nilai *maintainability* masing-masing versi perangkat lunak. Berdasarkan *ranking* tersebut kemudian dilakukan analisis korelasi antara *maintainability* dengan *understandability* menggunakan *spearman's rank correlation* untuk mengetahui seberapa kuat keterkaitan kedua variabel tersebut.

6. ANALISIS HASIL

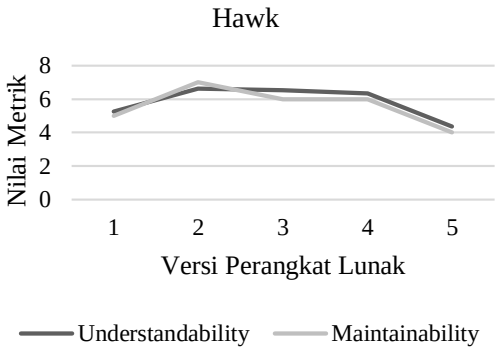
Dari hasil perhitungan *understandability* dan *maintainability* pada masing-masing versi perangkat lunak dapat dilihat bahwa ketika nilai *understandability* mengalami kenaikan maka nilai *maintainability* juga naik. Sebaliknya ketika nilai *understandability* mengalami penurunan maka nilai *maintainability* juga turun. Hal ini dapat dilihat pada Grafik 1 untuk perangkat lunak DockerFile-Maven, Grafik 2 untuk Jib, dan Grafik 3 untuk Hawk. Untuk mengetahui nilai seberapa kuat keterkaitan antara *maintainability* dan *understandability* dilakukan analisis hasil menggunakan pendekatan statistika yaitu dengan *spearman's rank correlation* (r_s). Cara perhitungan r_s dapat dilihat pada persamaan 1.



Grafik 1 Nilai *Understandability* dan *Maintainability* DockerFile-Maven



Grafik 2 Nilai *Understandability* dan *Maintainability* Jib



Grafik 3 Nilai *Understandability* dan *Maintainability* Hawk

Pada perangkat lunak yang pertama yaitu DockerFile-Maven berdasarkan hasil perhitungan *maintainability* dan *understandability* pada Tabel 2 maka disusunlah *ranking maintainability* dan *understandablity* sesuai dengan Tabel 3. Dengan menggunakan *spearman's rank correlation* (r_s) maka ditemukan bahwa untuk korelasi antara *understandability* dengan *maintainability* pada DockerFile-Maven sebesar 0,98 yang menjelaskan bahwa korelasi yang dimiliki sangat kuat.

Tabel 2 Nilai *Understandability* dan *Maintainability* DockerFile-Maven

Versi	***Understandability***	***Maintainability***
1.2	3,25	2
1.3	3,38	3
1.3.5	3,39	3
1.4	3,67	4
1.4.3	3,67	4

Tabel 3 *Rangking* Nilai Korelasi *Understandability* dan *Maintainability* DockerFile-Maven

Versi	***Understandability***	***Maintainability***	**d^2**
1.2	5	5	0
1.3	4	3,5	0,25
1.3.5	3	3,5	0,25
1.4	1,5	1,5	0
1.4.3	1,5	1,5	0
	r_s		**0,98**

Pada perangkat lunak yang kedua yaitu Jib berdasarkan hasil pehitungan *maintainability* dan *understandability* pada Tabel 4 maka disusunlah *ranking maintainability* dan *understandablity* sesuai

dengan Tabel 5. Dengan menggunakan *spearman's rank correlation* (r_s) maka ditemukan bahwa untuk korelasi antara *maintainability* dengan *understandability* pada Jib sebesar 0,90 yang menjelaskan bahwa korelasi yang dimiliki sangat kuat.

Tabel 4 Nilai *Understandability* dan *Maintainability* Jib

Versi	*Understandability*	*Maintainability*
0.9.3	4,60	4
0.9.4	4,90	5
0.9.5	4,63	5
0.9.6	4,89	5
0.9.7	5,86	6

Tabel 5 *Rangking* Nilai Korelasi *Understandability* dan *Maintainability* Jib

Versi	*Understandability*	*Maintainability*	d^2
0.9.3	5	5	0
0.9.4	2	3	1
0.9.5	4	3	1
0.9.6	3	3	0
0.9.7	1	1	0
	r_s		**0,90**

Pada perangkat lunak yang ketiga yaitu Hawk berdasarkan hasil pehitungan *maintainability* dan *understandability* pada Tabel 6 maka disusunlah *ranking maintainability* dan *understandablity* sesuai dengan Tabel 7. Dengan menggunakan *spearman's rank correlation* (r_s) maka ditemukan bahwa untuk korelasi antara *maintainability* dengan *understandability* pada Hawk sebesar 0,98 yang menjelaskan bahwa korelasi yang dimiliki sangat kuat.

Tabel 6 Nilai *Understandability* dan *Maintainability* Hawk

Versi	*Understandability*	*Maintainability*
1.20	5,27	5
1.21	6,63	7
1.22	6,54	6
1.23	6,35	6
2.0	4,35	4

Tabel 7 *Rangking* Nilai Korelasi *Understandability* dan *Maintainability* Hawk

Versi	*Understandability*	*Maintainability*	d^2
1.20	4	4	0
1.21	1	1	0
1.22	2	2,5	0,25
1.23	3	2,5	0,25
2.0	5	5	0
	r_s		**0,98**

Berdasarkan hasil pengamatan dan perhitungan korelasi antara *understandability* dengan *maintainability* yang dilakukan terhadap lima versi mutakhir pada tiga macam perangkat lunak maka ditarik rata-rata nilai korelasi yang mengasilkan nilai sebesar 0,95 yang dapat dilihat pada Tabel 8. Berdasarkan Tabel 1 nilai jangkuan sebesar 0,95 menjelaskan bahwa *understandability* memiliki pengaruh yang sangat kuat terhadap *maintainability* pada evolusi perangkat lunak.

Tabel 8 Rata-Rata Nilai *Spearman's Rank Correlation*

Perangkat Lunak	**r_s**
DockerFile-Maven	0,98
Jib	0,90
Hawk	0,98
Rata-Rata	0,95

7. KESIMPULAN

Parameter kualitas *understandability* dapat ditegaskan melalui penelitian ini sebagai salah satu faktor yang mempengaruhi proses *maintenance*. *Spearman's rank correlation* dinilai mampu untuk mengukur hubungan keterkaitan diantara *understandability* dengan *maintainability*. Dari percobaan yang dilakukan pada tiga macam perangkat lunak dengan masing-masing memiliki lima buah versi paling mutakhir didapatkan nilai rata-rata keterkaitan *understandability* terhadap *maintainability* pada proses evolusi perangkat lunak sebesar 0,95 yang menjelaskan bahwa korelasi kedua variabel tersebut sangatlah kuat.

8. DAFTAR PUSTAKA

BOGNER, J., FRITZSCH, J., WAGNER, S. & ZIMMERMANN, A., 2018. Limiting Technical Debt with Maintainability Assurance – AnIndustry Survey on Used Techniques and Differences with Service- and Microservice-Based Systems. *International Conference on Technical Debt*, pp. 125-133.

GENERO, M. & MARIO, P., 2001. A Controlled Experiment for Corraborating The Usefulness of Class Diagram Metrics. *International Journal of Multimedia and Ubiquitous Engineering*, pp. 369-376.

GENERO, M., OLIVAS, J. A., PIATTINI, M. & ROMERO, F. P., 2001. Fuzzy Prototypical Knowledge Discovery to Predict Information Systems Maintainability.

IZADKHAH, H. & HOOSHYAR, M., 2017. Class Cohesion Metric for Software Engineering: A Critical Review. *Computer Science Journal of Moldova*, pp. 788-804.

KUMAR, D. S. & PRASAD, R., 2015. New Metrics for System Understandability of Inheritance Hierarchies. *International Journal of Research Studies in Computer Science and Engineering*, pp. 59-62.

LEHMAN, M. M., 1996. Metrics and Laws of Software Evolution. Dalam: *Software Process Technology*. Berlin: Springer, pp. 108-124.

NAZIR, M., KHAN, R. A. & MUSTAFA, K., 2010. A Metric Based Model for Understandability

Quantification. *Journal of Computing*, pp. 90-94.

RAJNISH, K., 2014. Class Complexity Metric to Predict Understandability. *International Journal of Information Engineering and Electronic Business*, pp. 69-76.

SOMMERVILLE, I., 2011. *Software Engineering 9th Edition.* Boston: Addison-Wesley.

UCHIDA, S. & SHIMA, K., 2004. An Experiment of Evaluating Software Understandability. *Journal of Systemics, Cybernetics and Informatics*, pp. 7-11.

ZAR, J. H., 2005. Spearman Rank Correlation. Dalam: *Encyclopedia of Biostatistics 2nd Edition.* Hoboken: John Wiley & Sons, pp. 5095-5101.