

Estimasi Proyek Pengembangan Perangkat Lunak dengan Fuzzy Use Case Points

Muhadi Hariyanto

Program Studi Sistem Informasi, STMIK Nusa Mandiri
 muhadi.mho@nusamandiri.ac.id

Romi Satria Wahono

Fakultas Ilmu Komputer, Universitas Dian Nuswantoro
 romi@romisatriawahono.net

Abstract: Perangkat lunak memegang peranan penting agar sebuah komputer atau sistem dapat digunakan, sehingga dibutuhkan manajemen proyek dalam pengembangan perangkat lunak dan salah satu prosesnya adalah melakukan estimasi agar perangkat lunak yang dihasilkan sesuai dengan jadwal dan biaya yang telah ditentukan. Metode use case points banyak dipakai terutama untuk aplikasi yang berbasis obyek, tetapi ditemukan beberapa kelemahan berupa ketidakpastian faktor biaya dan penentuan klasifikasi use case memiliki beda nilai cukup tinggi yang mengakibatkan hasil estimasi kurang akurat. Metode use case points dimodifikasi dengan menambahkan logika fuzzy sehingga menjadi metode fuzzy use case points, komponen yang dimodifikasi yaitu pada penilaian klasifikasi use case. Modifikasi yang dilakukan berupa penentuan nilai use case berdasarkan jumlah transaksi. Dari hasil percobaan yang dilakukan, nilai effort dari metode ini lebih akurat atau mendekati effort aktual. Peningkatan akurasi metode ini mencapai 6 sampai 10%.

Keywords: estimasi proyek software, fuzzy use case point

1. PENDAHULUAN

Perangkat lunak merupakan abstraksi fisik yang memungkinkan kita untuk berbicara dengan mesin perangkat keras (Langer, 2008). Tanpa adanya perangkat lunak, maka perangkat keras yang telah diciptakan tidak akan dapat berguna atau berfungsi dengan optimal.

Untuk menghasilkan perangkat lunak yang berkualitas, dalam pengembangannya perlu adanya manajemen proyek dengan mengikuti pola SDLC (*Software Development Life Cycle*). Salah satu kegiatannya adalah melakukan estimasi. Metode estimasi yang dapat digunakan *Simply Method (Industri Information Based)* (Dennis, 2005), *Function Points Approach* (Albrecht, 1979), dan *Use Case Points* (Karner, 1993).

Penggunaan metode *use case points* memberikan perkiraan *effort* mendekati perkiraan sebenarnya yang dihasilkan oleh pengembang perangkat lunak berpengalaman (Anda, 2002). Ini menandakan pengalaman tim proyek akan mempengaruhi tingkat akurasi estimasi yang dilakukan. *Use case points* juga memiliki kelemahan berupa ketidakpastian faktor biaya dan penentuan klasifikasi yang memiliki beda nilai pengali cukup tinggi (Fan, 2009). Keterbatasan atau kelemahan yang akan mempengaruhi keakuratan estimasi yaitu adanya perbedaan nilai pengali cukup tinggi dari level kompleksitas tiap *use case* (Nassif, 2010).

Estimasi proyek perangkat lunak dengan menggunakan *use case points* mudah digunakan tetapi menjadi kurang akurat karena penentuan kompleksitas *use case* dianggap bersifat linier bila dilihat dari jumlah transaksi setiap *use case*. Sebagai alternatif, digunakan *fuzzy use case points* yang merupakan modifikasi dari *use case points* yaitu dengan menambahkan atau memodifikasi nilai pengali dari klasifikasi jumlah transaksi yang ada di *use case* dengan menggunakan logika fuzzy.

2. PENELITIAN TERKAIT

Mohammed Wajahat Kamal dan Moataz A. Ahmed melakukan perbandingan metode penghitungan *effort* dalam proyek pengembangan perangkat lunak di tahun 2011. Latar belakang masalah yang mendasari mereka melakukan penelitian tersebut karena terdapat beberapa metode yang dapat dilakukan untuk menghitung estimasi *effort* di awal proyek, tetapi akan sulit dilakukan karena data awal kurang akurat dan kurang detail. Mereka melakukan perbandingan beberapa metode, yaitu: *Use Case Points*, *Transactions*, *Paths*, *Extended Use Case Points*, *UCPm*, *Adapted Use Case Points*, *Use Case Size Points*, *Fuzzy Use Case Points*, *Simplified Use Case Points*, dan *Industrial use of Use Case Points*. Dari hasil pengujian metode yang ada, ternyata *Fuzzy Use Case Points* mampu menangani informasi yang tidak atau kurang tepat dan transparansi.

Mohammad Saber Iraj, Majid Aboutalebi dan Hodayun Motameni di tahun 2011 melakukan penelitian untuk menemukan metode yang secara otomatis dapat menentukan tingkat kompleksitas *use case* pada saat melakukan estimasi *effort* dengan menggunakan *use case points*. Latar belakang masalah yang mendasari karena metode *use case points* memiliki kelemahan berupa ketidakpastian dari faktor biaya dan penentuan klasifikasinya kurang detail. Metode yang disarankan oleh mereka adalah menggunakan *Neuro Fuzzy Use Case Points (NFUCP)*, dan dari hasil pengujian menunjukkan kalau memang pengolahan data dengan menggunakan NFUCP lebih akurat dan mendekati *actual effort* dari pengembangan sebuah perangkat lunak.

Ali Bou Nasif, Luiz Fernando Capretz dan Danny Ho melakukan penelitian untuk meningkatkan keakuratan estimasi metode *Use Case Points* dengan teknik *Soft Computing* di tahun 2010. Latar belakang penelitian karena identifikasi dari *International Society of Parametrics Analysis* dan *The Standish Group International* bahwa estimasi yang jelek sebagai salah satu penyebab dari kegagalan pengembangan perangkat lunak. Mereka melakukan modifikasi *Use Case Points* dengan *Fuzzy Logic*

untuk menghitung faktor kompleksitas *Use Case* dan menggunakan *Neural Network* untuk memetakan data input berupa *use case* dan aktor yang terlibat. Dari pengujian yang mereka lakukan, optimasi *Use Case Points* dengan *Fuzzy Logic* dan *Neural Network* meningkatkan keakuratan sampai 22% pada beberapa proyek.

3. LANDASAN TEORI

3.1 Use Case Points

Use case adalah cara formal yang menggambarkan bagaimana sebuah sistem bisnis berinteraksi dengan lingkungannya (Dennis, 2005). *Use Case Points* dikembangkan oleh Gustav Karner di tahun 1993 (Karner, 1993). Untuk melakukan estimasi proyek menggunakan *use case points*, dibuat dahulu sebuah *use case diagram*. Kemudian klasifikasikan aktor sesuai Tabel 1. Hasil perkalian jumlah aktor sesuai klasifikasi aktor disebut *Unadjusted Actor Weight* (UAW).

Tabel 1. *Unadjusted Actor Weighting Table* [2]

Actor Type	Description	Weighting Factor
Simple	External System with well-defined API	1
Average	External system using a protocol based interface	2
Complex	Human	3

Selain aktor, setiap *use case* diklasifikasikan berdasarkan jumlah transaksinya. Hasil perhitungan jumlah *use case* dan klasifikasinya disebut *Unadjusted Use Case Weight* (UUCW).

Tabel 2. *Unadjusted Use Case Weighting Table* [2]

Use Case	Description (Transactions)	Weighting Factor
Simple	1 – 3	5
Average	4 – 7	10
Complex	> 7	15

Nilai UAW dan UUCW dijumlahkan menjadi nilai *Unadjusted Use Case Points* (UUCP). Selain klasifikasi aktor dan *use case*, ada dua jenis faktor yang akan dihitung yaitu *Technical Complexity Factor* dan *Environmental Factor*. Setiap faktor memiliki beban atau nilai pengali yang berbeda-beda. Nilai awal yang diberikan untuk tiap faktor berkisar dari 0 – 5 dengan kriteria diberi nilai 0 apabila faktor tersebut tidak relevan dan 5 jika sangat penting. Sedangkan bila faktor tidak penting dan tidak relevan maka diberi nilai 3.

Tabel 3. *Technical Complexity Factors Table* [2]

No	Description	Weight
T ₁	Distributed Systems	2.0
T ₂	Response time or throughput performance objectives	1.0
T ₃	End-user online efficiency	1.0
T ₄	Complex internal processing	1.0
T ₅	Reusability of code	1.0
T ₆	Easy to install	0.5
T ₇	Ease of use	0.5
T ₈	Portability	2.0
T ₉	Ease of change	1.0
T ₁₀	Concurrency	1.0

No	Description	Weight
T ₁₁	Special security objectives included	1.0
T ₁₂	Direct access for third parties	1.0
T ₁₃	Special user training required	1.0

Tabel 3 digunakan untuk menghitung nilai *Technical Complexity Factor* (TCF) dengan rumus:

$$TCF = 0.6 + \left(0.01 \sum_{i=1}^{13} F_i * W_i \right)$$

Tabel 4. *Environmental Factors Table* [2]

No	Description	Weight
E ₁	Familiarity with system development process being used	1.5
E ₂	Application experience	0.5
E ₃	Object-oriented experience	1.0
E ₄	Lead analyst capability	0.5
E ₅	Motivation	1.0
E ₆	Requirements stability	2.0
E ₇	Part time staff	-1.0
E ₈	Difficulty of programming language	-1.0

Tabel 4 digunakan untuk menghitung nilai *Environmental Factor* (EF) dengan rumus:

$$EF = 1.4 + \left(-0.03 \sum_{i=1}^8 F_i * W_i \right)$$

Nilai UUCP, TCF dan EF digunakan untuk menghitung UCP atau *Unadjusted Case Points* dengan rumus:

$$UCP = UUCP * TCF * EF$$

Hasil UCP sudah menggambarkan ukuran dari sistem yang akan dibuat, langkah selanjutnya menghitung berapa banyak *effort* atau tenaga yang dibutuhkan. Nilai *effort* dapat diketahui dari hasil UCP dibagi dengan nilai PHM (*Person Hour Multiplier*). Nilai PHM diperoleh dengan aturan sebagai berikut:

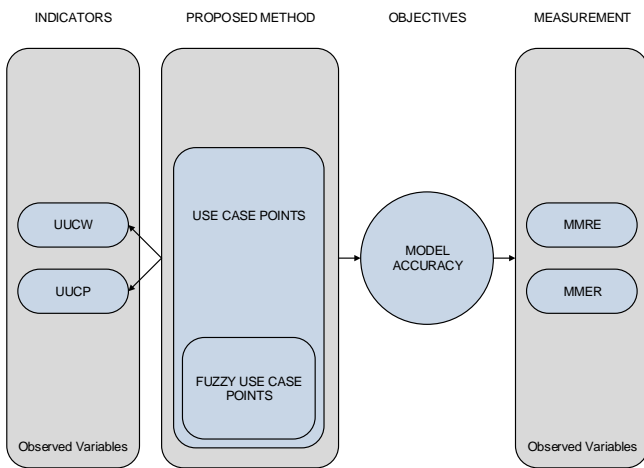
- F₁ = Jumlah E₁ sampai E₆ yang < 3
- F₂ = Jumlah E₇ sampai E₈ yang > 3
- Jika F₁ + F₂ ≤ 2 maka PHM = 20
- Jika F₁ + F₂ = 3 atau 4 maka PHM = 28
- Jika F₁ + F₂ > 4 maka proyek harus dibatalkan

3.2 Fuzzy Logic

Konsep logika *fuzzy* diperkenalkan oleh Prof. Lotfi Zadeh dari Universitas California di Berkeley pada 1965, dipresentasikan bukan sebagai metodologi kontrol, tetapi sebagai suatu cara pemrosesan data dengan memperkenankan penggunaan *partial set membership* dibanding *crisp set membership*.

4 METODE PENELITIAN

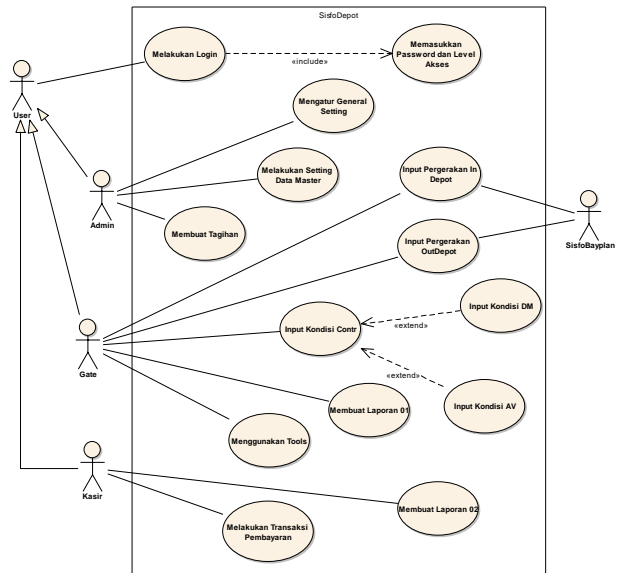
Kerangka pemikiran penelitian ini, seperti ditampilkan dalam Gambar 1, dimulai dari prediksi *effort* dalam proyek perangkat lunak. Penelitian ini menggunakan data enam *project*. Metode yang diusulkan adalah menggunakan *use case points* yang dimodifikasi dengan bantuan logika *fuzzy*.



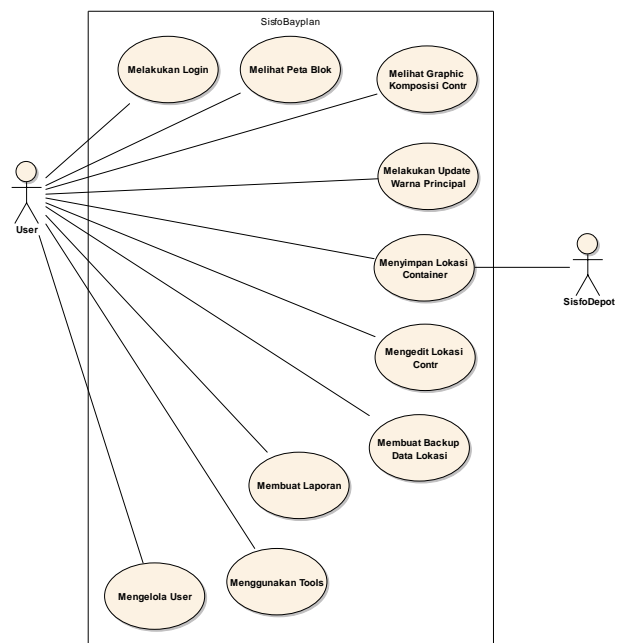
Gambar 1. Kerangka Pemikiran Penelitian

Data dalam penelitian ini diperoleh dengan menggunakan teknik wawancara ke *programmer* dari proyek perangkat lunak yang sedang atau sudah selesai dikerjakan. Data yang didapat berupa *use case metrics* dari enam proyek perangkat lunak.

Semua proyek memiliki *effort* aktual dalam satuan *hours*, nilai *effort* tersebut merupakan jumlah waktu yang diperlukan oleh *programmer* dalam menyelesaikan proyeknya, mulai dari tahap *planning, analysis, design, dan implementation*. Berikut ini diagram *use case* dari enam proyek yang dijadikan bahan penelitian.



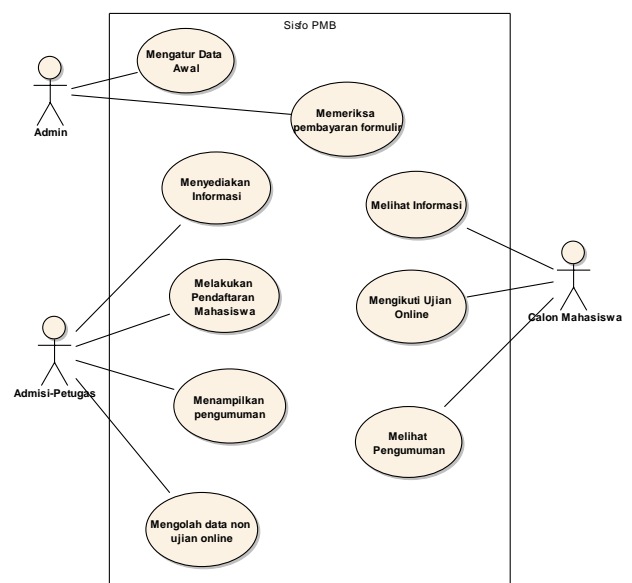
Gambar 3. Diagram Use Case Sistem Informasi Depot



Gambar 4. Diagram Use Case Sistem Informasi Bayplan



Gambar 2. Diagram Use Case Sistem Informasi Sekolah



Gambar 5. Diagram Use Case Sistem Informasi Penerimaan Mahasiswa Baru

5 METODE YANG DIUSULKAN

Metode yang diusulkan adalah dengan menyempurnakan menghitung kompleksitas *use case* dengan menggunakan *fuzzy inference system* metode Mamdani karena memiliki keuntungan lebih intuitif, memiliki penerimaan luas dan cocok untuk *input* manusia (Sivanandam, 2007). Sehingga dengan demikian, *fuzzy inference system* yang dibuat menjadi lebih mudah dalam penggunaan dan pembacaan hasilnya.

Logika *fuzzy* yang dibuat menggunakan representasi kurva segitiga karena bentuk kurva segitiga lebih sederhana dalam penentuan dan mudah divisualisasikan, selain itu dalam pemodelan sistem dinamis dengan menggunakan fungsi segitiga dapat memperkirakan perilaku atau nilai yang hampir presisi di setiap tingkat (Cox, 1994). Variabel masukan terdiri dari tiga kurva segitiga untuk memasukkan jumlah transaksi dari setiap *use case*. Variabel keluaran terdiri dari tiga kurva segitiga yang menghasilkan nilai pengali dari tiap *use case*.

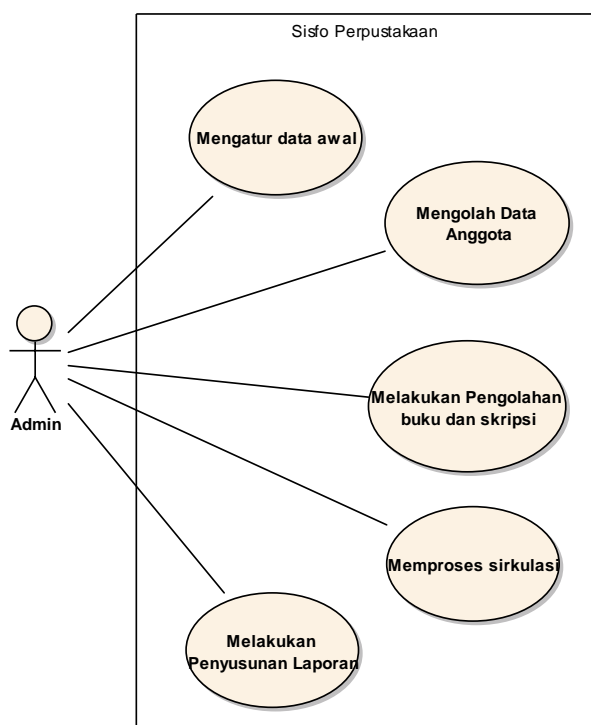
Gambar 8 menunjukkan ada empat kategori data yang digunakan sebagai masukan dalam menghitung estimasi dan akan menghasilkan satu keluaran berupa *effort*. Hanya satu kategori yaitu klasifikasi aktor yang menggunakan *fuzzy inference system*, sedangkan kategori yang lain diolah sesuai dengan metode *use case points*.

Penelitian menggunakan model eksperimen yang menyelidiki hubungan kausal menggunakan data yang dikendalikan sendiri oleh peneliti. *Fuzzy inference system* yang dibuat hanya terdiri dari satu variabel masukan dan satu variabel keluaran dengan menggunakan representasi kurva segitiga.

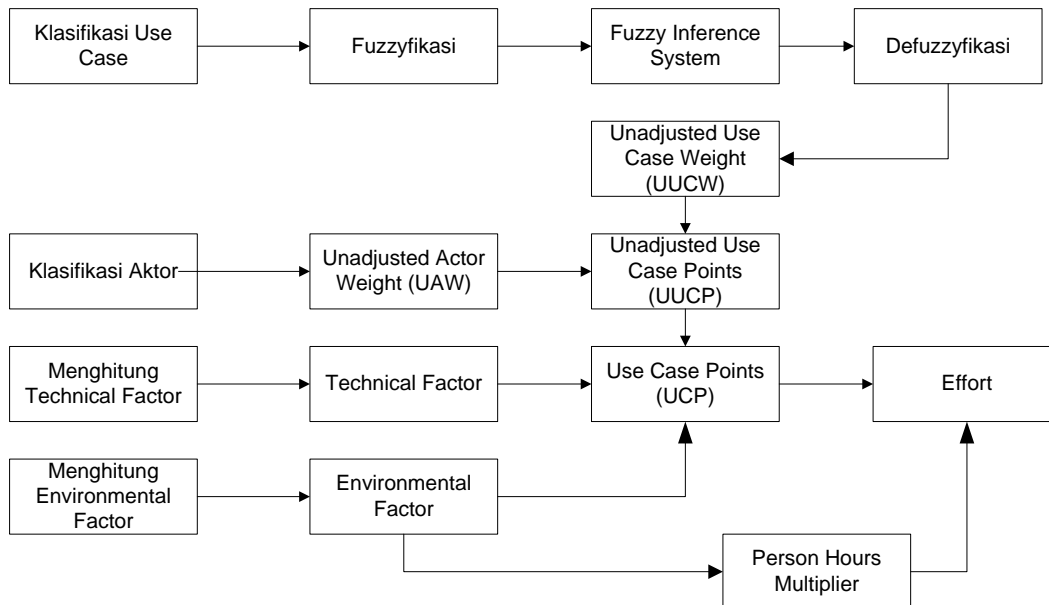
Klasifikasi *use case* dengan menggunakan metode *use case points* dikelompokkan menjadi tiga yaitu *simple*, *average* dan *complex*. Pada metode yang diusulkan dan hasil pengujian metode diperoleh klasifikasi kompleksitas *use case* menjadi 12 kelompok dengan asumsi jumlah maksimal transaksi pada setiap *use case* adalah 12 transaksi.



Gambar 6. Diagram Use Case Sistem Informasi Akademik



Gambar 7. Diagram Use Case Sistem Informasi Perpustakaan



Gambar 8. Metode yang Diusulkan

Tabel 5. Perbandingan Nilai Klasifikasi Use Case

Transaksi di use case	Nilai metode use case	Nilai fuzzy use case
1	5	5,00
2	5	5,00
3	5	6,45
4	10	7,50
5	10	8,55
6	10	10,00
7	10	11,40
8	15	12,50
9	15	13,60
10	15	15,00
11	15	15,00
12	15	15,00

Metode yang diusulkan pada penelitian ini adalah dengan menerapkan logika *fuzzy* yang digunakan untuk menentukan nilai pengali dari pengklasifikasian *use case* sehingga didapat nilai yang lebih bervariasi dan menghasilkan nilai effort yang lebih mendekati nilai effort sebenarnya.

Penelitian ini menggunakan dua metode untuk membandingkan nilai *effort*, yaitu *Mean Magnitude of Relative Error* (MMRE) yang merupakan kriteria yang sangat umum digunakan untuk mengevaluasi model estimasi *effort* perangkat lunak (Briand, 1998). Nilai MMRE merupakan nilai rata-rata dari sejumlah nilai MRE yang didapat dengan rumus:

$$MRE_i = \frac{|Actual Effort_i - Predicted Effort_i|}{Actual Effort_i}$$

Metode kedua adalah *Mean Magnitude of Error Relative* (MMER) sebagai alternatif metode lain untuk mengevaluasi model estimasi *effort* perangkat lunak (Kitchenham, 2001). Nilai MMER merupakan nilai rata-rata dari sejumlah nilai MER yang didapat dengan rumus:

$$MER_i = \frac{|Actual Effort_i - Predicted Effort_i|}{Predicted Effort_i}$$

Ketika menggunakan MMRE dan MMER untuk evaluasi, hasil yang bagus ditunjukkan dengan nilai yang rendah. Terkadang hasil dari MMRE dan MMER kurang akurat sehingga perlu ditambah dengan *Mean Error with Standard Deviation* (Nassif, 2010). Langkah pertama mencari nilai rerata error setiap metode dengan rumus:

$$\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i$$

Dimana

$$x_1 = (Actual Effort_i - Predicted Effort_i)$$

Nilai deviasi diperoleh dengan rumus:

$$SD = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2}$$

Sedangkan nilai *Mean Error with Standard Deviation* didapat dengan rumus: $\bar{x} \pm SD$

6 HASIL PENELITIAN DAN PEMBAHASAN

6.1 Eksperimen dan Pengujian Metode

6.1.1 Metode Use Case

Use case diagram yang terdiri dari *use case* dan aktor menggambarkan apa yang dapat dikerjakan oleh sistem. Sebagai contoh pengujian, diambil data salah satu proyek perangkat lunak yaitu aplikasi Sistem Informasi Depot.

Dari Gambar 3, diketahui terdapat lima aktor dan 14 *use case*. Tahapan yang dilakukan:

1. Klasifikasikan aktor yang terlibat untuk menghasilkan nilai *Unadjusted Actor Weight* (UAW).

Tabel 6. *Unadjusted Actor Weighting Table*

Actor Type	Description	Weighting Factor	Number	Result
Simple	External System with well-defined API	1	--	--
Average	External system using a protocol based interface	2	1	2
Complex	Human	3	4	12
Unadjusted Actor Weight Total (UAW)				14

2. Klasifikasikan *use case* yang ada di *use case* diagram untuk menghasilkan nilai *Unadjusted Use Case Weighting* (UUCW)

Tabel 7. *Unadjusted Use Case Weighting Table*

Use Case Type	Description	Weighting Factor	Number	Result
Simple	1 – 3 transactions	5	6	30
Average	4 – 7 transactions	10	3	30
Complex	> 7 transactions	15	5	75
Unadjusted Use Case Weight Total (UUCW)				135

3. Nilai UAW dan UUCW dijumlahkan menjadi nilai *Unadjusted Use Case Points* (UUCP), sehingga UUCP dari aplikasi tersebut adalah 149.

4. Menghitung nilai faktor teknik untuk memperoleh nilai *Technical Complexity Factor* (TCF).

Tabel 8. *Technical Complexity Factors Table*

Factor Number	Description	Weight	Assigned Value (0–5)	Weighted Value
T ₁	Distributed Systems	2,0	5	10,0
T ₂	Response time or throughput performance objectives	1,0	4	4,0
T ₃	End-user online efficiency	1,0	4	4,0
T ₄	Complex internal processing	1,0	4	4,0
T ₅	Reusability of code	1,0	2	2,0
T ₆	Easy to install	0,5	5	2,5
T ₇	Ease of use	0,5	3	1,5
T ₈	Portability	2,0	1	2,0
T ₉	Ease of change	1,0	3	3,0
T ₁₀	Concurrency	1,0	2	2,0
T ₁₁	Special security objectives included	1,0	2	2,0
T ₁₂	Direct access for third parties	1,0	5	5,0
T ₁₃	Special user training required	1,0	3	3,0
Technical Factor Value (TFactor)				45,0

Rumus yang digunakan untuk mendapatkan nilai *Technical Complexity Factor*:

$$TCF = 0,6 + (0,01 * TFactor)$$

$$TCF = 0,6 + (0,01 * 45)$$

$$TCF = 1,05$$

5. Menghitung nilai faktor lingkungan untuk memperoleh nilai *Environmental Factor* (EF)

Tabel 9. *Environmental Factor Table*

Factor Number	Description	Weight	Assigned Value (0–5)	Weighted Value
E ₁	Familiarity with system development process being used	1,5	4	6,0
E ₂	Application experience	0,5	3	1,5
E ₃	Object-oriented experience	1,0	4	4,0
E ₄	Lead analyst capability	0,5	4	2,0
E ₅	Motivation	1,0	3	3,0
E ₆	Requirements stability	2,0	4	8,0
E ₇	Part time staff	-1,0	0	0,0
E ₈	Difficulty of programming language	-1,0	3	-3,0
Environmental Factor Value (EFactor)				21,5

Rumus yang digunakan untuk mendapatkan nilai *Environmental Factor*:

$$EF = 1,4 + (-0,03 * EFactor)$$

$$EF = 1,4 + (-0,03 * 21,5)$$

$$EF = 0,76$$

6. Nilai UUCP, TCF dan EF digunakan untuk menghitung nilai *Unadjusted Case Points* dengan rumus:
 $UCP = UUCP * TCF * EF$
 $UCP = 149 * 1,05 * 0,76$
 $UCP = 118,90$
7. Berdasarkan data di Tabel 9, dapat diketahui nilai *Person Hour Multiplier* (PHM) adalah 20.
8. *Effort* aplikasi didapat dengan menggunakan rumus:

$$Effort = UCP * PHM$$

$$Effort = 118,90 * 20$$

$$Effort = 2.378$$

Dari hasil penghitungan estimasi proyek perangkat lunak sistem informasi depot dengan menggunakan metode *use case points* diperoleh hasil *effort* adalah sebesar 2.378 jam. Hasil pengujian semua proyek yang ada dengan metode *use case points* dapat dilihat pada Gambar 10.

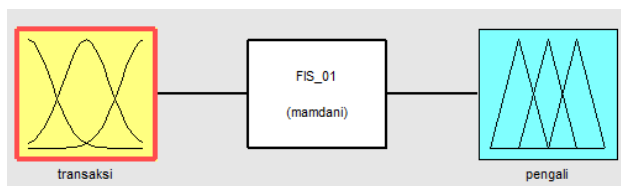
No	Nama Proyek	UAW	UUCW	UUCP	TFactor	Efactor	UCP	PHM	Effort
1	PROYEK 01 - SISTEM INFORMASI SEKOLAH	12.00	140.00	152.00	1.17	0.85	151.16	20	3023
2	PROYEK 02 - SISTEM INFORMASI DEPOT	14.00	135.00	149.00	1.05	0.76	118.90	20	2378
3	PROYEK 03 - SISTEM INFORMASI BAYPLAN	5.00	80.00	85.00	1.14	0.82	79.46	20	1589
4	PROYEK 04 - SISTEM INFORMASI PENERIMAAN MAHASISWA BARU	9.00	65.00	74.00	1.16	0.76	65.24	20	1305
5	PROYEK 05 - SISTEM INFORMASI AKADEMIK	12.00	135.00	147.00	1.09	0.85	136.20	20	2724
6	PROYEK 06 - SISTEM INFORMASI PERPUSTAKAAN	3.00	50.00	53.00	1.16	0.85	52.26	20	1045

Gambar 10. Hasil Pengujian dengan Metode Use Case Points

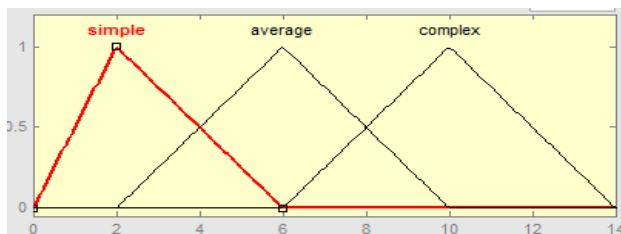
6.1.2 Metode Fuzzy Use Case Points

Metode *fuzzy use case points* yang digunakan dalam penelitian ini merupakan modifikasi metode *use case points* dengan menggunakan logika *fuzzy* untuk memberikan nilai klasifikasi *use case*.

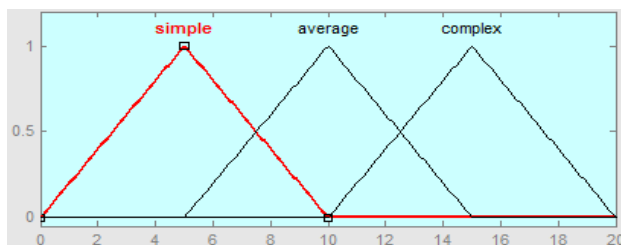
Logika *fuzzy* dibuat menggunakan *fuzzy inference system* metode Mamdani, terdiri dari variabel masukan dan keluaran yang masing-masing terdiri dari tiga kurva segitiga.



Gambar 11. FIS Metode Mamdani



Gambar 12. Variabel Masukan



Gambar 13. Variabel Keluaran

Logika *fuzzy* yang dibuat juga memerlukan pengaturan rules sebagai berikut:

- If transaksi is simple then pengali is simple*
- If transaksi is average then pengali is average*
- If transaksi is complex then pengali is complex*

Dari hasil pengujian logika dengan memberikan nilai masukan berupa jumlah transaksi akan menghasilkan nilai pengali sesuai Tabel 10.

Tabel 10. Hasil Keluaran Logika Fuzzy

Nilai masukan (transaksi)	Nilai keluaran (pengali)
1	5,00
2	5,00
3	6,45
4	7,50
5	8,55
6	10,00
7	11,40
8	12,50
9	13,60
10	15,00
11	15,00
12	15,00

6.2 Evaluasi dan Validasi Hasil

Dengan menggunakan data dari aplikasi yang sama, dilakukan penelitian untuk menghitung *effort* dengan menggunakan logika *fuzzy*. Tahapan yang dilakukan untuk menghitung *effort* dengan menggunakan logika *fuzzy* tidak banyak berbeda dengan metode *use case points*, yang membedakan hanya ketika mencari nilai *unadjusted use case weight*. Untuk menentukan nilai *unadjusted use case weight* menggunakan Tabel 10.

Nilai *UAW*, *TCF*, *EF* dan *PHM* menggunakan nilai yang diperoleh dengan metode *use case points*. Sedangkan nilai *UUCW* untuk aplikasi sistem informasi depot apabila menggunakan logika *fuzzy* tercantum pada Tabel 11.

Tabel 11. Klasifikasi *use case* dengan logika *fuzzy*

No	<i>Use case</i>	Trans aksi	Nilai
1	Melakukan <i>login</i>	1	5,00
2	Memasukkan <i>password</i> dan level akses	1	5,00
3	Mengatur <i>general setting</i>	2	5,00
4	<i>Input</i> pergerakan <i>in depot</i>	3	6,45
5	<i>Input</i> pergerakan <i>out depot</i>	5	8,55
6	<i>Input</i> kondisi <i>contr</i>	2	5,00
7	<i>Input</i> kondisi DM	8	12,50
8	<i>Input</i> kondisi AV	2	5,00
9	Melakukan transaksi pembayaran	8	12,50
10	Membuat laporan 1	10	15,00
11	Membuat laporan 2	10	15,00
12	Melakukan konfigurasi data master	10	15,00
13	Menggunakan <i>tools</i>	5	8,55
14	Membuat tagihan	7	11,40
Nilai <i>unadjusted use case weight</i>			129,95

Dari hasil pengujian diperoleh nilai *UUCW* adalah 129,95. Dengan data yang ada, nilai *UAW* adalah 14, sehingga nilai *UUCP* menjadi 143,95 dan nilai *UCP* 114,87.

Effort yang didapat dengan metode *fuzzy use case points* adalah 2.297 jam. Hasil lengkap pengujian dengan metode *fuzzy use case points* dapat dilihat pada Gambar 14.

No	Nama Proyek	UAW	FUUCW	FUUCP	TFactor	Efactor	FUCP	PHM	Effort FUCP
1	PROYEK 01 - SISTEM INFORMASI SEKOLAH	12.00	123.90	135.90	1.17	0.85	135.15	20	2703
2	PROYEK 02 - SISTEM INFORMASI DEPOT	14.00	129.95	143.95	1.05	0.76	114.87	20	2297
3	PROYEK 03 - SISTEM INFORMASI BAYPLAN	5.00	67.50	72.50	1.14	0.82	67.77	20	1355
4	PROYEK 04 - SISTEM INFORMASI PENERIMAAN MAHASISWA BARU	9.00	63.60	72.60	1.16	0.76	64.00	20	1280
5	PROYEK 05 - SISTEM INFORMASI AKADEMIK	12.00	123.60	135.60	1.09	0.85	125.63	20	2513
6	PROYEK 06 - SISTEM INFORMASI PERPUSTAKAAN	3.00	47.05	50.05	1.16	0.85	49.35	20	987

Gambar 14. Hasil Pengujian dengan Metode *Fuzzy Use Case Points*

6.3 Pembahasan

Semua data *use case metrics* dari enam aplikasi dihitung dengan menggunakan metode *use case points* dan *fuzzy use case points* sehingga diperoleh data *effort* untuk setiap aplikasi. *Effort* sesungguhnya dibandingkan dengan *effort* dari metode *use case points* dan *fuzzy use case points* menggunakan rumus membandingkan nilai *effort* yaitu

MMRE (Briand, 1998) dan MMER (Kitchenham, 2001) serta dilengkai pula dengan rumus *Mean Error with Standard Deviation* (Nassif, 2010).

Hasil perhitungan *effort* dari enam proyek perangkat lunak terlihat pada Gambar 15.

No	Nama Proyek	Effort			MER		MRE		Error	
		Actual	UCP	FUCP	UCP	FUCP	UCP	FUCP	UCP	FUCP
1	PROYEK 01 - SISTEM INFORMASI SEKOLAH	2200	3023	2703	0.27	0.19	0.37	0.23	823	503
2	PROYEK 02 - SISTEM INFORMASI DEPOT	2000	2378	2297	0.16	0.13	0.19	0.15	378	297
3	PROYEK 03 - SISTEM INFORMASI BAYPLAN	1000	1589	1355	0.37	0.26	0.59	0.36	589	355
4	PROYEK 04 - SISTEM INFORMASI PENERIMAAN MAHASISWA BARU	1200	1305	1280	0.08	0.06	0.09	0.07	105	80
5	PROYEK 05 - SISTEM INFORMASI AKADEMIK	2400	2724	2513	0.12	0.04	0.13	0.05	324	113
6	PROYEK 06 - SISTEM INFORMASI PERPUSTAKAAN	900	1045	987	0.14	0.09	0.16	0.10	145	87
Rerata					0.19	0.13	0.26	0.16	394.00	239.17
Standar Deviasi									249.25	158.52
Improvement						+6%	+10%			

Gambar 15. Perbandingan Hasil Pengujian dengan Dua Metode

Dari hasil pengujian menghitung estimasi proyek perangkat lunak antara metode *use case point* dan *fuzzy use case point*, ditemukan perbedaan hasil yang mempengaruhi akurasi. Peningkatan akurasi mendekati *effort* aktual menggunakan *fuzzy use case points* sebesar 6% untuk MMER dan 10% bila menggunakan metode pengukuran MMRE. Rerata standar deviasi kesalahan metode *use case points* sebesar $394,00 \pm 249,25$ sedangkan metode *fuzzy use case points* sebesar $239,17 \pm 158,52$. Nilai rerata standar deviasi kesalahan metode *fuzzy use case points* lebih kecil bila dibandingkan dengan nilai metode *use case points*. Dengan demikian, dari tiga metode pengukuran dapat terlihat bila

menghitung estimasi dengan logika *fuzzy* akan lebih mendekati *effort* aktual.

6.4 Tools

Dalam pengolahan data pada penelitian ini, penulis merancang sebuah aplikasi berbasis *web* dengan menggunakan bahasa pemrograman PHP sebagai alat bantu untuk mempermudah dalam proses perhitungan *effort* yang menggunakan metode *use case points* dan *fuzzy use case points*.

Aplikasi ini hanya digunakan untuk menyimpan data *use case metrics* dari enam proyek perangkat lunak kemudian

diolah dengan rumus dari metode *use case points* dan *fuzzy use case points*, Agar aplikasi dapat menghitung dengan metode *fuzzy use case points*, maka hasil pengolahan *fuzzy inference system* di masukkan ke dalam sistem.



Gambar 16. Tampilan Awal Aplikasi



Gambar 17. Halaman *Fuzzy Use Case Points*



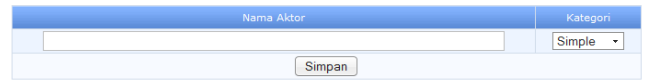
Gambar 18. Halaman Project



Gambar 19. Halaman Menambahkan Proyek Baru

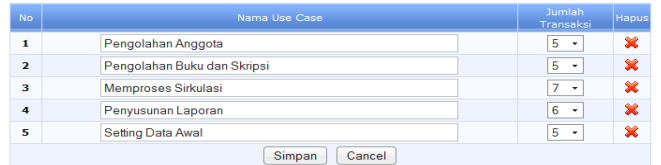


TAMBAH AKTOR



Gambar 20. Halaman Menambah dan Mengubah Data Aktor

USE CASE



TAMBAH USE CASE



Gambar 21. Halaman Edit Nama dan Jumlah Transaksi Setiap *Use Case*

TECHNICAL FACTOR

No	Code	Technical Factor	Multipier	Value	Description
1	T01	Distributed systems	2.0	5	The architecture of the solution may be centralized or single-tenant , or it may be distributed (like an n-tier solution) or multi-tenant. Higher numbers represent a more complex architecture.
2	T02	Response time or throughput performance objectives	1.0	4	The quickness of response for users is an important (and non-trivial) factor. For example, if the server load is expected to be very low, this may be a trivial factor. Higher numbers represent increasing importance of response time (a search engine would have a high number, a daily news aggregator would have a low number).
3	T03	End user online efficiency	1.0	4	Is the application being developed to optimize on user efficiency, or just capability? Higher numbers represent projects that rely more heavily on the application to improve user efficiency.
4	T04	Complex internal processing	1.0	4	Is there a lot of difficult algorithmic work to do and test? Complex algorithms (resource leveling, time-domain systems analysis, OLAP cubes) have higher numbers. Simple database queries would have low numbers.
5	T05	Reusability of code	1.0	2	Is heavy code reuse an objective or goal? Code reuse reduces the amount of effort required to deploy a project. It also reduces the amount of time required to debug a project. A shared library function can be re-used multiple times, and fixing the code in one place can resolve multiple bugs. The higher the level of re-use, the lower the number.
6	T06	Easy to install	0.5	5	Is ease of installation for end users a key factor? The higher the level of competence of the users, the lower the number.
7	T07	Ease of use	0.5	3	Is ease of use a primary criteria for acceptance? The greater the importance of usability, the higher the number.
8	T08	Portability	2.0	1	Is multi-platform support required? The more platforms that have to be supported (this could be browser versions, mobile devices, etc. or

Gambar 22. Halaman Untuk Mengisi Nilai *Technical Factor*

ENVIRONMENTAL FACTOR

No	Code	Environmental Factor	Multipier	Value	Description
1	E1	Familiarity with system development process being used	1.5	4	How much experience does your team have working in this domain? The domain of the project will be a reflection of what the software is intended to accomplish, not the implementation language. In other words, for an insurance compensation system written in java, you care about the team's experience in the insurance compensation space - not how much java they've written. Higher levels of experience get a higher number.
2	E2	Application experience	0.5	3	How much experience does your team have with the application. This will only be relevant when making changes to an existing application. Higher numbers represent more experience. For a new application, everyone's experience will be 0.
3	E3	Object-oriented experience	1.0	1	How much experience does your team have at OO? It can be easy to forget that many people have no object oriented programming experience if you are used to having it. A user-centric or use-case-driven project will have an inherently OO structure in the implementation. Higher numbers represent more OO experience.
4	E4	Lead analyst capability	0.5	4	How knowledgeable and capable is the person responsible for the requirements? Bad requirements are the number one killer of projects - the Standish Group reports that 40% to 60% of defects come from bad requirements. Higher numbers represent increased skill and knowledge.
5	E5	Motivation	1.0	4	How motivated is your team? Higher numbers represent more motivation.
6	E6	Requirements stability	2.0	4	Changes in requirements can cause increases in work. The way to avoid this is by planning for change and instituting a timing system for managing those changes. Most people don't do this, and some rework will be unavoidable. Higher numbers represent more change (or a less effective system for managing change). Note, the multiplier for this number is negative.

Gambar 23. Halaman Untuk Mengisi Nilai *Environmental Factor*

PERBANDINGAN HASIL METODE USE CASE POINTS DAN FUZZY USE CASE POINTS

No	Nama Proyek	Effort		MER		MRE		Error		
		Actual	UCP	FUCP	UCP	FUCP	UCP	FUCP	UCP	FUCP
1	PROYEK 01 - SISTEM INFORMASI SEKOLAH	2200	3023	2703	0.27	0.19	0.37	0.23	823	503
2	PROYEK 02 - SISTEM INFORMASI DEPOT	2000	2378	2297	0.16	0.13	0.19	0.15	378	297
3	PROYEK 03 - SISTEM INFORMASI BAYPLAN	1000	1589	1355	0.37	0.26	0.59	0.36	589	355
4	PROYEK 04 - SISTEM INFORMASI PENERIMAAN MAHASISWA BARU	1000	1164	1139	0.14	0.12	0.16	0.14	164	139
5	PROYEK 05 - SISTEM INFORMASI AKADEMIK	2400	2724	2513	0.12	0.04	0.13	0.05	324	113
6	PROYEK 06 - SISTEM INFORMASI PERPUSTAKAAN	900	1045	987	0.14	0.09	0.16	0.10	145	87
Rerata				0.20	0.14	0.27	0.17	403.83	249.00	
Standar Deviasi								238.59	149.94	
Improvement				+6%		+10%				

Back Hasil Metode Use Case Points Hasil Metode Fuzzy Use Case Points

Gambar 24. Halaman Hasil Proses Use Case Metrics

7 KESIMPULAN

Menghitung estimasi proyek pengembangan perangkat lunak dengan menggunakan metode *fuzzy use case points* memberikan kenaikan akurasi mendekati *effort* sebenarnya bila dibandingkan dengan menggunakan *use case points*. Kenaikan akurasi sebesar 6% bila menggunakan metode perhitungan MMR dan 10% dengan metode MMRE. Nilai standar deviasi kesalahan juga lebih kecil metode *fuzzy use case points* yaitu sebesar $239,17 \pm 158,52$. Sedangkan nilai standar deviasi kesalahan metode *use case points* sebesar $394,00 \pm 249,25$. Sehingga dapat disimpulkan bahwa estimasi menggunakan metode *fuzzy use case points* lebih mendekati *effort* sebenarnya dalam proses menghitung estimasi pengembangan perangkat lunak.

DAFTAR REFERENSI

- Anda, Bente., (2002). Comparing Effort Estimates Based on Use Case Points with Expert Estimates. *Empirical Assessment in Software Engineering (EASE 2002)*.
- Away, Gunaidi Abdia.,(2010). *The Shortcut of Matlab Programming*. Bandung: Informatika.
- Albrecht, Allan J., (1979). Measuring Application Development Productivity. *Joint SHARE/GUIDE/IBM Application Development Symposium*.
- Berndtsson, M., Hansson, J., Olsson, N., & Lundell, B. (2008). *A Guide for Students in Computer Science and Information Systems* (2nd ed). London: Springer.
- Briand, Lionel C., Emam, Khaled El., Surmann, Dagmar., Wiczorek, Isabella., & Maxwell, Katrina D., (1998). An Assessment and Comparison of Common Software Cost Estimation Modeling Techniques. *International Software Engineering Research Network Technical Report ISERN-98-27*.
- Cox, Earl., (1994). *The Fuzzy System Handbook (A Practitioner's Guide to Building, Using, and Maintaining Fuzzy Systems)*. Massachusetts: Academic Press, Inc.
- Dennis, A., Wixom, B. H., & Tegarden, D., (2005). *System Analysis and Design with UML Version 2.0 An Object-Oriented Approach* (2nd ed). USA: John Wiley & Sons, Inc.
- Dawson, C. W. (2009). *Projects in Computing and Information System A Student's Guide* (2nd ed). England: Addison-Wesley
- Fan, Wang., Xiahou, Yang., Xiachun, Zhu., & Lu, Chen. (2009). Extended Use Case Points Method for Software Cost Estimation. *The Center of Technology and Business Innovation*.
- Gray, D.E. (2004). *Doing Research in the Real World*. India: SAGE
- Gustafson, David A. (2002). *Theory and Problems of Software Engineering*. USA: McGraw-Hill.
- Iraji, Mohammad Saber., Aboutalebi, Majid., & Motameni, Homayun., (2012). Effort Estimate with Neuro Fuzzy Use Case Point Based on Exact Weights. *Progress in Computing Applications Volume 1 Number 1, March 2012*.

- Langer, Arthur M. (2008). *Analysis and Design of Information Systems* (3rd ed). London: Springer
- Lee, Kwang H. (2005). *First Course on Fuzzy Theory and Applications*. German: Springer.
- Lily, John H., (2010). *Fuzzy Control and Identification*. New Jersey: John Wiley
- Lughofer, Edwin. (2011). *Envolving Fuzzy Systems, Methodologies, Advanced Concepts and Applications*. German: Springer.
- Kamal, Mohammed Wahajat., & Ahmed, Moataz A., (2011). A Proposed Framework for Use Case based Effort Estimation using Fuzzy Logic: Building upon the outcomes of a Systematic Literature Review. *International Journal on New Computer Architectures and Their Applications (IJNCAA) 1(4):976-999 The Society of Digital Information and Wireless Communications, 2011 (ISSN: 2220-9085)*.
- Karner, Gustav. (1993). Resource Estimation for Objectory Projects. *Objective Systems SF AB, Rational Software*.
- Kitchenham, B.A., Pickard, L.M., MacDonell, S.G., & Shepperd, M.J., (2001). What accuracy statistics really measure. *IEE Proc-Softw., Vol. 148, No. 3, June 2001*.
- Kothari, C. R. (2004). *Research Methodology Methodes and Technique* (2nd ed). India: New Age International
- Kusumadewi, S., & Purnomo, H., (2010). *Aplikasi Logika Fuzzy untuk Pendukung Keputusan* (2nd ed). Yogyakarta: Graha Ilmu
- Nassif, Ali Bou., Capretz, Luiz Fernando., & Ho, Danny., (2010). Enhancing Use Case Points Estimation Method Using Soft Computing Techniques. *Journal of Global Research in Computer Science Volume 1, No. 4, November 2010*
- Sivanandam, S.N., Sumathi, S., & Deepa, S.N., (2007). *Introduction to Fuzzy Logic using MATLAB*. German: Springer.
- Sommerville, Ian. (2007). *Software Engineering* (8th ed). England: Addison-Wesley
- Webopedia Computer Dictionary. (n.d). October 10, 2012. http://www.webopedia.com/TERM/F/fuzzy_logic.html
- Yeates, D., & Wakefield, T. (2004). *System Analysis and Design*. England: Pearson Education Limited.

BIOGRAPHY OF AUTHORS



Muhadi Hariyanto. Menyelesaikan pendidikan S1 dan S2 di STMIK Nusa Mandiri, Jakarta. Saat ini bekerja sebagai programmer di Universitas Tarumanagara dan sebagai dosen S1 di STMIK Nusa Mandiri. Research interest di bidang software engineering.



Romi Satria Wahono. Menempuh pendidikan S1 (B.Eng), S2 (M.Eng) di bidang Software Engineering di Department of Computer Science di Saitama University, Jepang. Menyelesaikan pendidikan S3 (Ph.D) bidang Software Engineering di Universiti Teknikal Malaysia Melaka. Founder dan CEO PT Brainmatics, IlmuKomputer.Com dan

Intelligent Systems Research Center. Mengajar di beberapa program pasca sarjana ilmu komputer di Indonesia, salah satunya di Universitas Dian Nuswantoro. Research interest di bidang Software Engineering dan Machine Learning. Professional member dari IEEE CS, ACM dan PMI.