Terakreditasi SINTA Peringkat 4
Surat Keputusan Direktur Jenderal Pendidikan Tinggi, Riset, dan Teknologi, Nomor: 158/E/KPT/2021
masa berlaku mulai Volume 5 Nomor 2 Tahun 2018 sampai Volume 10 Nomor 1 Tahun 2023

Terbit online pada laman web jurnal: **https://jurnal.plb.ac.id/index.php/tematik/index**

TEMATIK
Jurnal **Teknologi Informasi Komunikasi (e-Journal**
Vol. 10 No. 1 (2023) 75 - 83 ISSN Media Elektronik: 24x43-3640x

End User Development Pada Use Case Point Untuk Peningkatan Estimasi Perangkat Lunak

End User Development on Use Case Points to improve Software Estimation

Puguh Jayadi[1], Alim Citra Aria Bima[2] , Yoga Prisma Yudha[3], Kelik Sussolaikah[4*]
[1,2,3,4]Prodi Teknik Informatika, Fakultas Teknik, Universitas PGRI Madiun
[1]puguh.jayadi@unipma.ac.id, [2]alim.cab@unipma.ac.id, [3]yogaprisma@unipma.ac.id, [4]kelik@unipma.ac.id

Abstract

The effort estimation stage is used as a basis for determining the amount of time and people involved in a software project. There are various methods that are often used, including COCOMO, Function Points, and Use Case Points. However, these methods have drawbacks such as subjectivity in measuring complexity and lack of measurement of the technical and environmental factors of project development. This study seeks to correct these weaknesses by using the Advance Use Case Point (AUCP) method by incorporating elements from End User Development (EUD) into the Use Case Point, thereby enabling end users to develop software more easily. and efficient. The AUCP method takes into account the functional and technical complexity of software (EUD_Technical Factors) and environmental factors (EUD_Environmental Factors) in software development. The results show that the Advance Use Case Point method provides more accurate estimation results with MMRE, MMER, MBRE, MIBRE having a value of 0.01 and MAE and RMSE having a value of 8.88. The results of these values are smaller than the results of several studies using other methods. The smaller the value, the more accurate the estimation results. The AUCP method can assist project managers and end users in optimizing the use of resources and guaranteeing the smooth running of software development projects. Besides that, it can speed up the software development process and reduce development costs because it can be done by end users without the need for the involvement of a professional development team. As well as helping to expand the accessibility of software development to a wider audience and increase end user participation in software development.

Keywords: advance use case point; end user development; technical factors; environmental factors

Abstrak

Tahap estimasi upaya (*Effort Estimation*) digunakan sebagai dasar dalam menentukan jumlah waktu dan orang yang terlibat pada suatu proyek perangkat lunak. Terdapat berbagai metode yang sering dipakai antara lain *COCOMO*, *Function Point*, dan *Use Case Point*. Namun, metode-metode ini memiliki kelemahan seperti adanya subyektifitas dalam mengukur kerumitan dan kurangnya pengukuran pada faktor teknis dan lingkungan pengembangan proyek. Penelitian ini berusaha untuk memperbaiki kelemahan tersebut dengan menggunakan metode *Advance Use Case Point* (AUCP) dengan dengan memasukkan unsur dari *End User Development (EUD)* ke dalam *Use Case Point*, dengan demikian memungkinkan pengguna akhir (*end user*) untuk mengembangkan perangkat lunak dengan lebih mudah dan efisien. Metode AUCP memperhitungkan kerumitan fungsional dan teknis perangkat lunak (*EUD_Technical Factors*) dan faktor lingkungan (*EUD_Environmental Factors*) di pengembangan perangkat lunak. Hasil penelitian menunjukkan bahwa metode *Advance Use Case Point* memberikan hasil estimasi yang lebih akurat dengan MMRE, MMER, MBRE, MIBRE bernilai 0,01 dan MAE serta RMSE bernilai 8,88. Hasil nilai tersebut lebih kecil dari pada hasil beberapa penelitian menggunakan metode lainnya. Nilai yang semakin kecil semakin menunjukan hasil akurasi estimasinya. Metode AUCP dapat membantu manajer proyek dan pengguna akhir dalam mengoptimalkan penggunaan sumber daya dan menjamin kelancaran proyek pengembangan perangkat lunak. Selain itu dapat mempercepat proses pengembangan perangkat lunak dan mengurangi biaya pengembangan karena dapat dilakukan oleh pengguna akhir tanpa perlu keterlibatan tim pengembang profesional. Serta membantu memperluas aksesibilitas pengembangan perangkat lunak ke kalangan yang lebih luas dan meningkatkan partisipasi pengguna akhir dalam pengembangan perangkat lunak.

Kata kunci: advance use case point; end user development; technical factors; environmental factors

Diterima Redaksi: 10-05-2022 | Selesai Revisi: 23-05-2023 | Diterbitkan Online: 01-06-2023

1. Pendahuluan

Estimasi untuk menentukan sumber daya yang diperlukan dalam pengembangan perangkat lunak harus dilakukan sebelum perangkat lunak dikembangkan [1]. Dengan instrumen perhitungan tertentu yang ada dalam estimasi perangkat lunak dapat memberikan dampak untuk perincian kegiatan dan kebutuhan pengembangan perangkat lunak [2], [3]. Yang menjadi dasar untuk menentukan sumber daya yaitu ukuran, biaya, jangka waktu, dan jumlah orang yang terlibat adalah hasil dari estimasi yang dilakukan oleh manajer proyek [4], [5]. Jika sumber daya sudah jelas, maka semua pihak yang terlibat dalam proses pengembangan akan lebih rinci untuk mengetahui apa saja yang harus dilakukan sesuai dengan waktu. Begitu juga dengan pengguna akhir akan lebih mudah untuk mengamati pengerjaan perangkat lunak sesuai dengan biaya yang telah dianggarkan.

Estimasi *effort* atau sumber daya menjadi kunci untuk memastikan bagaimana keberlangsungan proyek perangkat lunak. Jika estimasi dilakukan dengan benar, maka proyek akan berjalan dengan lancar dan berhasil. Namun, jika estimasi biaya tidak akurat, maka proyek akan berakhir dengan kegagalan [3]. Ada beberapa metode dan teknik telah dikembangkan untuk melakukan estimasi *effort* perangkat lunak, seperti COCOMO II yang memakai *Tabel Scale Driver, UFP* dan *Table Effort Mulplier* [1], [6], *Case Based Reasoning* yang menganggap banyak proyek perangkat lunak memiliki jumlah effort yang sama [7], [8], *Analogy Based Estimation* (ABE) yang memakai pendekatan regresi berganda yang dikombinasikan dengan hasil penilaian dari para ahli [3], *Function Point* yang memperhitungkan fungsionalitas awal perangkat lunak serta rincian panjang baris kode yang digunakan [9], [10]. Namun, meskipun telah dilakukan banyak penelitian, masih ada kecenderungan subjektivitas dan ketidakpastian karena kurangnya standar perhitungan yang pasti [11].

User Development atau pengembangan oleh pengguna akhir adalah konsep di mana pengguna akhir dapat mengembangkan aplikasi atau perangkat lunak yang mereka butuhkan tanpa bantuan dari programmer atau ahli TI [12]. Pengembangan oleh pengguna akhir terutama dilakukan oleh pengguna yang memiliki kebutuhan yang spesifik dan unik yang tidak dapat dipenuhi oleh perangkat lunak yang sudah ada di pasar [13]. Metode pengembangan ini dapat mengurangi biaya pengembangan dan waktu pengembangan yang dibutuhkan, karena pengguna dapat mengembangkan perangkat lunak yang dibutuhkan secara langsung tanpa melalui pihak ketiga [14]. Oleh karena itu, metode pengembangan ini dapat dianggap sebagai solusi alternatif dalam pengembangan perangkat lunak, terutama untuk kebutuhan pengguna yang spesifik dan unik.

Dalam penelitian ini, *End User Development* (EUD) diterapkan pada metode estimasi pengembangan perangkat lunak *Advance Use Case Point* (AUCP) untuk meningkatkan akurasi estimasi pengembangan perangkat lunak. Dalam pengembangan perangkat lunak, estimasi biaya dan waktu yang akurat sangat penting untuk menentukan kesuksesan proyek. Metode AUCP pada penelitian ini merupakan pengembangan dari *Use Case Point* yang lebih menitikberatkan peranan pengguna akhir yang pasca pengembangan aplikasi berdasarkan Use Case perangkat lunak [13]–[16]. Obyek penelitian ini pada pengembangan sistem informasi Perkumpulan Pendidikan Islam Usia Dini (PIAUD) sama yang dilakukan oleh Jayadi pada tahun 2022 dengan nilai estimasi 1241 jam [17]. Hasil akhir perhitungan AUCP memberikan nilai 1236 jam, yang lebih mendekati dengan nilai aktual yaitu 1228 jam. Hasil evaluasi dengan MMRE, MMER, MBRE, MIBRE mendapatkan nilai 0,01 dan MAE serta RMSE bernilai 8,88. Hal ini menunjukkan bahwa EUD dapat meningkatkan akurasi estimasi pengembangan perangkat lunak.

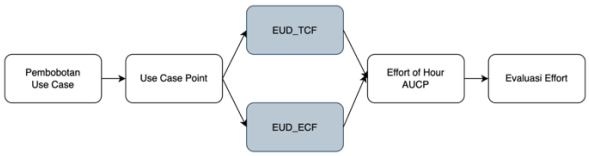
EUD pada metode AUCP dapat menjadi alternatif yang menarik untuk dikembangkan. Diharapkan membantu meningkatkan akurasi estimasi biaya, waktu, dan orang yang dilibatkan untuk pengembangan perangkat lunak. Selain itu, pengembangan oleh pengguna akhir juga dapat membantu mengurangi biaya pengembangan dan waktu yang dibutuhkan, karena pengguna dapat mengembangkan perangkat lunak yang dibutuhkan secara langsung tanpa melalui pihak ketiga. Oleh karena itu, EUD dapat menjadi solusi alternatif yang menarik dalam pengembangan perangkat lunak, terutama untuk kebutuhan pengguna yang spesifik dan unik.

2. Metode Penelitian

Gambar 1 berisi beberapa langkah yang dilakukan dalam penelitian ini. Langkah pertama adalah pemberian bobot *Use Case* diagram yang digunakan. Data proyek yang dipakai adalah dari proyek sistem manajemen dan portal Pendidikan Islam Anak Usia Dini (PIAUD). Semua data yang digunakan dalam pembobotan di penelitian ini didasarkan dari hasil wawancara kepada semua pihak yang terlibat dalam pengembangan PIAUD baik dari *programmer* serta manager proyek. Dari hasil pembobotan setiap *Use Case* dihitung dengan *Use Case Point*. Hasil dari *Use Case Point* sebenarnya sudah bisa digunakan untuk menghitung *Effort of Hour* namun pada penelitian ini menambahkan kontribusi faktor teknis (EUD_TCF) dan lingkungan (EUD_ECF) dari pengguna akhir (*End User Development*). Setelah kombinasi antara EUD_TCF dan EUD_ECF memberikan nilai *Effort of Hour* dari metode usulan *Advance Use Case Point* (AUCP). Setelah mendapatkan nilai *effort* kemudian dievaluasi

DOI:https://doi.org/10.38204/tematik.v10i1.1289

untuk mengukur akurasi dibandingkan nilai *effort* aktual/asli dari pengembangan proyek.



Gambar 1. Tahap Penelitian

2.1. End User Development (EUD)

End User Development (EUD) merujuk pada pengembangan perangkat lunak oleh pengguna akhir (*end user*), bukan oleh pengembang perangkat lunak profesional atau pengembang aslinya [12]. EUD dapat dianggap sebagai salah satu faktor yang memengaruhi estimasi proyek perangkat lunak [13]. EUD dapat terjadi ketika pengguna akhir membuat perangkat lunak sendiri atau mengubah perangkat lunak yang sudah ada untuk memenuhi kebutuhan yang spesifik dan lebih tepat. EUD dapat meningkatkan produktivitas dan efisiensi pengguna akhir, tetapi juga dapat menyebabkan masalah keamanan dan stabilitas perangkat lunak jika tidak dilakukan dengan benar [12].

Dalam konteks penelitian ini, EUD mungkin mempengaruhi estimasi proyek perangkat lunak karena EUD memiliki lebih banyak ketidakpastian dan kompleksitas yang terkait dengan kebutuhan pengguna akhir yang beragam. Oleh karena itu, penelitian ini mencoba untuk mengkolaborasikan metode estimasi proyek perangkat lunak yang mungkin lebih cocok untuk proyek yang melibatkan EUD [13].

Di lain sisi, pengguna akhir dapat memasukkan persyaratan bisnis dan estimasi biaya ke dalam *Use Case Point* (UCP). Hal ini memungkinkan pengguna akhir untuk memainkan peran yang lebih aktif dalam pengembangan perangkat lunak dan memberikan kontribusi yang lebih signifikan dalam estimasi biaya dan pemilihan fitur yang tepat. Dengan memasukkan faktor EUD dalam melakukan estimasi perangkat lunak diharapkan pengembangan perangkat lunak akan menjadi lebih efisien, akurat, dan efektif dalam memenuhi kebutuhan pengguna akhir. Hal tersebut juga dapat membantu mengurangi biaya pengembangan perangkat lunak dan meningkatkan kepuasan pengguna akhir [14].

2.2. EUD pada Use Case Point

Gustav Karner pada tahun 1993 pertama kali memperkenalkan *Use Case Points* (UCP) sebagai perhitungan estimasi perangkat lunak [18]. *Use Case* dipakai untuk menggambarkan kondisi atau fungsional dari sebuah sistem [17]. Dalam UCP, semua *Use Case* diberikan nilai bobot sesuai dengan seperangkat rumus untuk mengetahui *effort* yang kemudian bisa untuk merincikan biaya atau waktu yang dibutuhkan dalam mengembangkan perangkat lunak [19].

Pada penelitian sebelumnya disebutkan penggunaan UCP ditemukan beberapa kelemahan yang didapatkan seperti adanya kurang detail dan ketidakjelasan pada penentuan bobot serta klasifikasi dari *Use Case* [17], [19], [20]. Untuk mengatasi kelemahan dari UCP maka dikolaborasikan dengan EUD yang merupakan suatu pendekatan dengan menguraikan perhitungan dari faktor kompleksitas teknis serta lingkungan [21].

Selain itu berdasarkan pemahaman yang mendalam tentang kebutuhan dan persyaratan pengguna akhir, yang dapat membantu dalam memfasilitasi proses EUD untuk mengembangkan proyek sesuai dengan spesifikasi yang diinginkan [14]. Metode ini dirancang untuk memperhitungkan berbagai faktor, termasuk kompleksitas fungsional, efisiensi sistem, dan keandalan sistem untuk menghasilkan perkiraan biaya yang lebih akurat dalam pengembangan perangkat lunak [22]**.**

2.3. Unadjusted Use Case Weight (UUCW)

Dalam *Use Case* (UC) terdapat aktor yang memiliki tingkat kompleksitas (CA) yang didasarkan pada transaksi data mapun informasi yang dibutuhkan maupun diberikan dari Use Case. Untuk perhitungan UUCW, semua UC diidentifikasi dan diklasifikasikan serta dihitung sesuai dengan jumlah transaksi yang ada pada UC tersebut. Detail UUCW tersaji pada Tabel 1 dan untuk menghitung total UUCW menggunakan persamaan 1.

Tabel 1. Rincian UUCW

CA	Transaksi	Bobot
Simple (Sederhana)	1 – 3	5
Average (Rata-rata)	4 – 7	10
Complex (Komplek)	> 7	15

$$\mathrm{UUCW} = \sum_{i=1}^{n} \mathrm{CA_i} \quad (1)$$

2.4. Unadjusted Actor Weight (UAW)

Total Unadjusted Actor Weights (UAW) yang diperoleh dari penghitungan banyaknya aktor sesuai tipe tingkat kompleksitas dikalikan dengan bobot. Detail untuk bobot aktor terdapat pada Tabel 2. Total Unadjusted Actor Weight (UAW) dihitung dengan persamaan 2.

Tabel 2.Rincian UAW

AC	Deskripsi	Bobot
Simple (Sederhana)	Dengan API atau Command Prompt	1
Average (Rata-rata)	Dengan Protokol, TCP/IP, HTP	2
Complex (Komplek)	Dengan GUI atau Halaman Web	3

$$\mathrm{UAW} = \sum_{i=1}^{n} \mathrm{AC_i} \quad (2)$$

2.5. Unadjusted Use Case Point (UUCP)

UUCP diperoleh dari penjumlahan *Unadjusted Use Case Weights* (UUCW) dengan *Unadjusted Actor*

DOI:https://doi.org/10.38204/tematik.v10i1.1289

Weights (UAW). Rumus UUCP dihitung dengan persamaan 3.

$$\text{UUCP} = \text{UUCW} + \text{UAW} \quad (3)$$

2.6. Technical Complexity Factor (TCF)

TCF dihitung dari 13 faktor teknis yang memiliki pengaruh besar pada kinerja tim [20]. Hal tersebut kemudian disesuaikan dengan faktor teknis yang menggambarkan besarnya kompleksitas teknis yang terlibat dalam pengembangan dan implementasi sistem [18]. Detail TCF terdapat pada Tabel 3. Rumus total TCF dihitung dengan persamaan 4. Skala yang digunakan untuk mengisi TCF adalah antara 0-5 [14].

Tabel 3. Rincian TCF

Ti	Factors	Deskripsi	Bobot
Tl	Distributed System Required	Proses terdistribusi dengan sistem atau perangkat lain	2
T2	Response Time Is Important	Waktu respons pada sistem dibutuhkan	1
T3	End User Efficiency	Dapat meningkatkan efisiensi pengguna	1
T4	Complex Internal Processing	Sistem terdapat proses yang rumit	1
T5	Reusable Code Must Be A Focus	Kode digunakan kembali pada module lain	1
T6	Installation Easy	Instalasi yang mudah	0,5
T7	Usability, Operational ease	Kemudahan penggunaan pada sistem	0,5
T8	Cross-Platform Support	Sistem dapat multi platform	2
T9	Easy To Change	Sistem mudah untuk diubah sesuai kebutuhan	1
T10	Highly Concurrent	Sistem terdapat proses yang berkaitan	1
T11	Custom Security	Sistem terdapat fitur keamanan	1
T12	Dependence On Third Part Code	Ketergantungan pada kode atau aplikasi pihak ketiga	1
T13	User Training	Mudah untuk pelatihan kepada pengguna	1

$$\text{TCF} = 0.6 + (0.01 * \sum_{i=1}^{n} \text{Ti}_i) \quad (4)$$

2.7. Environmental Complexity Factor (ECF)

ECF dihitung dari 8 faktor lingkungan yang memiliki pengaruh besar pada perkiraan seberapa efisien proyek yang dikembangkan tim [18]. Hal tersebut berkaitan dengan besarnya kompleksitas lingkungan yang digunakan dalam pengembangan dan implementasi sistem [18]. Detail ECF terdapat pada Tabel 4. Rumus total ECF dihitung dengan persamaan 5. Skala yang digunakan untuk mengisi ECF adalah antara 0-5 [14].

$$\text{ECF} = 1.4 + (-0.03 * \sum_{i=1}^{n} \text{Fi}_i) \quad (5)$$

Tabel 4.Rincian ECF

Ti	Factors	Deskripsi	Bobot
F1	Familiar with Objectory	Tim pengembang terbiasa dengan metode Objectory	1,5
F2	Part time workers	Pekerja paruh waktu	-1
F3	Analyst capability	Memiliki kemampuan analisis	0,5
F4	Application experience	Berpengalaman mengembangkan proyek	0,5
F5	Object oriented experience	Berpengalaman menggunakan OOP	1
F6	Motivation	Memiliki motivasi tinggi	1
F7	Difficult programming language	Menggunakan bahasa pemrograman yang sulit	-1
F8	Stable requirements	Kebutuhan pengembangan yang stabil	2

2.8. Use Case Point (UCP)

UCP diperoleh dari perkalian UUCP dengan *TCF* dan ECF. Rumus UCP dihitung dengan persamaan 6.

$$\text{UCP} = \text{UUCP} * \text{TCF} * \text{ECF} \quad (6)$$

2.9. End User Development Technical Complexity Factor (EUD_TCF)

EUD_TCF merupakan faktor yang mewaliki dampak penambahan fitur oleh pengguna akhir (*End User*) yang menjadi syarat pada faktor teknis di Use Case proyek yang dikerjakan [11]. Nilai yang digunakan untuk mengisi EUD_TCF adalah 0 atau 1 [14]. Detail bobot EUD_TCF terdapat perbedaan dari beberapa penelitian yang tersaji pada Tabel 5 [13], [14], [22]. Total EUD_TCF dihitung dengan persamaan 7.

$$\text{EUD\_TCF} = 0.6 + (0.01 * \sum_{i=1}^{n} \text{EUD\_Ti}_i) \quad (7)$$

2.10. End User Development Environment Complexity Factor (EUD_ECF)

EUD_ECF merupakan faktor yang mewaliki dampak penambahan fitur oleh pengguna akhir (*End User*) yang menjadi syarat pada faktor lingkungan dimana proyek yang dikerjakan [11]. Nilai yang digunakan untuk mengisi EUD_ECF adalah 0 atau 1 [14]. Detail bobot EUD_EF terdapat perbedaan dari beberapa penelitian yang tersaji pada Tabel 6 [13], [14], [22]. Total EUD_EF dihitung dengan persamaan 8.

$$\text{UD\_ECF} = 1.4 + (0.03 * \sum_{i=1}^{n} \text{EUD\_Ei}_i) \quad (8)$$

2.11. Perhitungan Advance Use Case Point (AUCP)

Nilai dari AUCP dihitung dari perkalian UCP dengan nilai EUD_TCF dan EUD_ECF. AUCP dihitung dengan persamaan 9.

$$\text{AUCP} = \text{UCP} * \text{EUD\_TCF} * \text{EUD\_ECF} \quad (9)$$

2.12. Hour of Effort

Setelah mendapatkan AUCP, nilai tersebut bisa difungsikan untuk menghitung effort (dalam satuan jam) caranya mengalikan nilai AUCP dengan besaran

DOI:https://doi.org/10.38204/tematik.v10i1.1289

nilai produktifitas.

Tabel 5. Rincian EUD_TCF

EUD_Ti	Factors	Deskripsi	Bobot [13]	Bobot [14]	Bobot [22]
Tl	Creating throwaway codes	Membuat kode sementara sebelum kode yang sebenarnya dipakai di aplikasi	0,5	0,5	1
T2	Creating reusable codes	Membuat kode yang bisa digunakan kembali dan berkali-kali	1,2	1	1,2
T3	Sharing reusable code	Berbagi kode yang dapat digunakan ulang	1,4	1	1,4
T4	Easy understandable codes	Mudah dalam memahami kode	1	1	1,2
T5	Security features in codes for more control by end users	Memperhatikan faktor keamanan agar dapat memberikan kontrol yang lebih besar bagi pengguna akhir dalam mengelola keamanan aplikasi	1,3	1,5	1,5
T6	Authentication features	Fitur keamanan dengan autentikasi	1,12	1,5	1,12
T7	Inbuilt feedback about the correctness	Semakin sulit fitur semakin sulit mendapat feedback yang benar tentang aplikasi	1,3	1,5	1,3
T8	Testable codes	Kode dapat dites/diuji coba	1,2	0,5	1,2
T9	Tools for analyzing by debugging	Mengacu pada alat atau software yang digunakan untuk menganalisis dan memperbaiki kode	1,4	1,5	1,4
T10	Error detection tools	Berkaitan dengan alat atau software yang digunakan untuk mendeteksi kesalahan atau bug pada aplikasi	1,2	1,5	1,5
T11	Online help availability	Ketersediaan bantuan online yang dapat membantu pengguna aplikasi dalam memecahkan masalah	1,3	1,5	1,5
T12	Self — efficiency (High sense of control over the environment)	Kemampuan pengguna untuk mengontrol lingkungan aplikasi dengan memastikan pengguna merasa nyaman dan dapat menggunakan aplikasi dengan efektif	1,11	1	1,11
T13	Perceived ease of use: Apart from extrinsic motivation intrinsic motivation (enjoyment) should be present.	Persepsi pengguna tentang tingkat kemudahan dan kepuasan dalam penggunaan aplikasi	1,2	1,5	1,2
T14	Perceived usefulness	Tingkat kegunaan dan manfaat yang dirasakan oleh pengguna akhir terhadap sebuah aplikasi.	1	1	1
T15	Flexible codes	Kode yang fleksibel sesuai kebutuhan	1,2	0,5	1,2
T16	Scalability features	Semakin kompleks dan banyaknya fitur yang dibutuhkan untuk membuat aplikasi yang terukur, maka semakin sulit pula pengembangan perangkat lunak.	1,25	0,5	1,25
T17	Ease of Maintenance	Kemudahan dalam pemeliharaan	1,2	1	1,2

Tabel 6. Rincian EUD_ECF

EUD_Ei	Factors	Deskripsi	Bobot [13]	Bobot [14]	Bobot [22]
E1	Content Level of EUP	Mengacu pada tingkat kompleksitas dan kekayaan konten yang terkait dengan penggunaan aplikasi oleh pengguna akhir.	1,4	1,5	1,4
E2	End User Computing Capability	Mengacu pada kemampuan pengguna akhir dalam menggunakan teknologi dan perangkat lunak.	0,25	1	0,75
E3	Ease of Use & Feedback	Mengacu pada tingkat kemudahan penggunaan aplikasi dan kemampuan untuk memberikan umpan balik kepada pengguna.	1,2	1	1,5
E4	Inbuilt System Assistance for EUP	Berkaitan dengan fitur sistem bantuan yang disediakan dalam aplikasi untuk membantu pengguna akhir dalam memecahkan masalah.	1,25	1	1,25
E5	Training & learning Time Constraint for end user	Mengacu pada waktu dan sumber daya yang tersedia untuk melatih pengguna akhir dalam menggunakan aplikasi.	1,12	1,5	1,12
E6	Reliability of End User Code	Berkaitan dengan keandalan dan keamanan kode yang dibuat oleh pengguna akhir	1,2	1	1,2
E7	End User Storage Constraint	Mengacu pada batasan penyimpanan yang tersedia untuk pengguna akhir pada perangkat mereka.	1,02	1	1,02
E8	Risk Factors	Berkaitan dengan faktor risiko yang terkait dengan penggunaan aplikasi oleh pengguna akhir, seperti faktor keamanan dan privasi.	1,12	1,5	1,5

Untuk nilai produktitas yang umum digunakan adalah 20 [11], [13], [14], [18], [22]. Perhitungan effort didapatkan dengan persamaan 10.

$$\text{Effort} = \text{AUCP} * 20 \quad (10)$$

2.13. Evaluasi Effort

Untuk mengevaluasi nilai effort, terdapat beberapa metode yang umum digunakan seperti *Mean Magnitude of Relative Error* (MMRE) yaitu ukuran rata-rata

DOI:https://doi.org/10.38204/tematik.v10i1.1289

perbedaan antara effort aktual dan effort perkiraan yang dihitung sebagai persentase dari effort aktual [23]–[25]. Nilai MMRE yang baik adalah sekitar 10% atau kurang. Artinya, jika MMRE dari sebuah proyek adalah 10%, maka estimasi biaya proyek tersebut cenderung memiliki perbedaan sekitar 10% dengan biaya aktual yang sebenarnya. Nilai MMRE didapatkan dengan rumus pada persamaan 11.

$$\text{MMRE} = \frac{|\text{Act Effort} - \text{Pred Effort}|}{\text{Act Effort}} \quad (11)$$

Mean Magnitude of Error Relative (MMER) merupakan metrik evaluasi yang digunakan untuk mengukur kesalahan rata-rata dalam perkiraan effort perangkat lunak [23]–[25]. MMRE adalah ukuran rata-rata perbedaan antara effort aktual dan effort perkiraan yang dihitung sebagai persentase dari effort aktual. Nilai MMER yang didapat dengan rumus pada persamaan 12.

$$\text{MMER} = \frac{|\text{Act Effort} - \text{Pred Effort}|}{\text{Pred Effort}} \quad (12)$$

Mean Balanced Relative Error (MBRE) merupakan metrik evaluasi yang digunakan untuk mengukur rata-rata kesalahan relatif hingga minimum antara *effort* aktual dan prediksi [23], [26]. Nilai MBRE yang didapat dengan rumus pada persamaan 13.

$$\text{MBRE} = \frac{1}{n}\sum_{i=1}^{n} \frac{|\text{Act Effort} - \text{Pred Effort}|}{\min(\text{Act Effort}, \text{Pred Effort})} \quad (13)$$

Mean Inverse Balanced Relative Error (MIBRE) merupakan metrik evaluasi yang digunakan untuk mengukur rata-rata kesalahan relatif hingga maksimum antara *effort* aktual dan prediksi [23], [26]. Nilai MIBRE yang didapat dengan rumus pada persamaan 14.

$$\text{MIBRE} = \frac{1}{n}\sum_{i=1}^{n} \frac{|\text{Act Effort} - \text{Pred Effort}|}{\max(\text{Act Effort}, \text{Pred Effort})} \quad (14)$$

Mean Absolute Error (MAE) merupakan metrik evaluasi yang digunakan untuk mengukur rata-rata kesalahan antara *effort* aktual dan prediksi [23], [26]. Nilai MAE yang didapat dengan rumus pada persamaan 15.

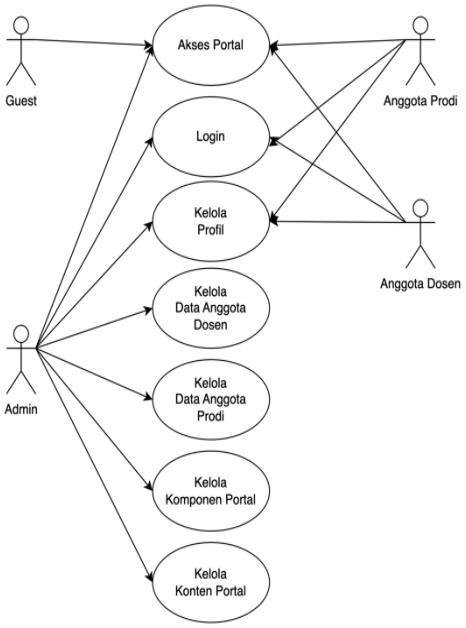
$$\text{MAE} = \frac{\sum_{i=1}^{n} |\text{Act Effort} - \text{Pred Effort}|}{n} \quad (15)$$

Root Mean Squared Error (RMSE) merupakan ukuran rata-rata dari selisih antara nilai prediksi yang dihasilkan oleh model dan nilai aktual, yang dihitung sebagai akar dari nilai kuadrat dari selisih tersebut [23], [26]. Nilai RMSE yang didapat dengan rumus pada persamaan 16.

$$\text{RMSE} = \sqrt{|\text{Act Effort} - \text{Pred Effort}|^2} \quad (16)$$

3. Hasil dan Pembahasan

Gambar 2 menyajikan *Use Case* yang digunakan dalam pengembangan proyek sistem PIAUD. Terdapat 3 aktor yang berkaitan dengan 7 entitas pokok dalam sistem (kecuali *Guest*). Sistem PIAUD digunakan untuk mengelola data anggota dosen serta program studi Pendidikan Islam Anak Usia Dini secara nasional. Oleh karena itu 3 aktor yang dimaksud adalah Admin, Anggota Dosen, Anggota Program Studi. Selain itu sistem PIAUD juga bisa diakses secara umum (*public*) sebagai tamu (*Guest*) namun hanya menampilkan informasi terbatas seperti informasi pendaftaran tanpa ada akses untuk melihat dan mengubah data pokok didalam sistem.



Gambar 2. Use Case Diagram PIAUD

Setiap aktor dan entitas yang ada di dalam Use Case diberikan bobot berdasarkan dari hasil wawancara ke *programmer* dan manager proyek. Setelah proses pemberian bobot maka dapat dilakukan perhitungan waktu untuk pengembangan proyek tersebut sesuai dengan langkah-langkah yang sudah dijelaskan.

Langkah pertama adalah menghitung nilai UUCW. Perhitungan didasarkan pada jumlah transaksi yang dibutuhkan dan tanggapan yang dilakukan oleh aktor di dalam *Use Case* [4], [21]. Jumlah aktor yang terlibat (Admin, Anggota Dosen, Anggota Program Studi) dan masing-masing aktor memiliki jumlah transaksi diatas 7. Sehingga nilai akhir UUCW yang dihitung dengan persamaan 1 adalah 45.

Langkah kedua adalah menghitung UAW. Perhitungan tersebut berdasarkan jumlah aktor yang berhubungan dengan sistem melalui GUI atau Halaman Web. Sehingga nilai akhir UAW yang dihitung dengan persamaan 2 adalah jumlah aktor x bobot = 9.

Langkah ketiga adalah menghitung UUCP diperoleh dari penjumlahan UUCW dan UAW [14], [18]. Hasil

DOI:https://doi.org/10.38204/tematik.v10i1.1289

perhitungan UUCP menggunakan persamaan 3 mendapatkan nilai 54.

Langkah keempat adalah menghitung TCF. Nilai dari setiap faktor teknis berhubungan dengan kompleksitas teknis yang dihadapi oleh tim dalam pengembangan perangkat lunak [13], [22]. Rincian penilaian TCF terdapat pada tabel 7 diisi dengan nilai skala antara 1 -5 kemudian dihitung dengan persamaan 4 mendapatkan nilai TCF sebesar 1.09.

Tabel 7. Hasil Penilaian TCF

Faktor	Nilai	Nilai x Bobot
T1	1	2
T2	5	5
T3	5	5
T4	3	3
T5	4	4
T6	4	2
T7	4	2
T8	4	8
T9	5	5
T10	3	3
T11	3	3
T12	2	2
T13	5	5
Total		49

Langkah kelima adalah menghitung ECF. Nilai dari setiap faktor lingkungan berhubungan dengan kompleksitas kondisi lingkungan yang ada pada tim dalam pengembangan perangkat lunak [13], [27]. Rincian penilaian ECF terdapat pada tabel 8 diisi dengan nilai skala antara 1 -5 kemudian dihitung dengan persamaan 5 mendapatkan nilai ECF sebesar 0.95.

Tabel 8. Hasil Penilaian ECF

Faktor	Nilai	Nilai x Bobot
F1	3	4,5
F2	5	-5
F3	3	1,5
F4	4	2
F5	4	4
F6	4	4
F7	2	-2
F8	3	6
Total		15

Langkah keenam adalah penghitungan UCP. Nilai UCP didapatkan dari perkalian nilai UUCP dengan perkalian TCF serta ECF. Hasil perhitungan UCP dengan persamaan 6 mendapatkan UCP sebesar 55,917. Langkah ketujuh adalah menghitung EUD_TCF. Nilai ini merepresentasikan hubungan faktor teknis dari pengguna akhir terhadap keberlangsungan pengembangan perangkat lunak berikutnya [13], [27]. Nilai yang digunakan adalah faktor 0 bermakna tidak berpengaruh atau 1 bermakna ada berpengaruh [13]. Rincian penilaian EUD_TCF terdapat pada tabel 9 kemudian dihitung dengan persamaan 7 mendapatkan nilai EUD_TCF dengan bobot dari masing-masing referensi.

Tabel 9. Hasil Penilaian EUD_TCF

Faktor	Nilai	Nilai x Bobot [13]	Nilai x Bobot [14]	Nilai x Bobot [22]
T1	1	0,5	0,5	1
T2	1	1,2	1	1,2
T3	1	1,4	1	1,4
T4	1	1	1	1,2
T5	0	0	0	0
T6	1	1,12	1,5	1,12
T7	0	0	0	0
T8	1	1,2	0,5	1,2
T9	0	0	0	0
T10	0	0	0	0
T11	1	1,3	1,5	1,5
T12	0	0	0	0
T13	0	0	0	0
T14	1	1	1	1
T15	1	1,2	0,5	1,2
T16	1	1,25	0,5	1,25
T17	1	1,2	1	1,2
Total		12,37	10	13,27
Total EUD_TCF		0,72	0,70	0,73

Langkah kedelapan adalah menghitung EUD_ECF. Nilai ini hubungan dengan faktor lingkungan dari pengguna akhir terhadap keberlangsungan pengembangan perangkat lunak berikutnya [13], [22]. Nilai yang digunakan adalah faktor 0 bermakna tidak berpengaruh atau 1 bermakna ada berpengaruh [13]. Hasil perhitungan EUD_ECF tersaji pada tabel 10 dan dihitung dengan persamaan 8 mendapatkan nilai EUD_ECF masing-masing sesuai dengan bobot yang ada pada 3 referensi.

Tabel 10. Hasil Penilaian EUD_ECF

Faktor	Nilai	Nilai x Bobot [13]	Nilai x Bobot [14]	Nilai x Bobot [22]
E1	0	0	0	0
E2	0	0	0	0
E3	1	1,2	1	1,2
E4	1	1,25	1	1,25
E5	1	1,12	1,5	1,12
E6	1	1,2	1	1,2
E7	0	0	0	0
E8	1	1,12	1,5	1,12
Total		5,89	6	5,89
Total EUD_TCF		1,58	1,58	1,58

Langkah kesembilan adalah menghitung nilai AUCP. Hasil perhitungan AUCP terdapat pada tabel 11 yang dihitung menggunakan persamaan 9.

Tabel 11. Hasil Penilaian AUCP

AUCP dengan bobot [13]	AUCP dengan bobot [14]	AUCP dengan bobot [22]
63,80	61,84	64,60

Langkah kesepuluh adalah menghitung nilai *Hours of Effort* yaitu nilai AUCP dibagi dengan nilai produktifitas (20) sesuai dengan persamaan 10 [4], [21] dan hasilnya terdapat pada tabel 12.

Tabel 12. Hasil *Hours of Effort*

AUCP dengan bobot [13]	AUCP dengan bobot [14]	AUCP dengan bobot [22]
1276,09	1236,88	1291,96

DOI:https://doi.org/10.38204/tematik.v10i1.1289

Langkah terakhir adalah menghitung evaluasi *effort* aktual yaitu 1228 dengan nilai hasil estimasi AUCP menggunakan beberapa rumus seperti MMER, MMRE, MBRE, MIBRE, MAE dan RMSE sesuai dengan persamaan 11, 12, 13, 14, 15, dan 16. Hasil dari masing-masing perhitungan evaluasi *effort* terdapat pada tabel 13.

Tabel 13. Hasil *Hours of Effort*

Evaluasi	*Hours of Effort UCP*	*Hours of Effort* AUCP dengan bobot [13]	*Hours of Effort* dengan bobot [14]	*Hours of Effort* dengan bobot [22]
	1118,34	1276,09	1236,88	1291,96
MMER	0,10	0,04	0,01	0,05
MMRE	0,09	0,04	0,01	0,05
MAE	109,66	48,09	8,88	63,96
MBRE	0,10	0,04	0,01	0,05
MIBRE	0,09	0,04	0,01	0,05
RMSE	109,66	48,09	8,88	63,96

4. Kesimpulan

Sesuai dengan hasil penelitian sebelumnya, pengguna (*End User*) memiliki peran penting dalam pengembangan perangkat lunak. Pengguna memiliki kemampuan untuk mengembangkan perangkat lunak mereka sendiri, yang dikenal sebagai *End User Development* (EUD). EUD melibatkan pengguna dalam pengembangan perangkat lunak dengan memberikan masukan dan perubahan pada perangkat lunak yang sedang dikembangkan. Hal ini membantu meningkatkan estimasi pengembangan perangkat lunak dengan mengambil kualitas teknis EUD_TCF dan lingkungan EUD_ECF sebagai faktor tambahan dalam perhitungan estimasi. Pembobotan serta perhitungan estimasi dalam pengembangan proyek sistem PIAUD dengan metode *Advance Use Case Point* menghasilkan nilai akurasi hampir sama dengan *effort of hour* sebenarnya.

Pada evaluasi akurasi dengan bobot EUD_TCF dan EUD_ECF dari penelitian Srivastava pada tahun 2017 [13] memberikan nilai yang lebih mendekati yaitu 1236,88 dari 1228 nilai aktual. Hasil tersebut berbeda dengan UCP tanpa menggunakan EUD yang menghasilkan nilai 1118,34. Dari tersebut bisa disimpulkan bahwa *End User Development* (EUD) pada *Use Case Point* (UCP) meningkatkan *effort* estimasi mendekati dengan *effort* aktual dikarenakan adanya kontribuasi faktor lingkungan EUD_TCF dan EUD_ECF.[1]

Daftar Rujukan

[1] L. Indriyani, "Perbandingan Metode Cocomo II Dan Metode Analogy Untuk Estimasi Effort Pengembangan Software," *J. Tek. Komput. AMIK BSI*, vol. 6, no. 2, pp. 174–180, 2020.

[2] A. Y. P. Putri, "Modifikasi Metode Function Point Dengan Menambahkan Kompleksitas Proses Bisnis Pada General System Characteristics Untuk Estimasi Biaya Perangkat Lunak," Institut Teknologi Sepuluh Nopember, 2018.

[3] A. Kaushik, P. Kaur, N. Choudhary, and Priyanka, "Stacking regularization in analogy-based software effort estimation," *Soft Comput.*, vol. 26, no. 3, pp. 1197–1216, 2022.

[4] F. Ristanti, A. Dwi Herlambang, and M. C. Saputra, "Evaluasi Biaya Pengembangan Perangkat Lunak Dengan Menggunakan Metode Extended Use Case Point Dan Use Case Size Point," 2018.

[5] N. Marcheta, "Effort Estimation Modeling Of E-Government Application Development Using Function Points Based On TOR And SRS Document," *J. Inf. Technol. Its Util.*, vol. 3, no. 1, p. 5, 2020.

[6] M. A. A. K. Alabajee, N. A. AL-Saati, and T. R. Alreffaee, "Parameter tuning of software effort estimation models using antlion optimization," *Telkomnika (Telecommunication Comput. Electron. Control.*, vol. 19, no. 3, pp. 817–828, Jun. 2021.

[7] F. Uysal and R. Sonmez, "Bootstrap Aggregated Case-Based Reasoning Method for Conceptual Cost Estimation," *Buildings*, vol. 13, no. 3, p. 651, Feb. 2023.

[8] S. Jung, J. H. Pyeon, H. S. Lee, M. Park, I. Yoon, and J. Rho, "Construction cost estimation using a case-based reasoning hybrid genetic algorithm based on local search method," *Sustain.*, vol. 12, no. 19, Oct. 2020.

[9] S. Sariyanti, "Pengembangan Kakas Estimasi Perangkat Lunak Dengan Function Point Dan Use Case Point Untuk Praktikum Rekayasa Perangkat Lunak," *J. Sarj. Tek. Inform.*, vol. 6, no. 2, pp. 1–9, 2018.

[10] S. Sariyanti and Ardiansyah, "Kakas Estimasi Perangkat Lunak Menggunakan Function Point dan Use Case Point untuk Praktikum Rekayasa Perangkat Lunak di Universitas Ahmad Dahlan," *Annu. Res. Semin.*, vol. 3, no. 1, 2017.

[11] M. C. Saputra *et al*., "Perbandingan Antara Metode Advance Use Case Point Dan Revised Use Case Point Untuk Evaluasi Biaya Pengembangan Sistem Informasi Reservasi Ruangan," *Jurnan Tek. Inform. Dan Sist. Inform.*, vol. 7, no. 1, 2020.

[12] B. R. Barricelli, F. Cassano, D. Fogli, and A. Piccinno, "End-user development, end-user programming and end-user software engineering: A systematic mapping study," *J. Syst. Softw.*, vol. 149, pp. 101–137, Mar. 2019.

[13] A. Srivastava, S. K. Singh, and S. Q. Abbas, "Evaluation of Software Project Estimation Methodology: AUCP," *Int. J. Control Theory Applocations*, vol. 10, no. 19, pp. 183–191, 2017.

[14] A. Srivastava, S. S.K, and S. Q. Abbas, "Advancement Of UCP With End User Development Factor: AUCP," *Int. J. Softw. Eng. Appl.*, vol. 6, no. 2, pp. 01–10, Mar. 2015.

[15] A. Srivastava, S. K. Singh, and S. Q. Abbas, "Impact of end user development technical and environmental factors on software cost," *Int. J. Eng. Technol.*, vol. 7, no. 4, pp. 2203–2208, 2018.

[16] A. Kaur and K. Kaur, "Systematic literature review of mobile application development and testing effort estimation," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 34, no. 2, pp. 1–15, Feb. 2022.

[17] P. Jayadi, Juwari, M. Luthfi Azis, and K. Sussolaikah, "Estimasi Pengembangan Perangkat Lunak Dengan Use Case Size Point," *Bull. Inf. Technol.*, vol. 3, no. 4, pp. 332–340, 2022.

[18] G. Karner, "Resource Estimation for Objectory Projects," *Object. Syst. SF AB*, pp. 1–9, 1993.

[19] W. H. N. Putra and A. R. Perdanakusuma, "Estimasi Biaya Proyek Perangkat Lunak Menggunakan Use Case-Based Effort Estimation," *J. Tecnoscienza*, vol. 4, no. 2, pp. 283–300, 2020.

[20] M. Azzeh, A. B. Nassif, and C. L. Martín, "Empirical analysis on productivity prediction and locality for use case points method," *Softw. Qual. J.*, vol. 29, no. 2, pp. 309–336, Jun. 2021.

[21] M. R. Braz and S. R. Vergilio, "Software Effort Estimation Based on Use Cases," in *Proceedings - International Computer Software and Applications Conference*, 2006, vol. 1, pp. 221–228.

[22] A. Srivastava, S. K. Singh, and S. Q. Abbas, "Performance

DOI:https://doi.org/10.38204/tematik.v10i1.1289

measure of the proposed cost estimation model: Advance use case point method," in *Advances in Intelligent Systems and Computing*, 2019, vol. 742, pp. 223–233.

[23] M. Azzeh, A. Bou Nassif, and I. B. Attili, "Predicting software effort from use case points: A systematic review," *Sci. Comput. Program.*, vol. 204, p. 102596, 2021.

[24] V. Van Hai, H. L. T. K. Nhung, Z. Prokopova, R. Silhavy, and P. Silhavy, "Toward Improving the Efficiency of Software Development Effort Estimation via Clustering Analysis," *IEEE Access*, vol. 10, no. June, pp. 83249–83264, 2022.

[25] S. A. Butt, S. Misra, G. Piñeres-Espitia, P. Ariza-Colpas, and M. M. Sharma, "A Cost Estimating Method for Agile Software Development," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2021, vol. 12955 LNCS, pp. 231–245.

[26] K. Qi, A. Hira, E. Venson, and B. W. Boehm, "Calibrating use case points using Bayesian analysis," in *International Symposium on Empirical Software Engineering and Measurement*, 2018.

[27] Y. Mahmood, N. Kama, and A. Azmi, "A systematic review of studies on use case points and expert-based estimation of software development effort," *J. Softw. Evol. Process*, vol. 32, no. 7, Jul. 2020.

DOI:https://doi.org/10.38204/tematik.v10i1.1289