Estimasi Biaya Proyek Perangkat Lunak Menggunakan *Use Case-Based Effort Estimation*

Widhy Hayuhardhika Nugraha Putra[1], Andi Reza Perdanakusuma[2]
Program Studi Sistem Informasi, Universitas Brawijaya[1,2]
Email: widhy@ub.ac.id[1], andireza@ub.ac.id[2]

Abstrak

Estimasi Biaya Perangkat Lunak erat kaitannya dengan proses manajemen biaya proyek perangkat lunak. Proses manajemen biaya ini dilakukan pada fase perencanaan sebelum fase pembangunan perangkat lunak dimulai. Fase ini merupakan fase kritis yang akan menentukan *triple constraint* dari proyek. Salah satu pendekatan dalam estimasi biaya perangkat lunak adalah dengan menghitung upaya atau *Effort* yang diperlukan dalam mengerjakan proyek perangkat lunak (*Software Effort Estimation*). *Use Case* perangkat lunak dapat menjadi salah satu tolok ukur yang akurat dalam menghitung *Software Effort Estimation*. Dalam hal menghitung Software Effort Estimation, perusahaan pelaksana proyek TI tentunya mengharapkan metode yang paling mendekati biaya aktual yang akan dikeluarkan untuk pelaksanaan proyek untuk menghindari *underestimate* atau *overestimate* dalam memperkirakan biaya perangkat lunak. Terdapat beberapa metode perhitungan *Software Effort Estimation* berbasis *Use Case* yaitu UCP, E-UCP dan Re-UCP. Penelitian ini mencoba mengimplementasikan penggunaan tiga metode tersebut dalam memperkirakan biaya proyek perangkat lunak, dan menganalisis ciri khas masing-masing metode dalam menghitung Software Effort Estimation. Dari hasil penelitian yang dilakukan, didapatkan hasil bahwa ketiga metode tersebut dapat secara akurat menghitung *Software Effort Estimation*.

Kata kunci: Estimasi Biaya Perangkat Lunak, Manajemen Proyek, *Use Case Point*

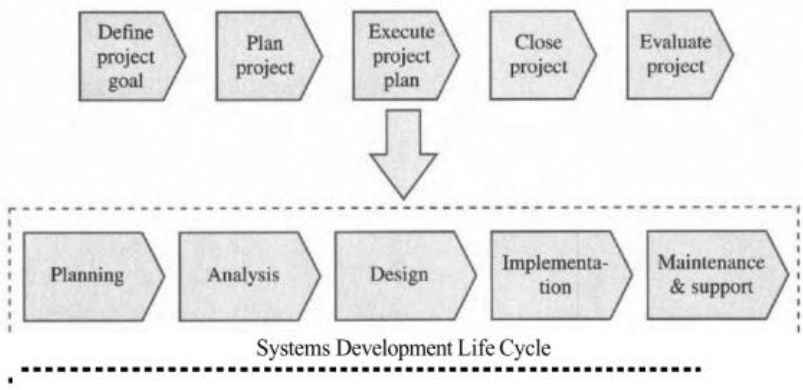
Abstract

Software Cost Estimation is related to the software project cost management process. This cost management process is the part of planning phase before software development phase is executed. This phase is a critical phase that will define the triple constraint of the project. One approach in estimating software costs is to calculate the effort needed in process of software development (Software Effort Estimation). Use Case software can be one of the accurate benchmarks in calculating Software Effort Estimation. In purpose of calculating Software Effort Estimation, companies certainly expect the method that is closest to the actual costs needed for implementation to avoid underestimate or overestimate in estimating software costs. There are several methods of calculating Software Effort Estimation based on Use Cases: UCP, E-UCP and Re-UCP. This study tries to implement the use of these three methods in estimating software project costs, and analyzing the characteristics of each method in calculating Software Effort Estimation. From the results of research conducted, it was found that the three methods can accurately calculate the Software Effort Estimation.

Keywords: *Software Cost Estimation, Project Management, Use Case Point*

A. PENDAHULUAN

Tantangan dalam proyek perangkat lunak saat ini menjadi jauh lebih kompetitif. Produk perangkat lunak yang harus dihasilkan menjadi jauh lebih kompleks, tidak mudah untuk diprediksi dan memiliki lebih banyak tantangan (Nhung, Hoc, & Hai, 2019). Perkembangan teknologi dan kebutuhan bisnis juga turut serta meningkatkan ekspektasi klien proyek akan biaya, waktu dan sumber daya yang diperlukan dalam pengerjaan proyek. Tugas utama seorang manajer proyek adalah untuk merencanakan, mengoranisasikan, mengarahkan dan mengendalikan sumber daya proyek agar dapat memenuhi kebutuhan proyek (Project Management Institute, 2013). Untuk menyelesaikan proyek dengan sukses, manajer proyek perlu memastikan tiga aspek, antara lain lingkup proyek, perkiraan waktu pelaksanaan proyek dan perkiraan biaya proyek dengan seakurat mungkin agar proyek bisa diselesaikan dengan sukses (Schwalbe, 2011).



Gambar 1. Project Life Cycle dan Software Development Life Cycle

(Marchewka, 2003)

Bagian yang penting dalam manajemen proyek adalah perencanaan strategis proyek. Hal ini dapat dilakukan dengan menggunakan metode *Work Breakdown Structure* (WBS), yaitu membuat dokumen yang konsisten dan koheren yang dapat digunakan untuk memandu pelaksanaan proyek dan mengontrol aktivitas proyek (Project Management Institute, 2013). Dengan menggunakan WBS, proyek dapat diidentifikasi, diperkirakan, dijadwalkan dan dibuatkan anggaran sesuai dengan kebutuhan proyek. WBS dapat menghubungkan antara ruang lingkup (*scope*), waktu (*time*), dan biaya (*cost*) sebuah proyek (Schwalbe, 2011).

Perkiraan biaya menjadi salah satu constrain yang penting dalam kesuksesan proyek. Beberapa faktor penentu biaya (*cost driver*) dari sebuah proyek antara lain adalah waktu pelaksanaan proyek dan sumber daya manusia dalam proyek (Boehm, 1981). Oleh karena itu, peran *Software Effort Estimation* (SEE) menjadi sangat penting dalam sebuah proyek perangkat lunak (Nhung, Hoc, & Hai, 2019). SEE akan membantu manajer proyek dalam mengambil keputusan dalam mengelola sumber daya, merencanakan proyek, mengontrol proyek, dan menyelesaikan

proyek dalam waktu, biaya yang tepat (Nhung, Hoc, & Hai, 2019). Telah banyak riset dilakukan sebelumnya untuk mencari metodologi SEE yang akurat, antara lain: (Azzeh & Nassif, 2018), dan (Trendowicz A., 2008).

Namun dalam menghitung SEE, perusahaan pelaksana proyek perangkat lunak seringkali mengalami kesulitan. Perusahaan biasanya berpedoman pada spesifikasi kebutuhan perangkat lunak. Pada kenyataanya spesifikasi kebutuhan perangkat lunak merupakan gambaran awal berupa model perangkat lunak yang akan dibangun dalam proyek seringkali masih tidak jelas. Memperhitungkan SEE menggunakan sebuah model perangkat lunak yang masih belum jelas merupakan hal yang sulit untuk dilakukan. Apalagi dengan model perangkat lunak yang tidak akurat akibat elisitasi kebutuhan yang kurang optimal akan memperburuk perencanaan proyek. Hal ini akan beresiko terhadap kegagalan proyek. Data yang diterbitkan oleh The 2018 Standish Group CHAOS menggambarkan bahwa pada tahun 2018 masih banyak perusahaan yang tidak melakukan perencanaan biaya dengan matang atau jadwal yang tepat, sehingga mengakibatkan pelaksanaan proyek mereka mengalami keterlambatan dan kelebihan biaya (48%-65%) atau gagal menyelesaikan proyek (48%-56%) (The Standish Group, 2018). Dalam laporan tersebut juga disajikan bahwa biaya aktual yang dikeluarkan perusahaan dan waktu yang diperlukan dalam menyelesaikan proyek meleset jauh dari perencanaan yang mereka lakukan.

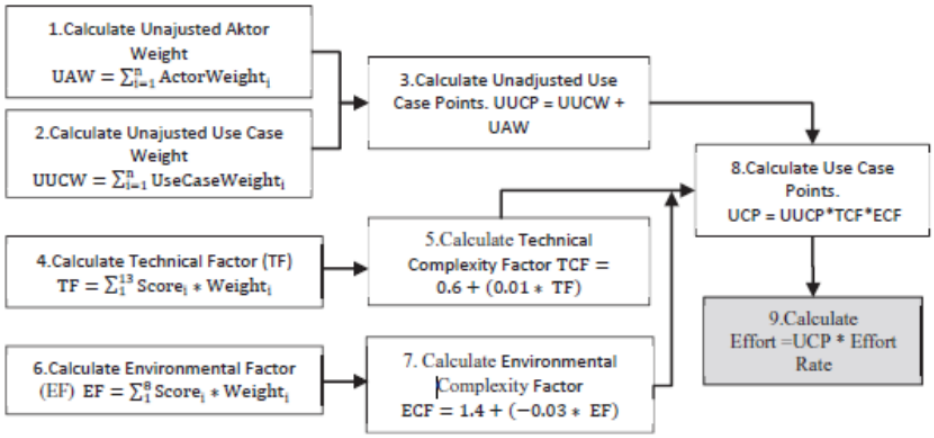
Apabila perkiraan biaya perangkat lunak jauh dibawah biaya aktual yang dikeluarkan, maka proyek akan menjadi sangat tidak efisien. Bahkan agar proyek dapat diselesaikan tepat waktu pun, akan diperlukan sumberdaya yang lebih banyak dari yang direncanakan dan kadang juga akan terjadi pengurangan fungsionalitas dan kualitas dari produk yang

dihasilkan (Boehm, 1981). Hal ini dapat menyebabkan kerugian dalam proyek perangkat lunak. Oleh karena itu, Metode SEE menjadi hal yang sangat berpengaruh dalam Software Project Planning (SPP) (Silhavy, Silhavy, & Prokopova, 2018).

Pendekatan SEE dapat dikategorikan menjadi dua yaitu metode *algorithmic* dan *non-algorithmic*. Metode *non-algorithmic* menggunakan pendekatan analogis, historis, dan pertimbangan atau opini dari ahli. Metode ini masih banyak digunakan oleh perusahaan untuk merencanakan proyek perangkat lunak (Aljohani & Qureshi , 2017), salah satunya adalah metode *Guesstimating*. Sedangkan metode *algorithmic* lebih menggunakan pendekatan matematis dan pemodelan perangkat lunak untuk menghitung sumber daya yang diperlukan dalam proyek. Salah satu model perangkat lunak yang bisa digunakan dalam memodelkan kebutuhan perangkat lunak adalah *Use Case* (Somerville, 2011).

Use Case dalam penelitian yang dilakukan oleh Neill dkk. dinyatakan dapat menjadi pendekatan SEE yang lebih reliabel (Neill & Laplante, 2003). SEE dengan menggunakan *Use Case* sebagai model estimasi telah banyak dilakukan, salah satunya adalah metode *Use Case Point* (UCP) (Karner, 1993). Metode UCP telah dibuktikan dalam penelitian Khatibi dkk bahwa metode tersebut dapat digunakan untuk memperkirakan SEE dengan baik (Khatibi & Jawawi, 2010). Metode UCP merupakan turunan dari metode *Function Point Analysis* (FPA) yang bertujuan untuk menyediakan metode estimasi sederhana dengan berorientasi pada objek proyek perangkat lunak. Objek-objek perangkat lunak yang dilibatkan dalam metode ini adalah: Tipe Aktor (UAW), Bobot use case (UUCW), kompleksitas teknis pengembangan sistem (TF),

kompleksitas lingkungan pengembangan sistem (EF). Langkah-langkah perhitungan UCP seperti yang digambarkan pada gambar 1.



Gambar 2. Langkah-Langkah UCP

(Primandari A & Sholiq, 2015)

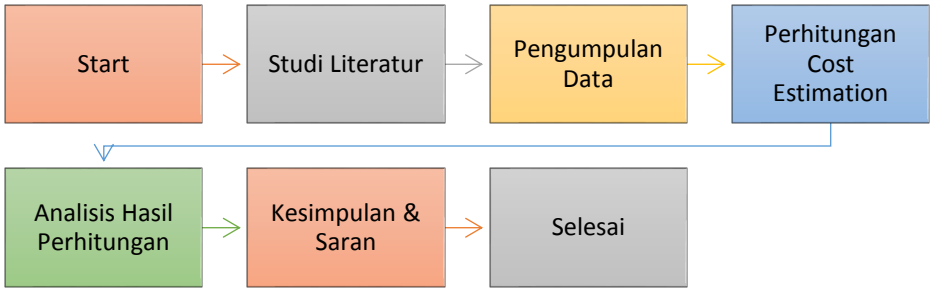
Metode UCP kemudian dikembangkan menjadi metode Extended Use Case Point (E-UCP) dimana metode E-UCP berfokus pada detail internal *use case*. Masing-masing use case didukung dengan adanya *use case narrative* yang menjelaskan detail *internal use case* (Periyasamy & Ghode, 2009). Unsur-unsur *Use case* yang digunakan pada metode E-UCP meliputi: *Input Parameter, Output parameter, Precondition, Postcondition, Successful Scenario*, dan *Exception*. Dalam E-UCP ditambahkan variabel *Unadjusted Use Case Narrative Weights* (UNW) sebagai salah satu objek perangkat lunak. Langkah perhitungan metode E-UCP pada dasarnya sama seperti yang dilakukan pada metode UCP, yang membedakan adalah pada perhitungan UUCP ditambahkan dengan variabel UNW.

Kirmani dkk dalam penelitiannya (Kirmani & Wahid, 2015) mengembangkan metode Revised Use Case Point (Re-UCP). Re-UCP merupakan kelanjutan dari UCP dan E-UCP sehingga seluruh langkah dan

parameter implementasi kedua metode tersebut dianalisis secara komprehensif untuk menghasilkan kerangka kerja estimasi usaha perangkat lunak yang dapat beradaptasi pada berbagai cakupan proyek. Re-UCP dipersiapkan untuk dapat digunakan pada berbagai tingkat kompleksitas dan menangani proyek *Agile*. Re-UCP memiliki 4 tipe aktor dan tipe *use case*. Empat tipe aktor Re-UCP yaitu *simple*, *average*, *complex*, dan *critical*. Bobot 4 tipe aktor secara berurutan adalah 1,2,3 dan 4. Bobot *use case* juga terdiri dari *simple*, *average*, *complex*, dan *critical*. Sementara untuk bobot *use case* secara berurutan adalah 5,10,15, dan 20. Formula yang digunakan untuk mencari nilai *unadjusted use case point* (UUCP) sama dengan formula pada *Use Case Point*. Pada *technical factor* terdapat faktor tambahan yaitu *scalability*. Sementara pada environment factor terdapat tambahan project *methodology*. Seperti mencari nilai UUCP, nilai Re-UCP juga diperoleh dari perkalian UUCP, TCF, dan ECF.

Pada penelitian ini penulis mencoba mengimplementasikan tiga metode *algorithmic* (UCP, E-UCP, Re-UCP) pada proyek perangkat lunak yang dikerjakan oleh perusahaan PT.XYZ di Kota Malang. Kemudian hasil dari implementasi tiga metode *algorithmic* tersebut dikomparasikan dengan metode *non-algorithmic* yang masih dilakukan oleh perusahaan tersebut dalam menghitung SEE. Kontribusi dari penelitian ini adalah untuk memberikan pertimbangan bagi manajer proyek perusahaan tersebut dan peneliti lainnya untuk memilih teknik SEE yang akurat. Susunan dari artikel ini adalah: (1) studi literatur teknik SEE berbasis Use Case; (2) simulasi metode SEE UCP, E-UCP, dan Re-UCP pada beberapa studi kasus di PT. XYZ (3) analisis hasil komparasi dengan estimasi *non-algorithm*ic yang dilakukan oleh PT.XYZ; (4) penarikan kesimpulan dan saran penelitian.

B. METODE



Gambar 3. Metodologi Penelitian

Adapun metodologi penelitian pada artikel ini seperti digambarkan pada gambar 3. Langkah pertama yang dilakukan adalah studi literatur tentang metode Effort Estimation menggunakan pendekatan Use Case. Dari hasil studi literatur didapatkan beberapa penelitian terdahulu yang terkait.

Langkah berikutnya adalah melakukan pengumpulan data terkait proyek yang dipilih baik berupa dokumen maupun kuesioner yang diisi oleh responden. Data yang diambil dalam penelitian ini adalah data terkait nilai biaya proyek berdasarkan metode *Guesstimating* yang dilakukan oleh perusahaan. Disamping itu digali juga data kebutuhan fungsional sistem berdasarkan dokumen *Software Requirement and Specification* (SRS), dokumen kontrak proyek, Kerangka Acuan Kerja (KAK) proyek, *use case diagram* perangkat lunak, dan skenario *use case* perangkat lunak proyek. Diagram dan skenario *use case* data uji divalidasi berdasarkan penelitian Anda dkk (Anda & Sjøberg, 2002). Data proyek yang digunakan dalam penelitian ini beserta dengan jumlah aktor dan case sesuai dengan rancangan kebutuhan sistem disajikan pada Tabel 1.

Tabel 1. Data Proyek

Nama Proyek	Biaya Proyek Aktual (ribu Rp.)	Jumlah Aktor	Jumlah Case	Sumber Dana
SIMPRO	44.150	4	19	Pemerintah
SIPAP	350.000	4	37	Swasta
SIMJALAN	250.000	4	40	Swasta

Responden dalam penelitian ini adalah tim proyek yang terdiri dari manajer proyek, analis sistem dan *programmer*. Responden kemudian diminta untuk melakukan pengisian lembar penilaian kompleksitas faktor teknis (*technical complexity factor*) yang di isi oleh analis sistem karena membahas mengenai kebutuhan non fungsional sistem. Sementara lembar penilaian kompleksitas faktor lingkungan (*environmental complexity factor*) diisi oleh manajer proyek karena berkaitan pengalaman dan motivasi tim.

Data tersebut menjadi dasar menghitung estimasi usaha melalui metode UCP, E-UCP dan Re-UCP. Estimasi usaha dijadikan dasar membuat estimasi waktu, sumber daya manusia, dan biaya sehingga dapat membuat penjadwalan dengan *gantt chart*. Setelah itu dilanjutkan menganalis hasil estimasi usaha, biaya, waktu, dan sumber daya manusia untuk memperoleh informasi lebih mendalam dan dapat menyelesaikan permasalahan proyek.

Perhitungan *Effort* dimulai dengan nilai *Unajusted Actor Weight* (UAW). UAW dihitung berdasarkan aktor yang terlibat dalam sistem dipetakan terhadap kategori aktor untuk mendapatkan bobot masing-masing aktor. Terdapat tiga kategori aktor pada metode UCP yaitu: 1) *Simple*, contohnya tipe aktor yang berinteraksi dengan sistem melalui *Application Programming Interface* (API); 2) *Average*, contohnya tipe aktor yang berinteraksi dengan sistem melalui protokol seperti

TCP/IP,FTP,HTTP dan aktor yang berinteraksi dengan sistem melalui command line; 3) *Complex*, tipe aktor yang berinteraksi dengan sistem melalui Graphical User Interface (GUI) atau halaman web. Sedangkan pada metode E-UCP tipe aktor dibagi ke dalam tujuh kategori yaitu: 1) *Very Simple*; 2) *Simple*; 3) *Less Average*; 4) *Average*; 5) *Complex*; 6) Very *Complex*; 7) *Most Complex*. Sedangkan pada metode Re-UCP digunakan empat tipe aktor yaitu: 1) *Simple*; 2) *Average*; 3) *Complex*; 4) *Critical* (Kirmani & Wahid, 2015). Perbedaan UAW pada ketiga metode ini disajikan pada Tabel 2.

Tabel 2. Perbandingan Tipe Aktor dan Bobotnya

Metode	Tipe Aktor	Bobot
UCP	Simple	1
	Average	2
	Complex	3
E-UCP	Verly Simple	0.5
	Simple	1
	Less Average	1.5
	Average	2
	Complex	2.5
	Very Complex	3
	Most Complex	3.5
Re-UCP	Simple	1
	Average	2
	Complex	3
	Critical	4

(Nhung, Hoc, & Hai, 2019)

Perhitungan yang dilakukan berikutnya adalah nilai *Unajusted Use Case Weight* (UUCW). Perhitungan dilakukan dengan memetakan setiap *use case* dalam sistem terhadap kriteria *use case* untuk mendapatkan bobot. Terdapat perbedaan kriteria use case pada masing-masing metode. Pada metode UCP terdapat tiga kriteria *use case* yaitu: 1) *Simple*; 2) *Average*; 3) *Complex*. Sedangkan pada metode E-UCP tipe *use case* dibagi ke dalam

lima kategori yaitu: 1) *Very Simple*; 2) *Simple*; 3) *Average*; 4) *Complex*; 6) Very *Complex*. Sedangkan pada metode Re-UCP digunakan empat tipe aktor yaitu: 1) *Simple*; 2) *Average*; 3) *Complex*; 4) *Critical*. Bobot masing-masing kriteria use case disajikan pada Tabel 3.

Tabel 3. Perbandingan Tipe Use Case dan Bobotnya

Metode	Tipe Use Case	Bobot
UCP	Simple	5
	Average	10
	Complex	15
E-UCP	Verly Simple	5
	Simple	10
	Average	15
	Complex	20
	Very Complex	25
Re-UCP	Simple	5
	Average	10
	Complex	15
	Critical	20

(Nhung, Hoc, & Hai, 2019)

Setelah didapatkan bobot aktor dan bobot *use case*, kemudian dilakukan perhitungan *Unajusted Use Case Point* (UUCP) dengan rumus:

$$UUCP = UUCW + UAW$$

khusus pada metode E-UCP ditambahkan variabel *Unadjusted Use Case Narrative Weights* (UNW) sebagai salah satu objek perangkat lunak yang berasal dari analisis narasi use case. Sehingga rumusnya menjadi:

$$UUCP = UUCW + UAW + UNW$$

Langkah berikutnya adalah menghitung *Technical Complexity Factor* (TCF) dan *Environmental Complexity Factor* (ECF) dimana pada metode UCP dan E-UCP digunakan tiga belas faktor teknis dan delapan faktor lingkungan. Sedangkan pada metode Re-UCP terdapat penambahan satu faktor teknis yaitu *Scalability* dan satu faktor lingkungan yaitu *Project*

Methodology. Masing-masing faktor dinilai bobotnya dengan cara melakukan observasi menggunakan instrumen kuesioner terhadap tim pelaksana proyek. Nilai yang didapatkan dari kuesioner akan dikalikan dengan bobot masing-masing faktor untuk mendapatkan total TCF dan ECF.

Setelah didapatkan TCF dan ECF, langkah berikutnya adalah menghitung Use Case Point dengan rumus:

$$UCP = UUCP \times TCF \times ECF$$

Hasil perhitungan UCP ini akan digunakan untuk menghitung effort dengan cara mendistribusikan hasil total konversi UCP terhadap *Hours of Effort* dengan rumus:

$$Hours\ of\ Effort = UCP \times staff\ hours$$

dimana *staff hours* adalah rata-rata jumlah jam kerja staf yang diperlukan untuk menyelesaikan sebuah *use case* (Saleh, 2011). *Hours of Effort* merupakan total *effort* dalam satuan *staff hour* yang kemudian didistribusikan pada setiap fase proyek untuk menghitung persentase *effort* pada masing-masing fase.

Setelah diketahui persentase *effort* masing-masing fase proyek, maka dapat dihitung beban biaya alokasi sumber daya dan waktu yang diperlukan pada setiap fase proyek. Sehingga dapat dihitung total biaya dan waktu yang diperlukan untuk menyelesaikan proyek perangkat lunak.

C. HASIL DAN PEMBAHASAN

Dalam penelitian ini, telah dilakukan penilaian terhadap tiga proyek perangkat lunak dan menggunakan tiga metode SEE, didapatkan hasil perhitungan UAW, UUCP, TCF, ECF dan *Effort Estimation*. Perbandingan

hasil perhitungan UAW disajikan pada Tabel 4, sedangkan perbandingan hasil perhitungan UUCW disajikan pada Tabel 5.

Tabel 4. Hasil Perhitungan ∑ UAW

Nama Proyek	∑ UAW		
	UCP	**E-UCP**	**Re-UCP**
SIMPRO	8	8	8
SIPAP	9	8	9
SIMJALAN	12	11	12

Tabel 5. Hasil Perhitungan ∑ UUCP

Nama Proyek	∑ UUCP		
	UCP	**E-UCP**	**Re-UCP**
SIMPRO	130	142	130
SIPAP	245	240	245
SIMJALAN	335	340	335

Dari hasil perhitungan UAW, UUCW dan TCF serta ECF, dilakukan perhitungan *Hours of Effort* dan kemudian jumlah *effort* yang didapatkan didistribusikan terhadap seluruh fase SDLC maksimal 42%, *ongoing process* pelaksanaan proyek seperti kegiatan manajemen proyek, dokumentasi maksimal 21%, dan untuk aktifitas *Quality Assurance*, Pengujian serta evaluasi proyek maksimal 37% (Saleh, 2011). Hasil pemetean *effort* untuk setiap aktifitas proyek kemudian digunakan untuk menghitung biaya sumber daya dengan berpedoman pada standar gaji SDM tiap kualifikasi yang diperlukan. Hasil pemetaan *effort* dan biaya dicontohkan pada Gambar 4.

No.	Aktivitas	Peran dalam *salary guide*	*% Effort*	*Effort*	Gaji/jam (Rp)	Total
			Software Development			
1.	*Requirement*	*System Analyst*	7,50%	55,11	113.124	6.234.496
2.	*Spesification*	*System Analyst*	7,50%	55,11	113.124	6.234.496
3.	*Design*	*System Analyst*	10%	73,48	113.124	8.312.662
4.	*Implementation*	*Software Engineer*	10%	73,48	75.416	5.541.775
5.	*Integration Testing*	*Test Analyst*	7,50%	55,11	113.124	6.234.496
6.	*Acceptance & Deployment*	*Software Engineer*	7,50%	55,11	75.416	4.156.331
					Sub-Total	36.714.257
			Ongoing Activity			
1.	*Project Management*	*Project Manager*	8,34%	61,28	137.499	8.426.567
2.	*Configuration Management*	*Software Engineer*	4,16%	30,57	75.416	2.305.378
3.	*Documentation*	*System Analyst*	4,16%	30,57	113.124	3.458.067
4.	*Training & Support*	*Software Engineer*	4,16%	30,57	75.416	2.305.378
5.	*Quality Assurance*	*Software QA*	8,34%	61,28	113.124	6.932.760
6.	*Evaluation & Testing*	*Test Analyst*	20,84%	153,14	113.124	17.323.587
					Sub-Total	40.751.739
					Total	**77.465.995**

Gambar 4. Pemetaan *Effort* tiap fase dan biaya SDM

Total biaya yang didapatkan pada perhitungan effort dan biaya SDM ini akan digunakan sebagai perbandingan dengan nilai aktual biaya proyek untuk mengukur deviasi yang terjadi. Adapun perhitungan deviasi biaya proyek yang didapatkan disajikan pada Tabel 6.

Tabel 6. Perhitungan Biaya Proyek

Nama Proyek	Biaya Proyek (dalam ribu Rp.)			
	Aktual/Budget	**UCP**	**E-UCP**	**Re-UCP**
SIMPRO	44.150	169.233	77.465	169.233
SIPAP	350.000	184.769	210.445	184.769
SIMJALAN	250.000	210.302	215.450	210.302

Dari hasil penelitian ini, didapatkan hasil perhitungan biaya yang relatif mendekati untuk 3 metode (UCP, E-UCP, dan Re-UCP). Ketiga metode tersebut dapat digunakan oleh perusahaan pelaksana proyek untuk memperkirakan effort yang diperlukan dan biaya yang harus disiapkan

untuk menyelesaikan pekerjaan perangkat lunak. Metode ini dapat memperkirakan biaya pengembangan perangkat lunak dengan akurat sesuai dengan *effort* yang dikeluarkan dalam proses membangun perangkat lunak.

Dalam memperhitungkan bobot *use case*, E-UCP memiliki kelebihan yaitu adanya tambahan parameter faktor kompleksitas yang akan memberikan lebih banyak spektrum penilaian kompleksitas use case sehingga penilaian kompleksitas use case menjadi lebih akurat. Lebih lengkap lagi, metode E-UCP menambahkan penilaian bobot naratif use case untuk menambah akurasi. Demikian juga dengan memperhitungkan bobot aktor, metode E-UCP menambahkan faktor kompleksitas agar lebih akurat.

Metode Re-UCP tidak jauh berbeda dengan UCP dalam melakukan penilaian bobot use case dan aktor. Namun dalam metode Re-UCP ditambahkan faktor *scalability* pada TCF dan *Project Methodology* pada ECF, kedua parameter kompleksitas ini cocok digunakan untuk menilai kompleksitas proyek perangkat lunak dengan pendekatan *agile*.

Ketiga metode tersebut berhasil mengidentifikasi *effort* dan biaya secara realistis dilihat dari sudut pandang *software cost estimation*. Namun jika dibandingkan dengan *budget* yang ditawarkan oleh pemberi kerja, terdapat selisih atau *gap* yang cukup jauh. Kemungkinan terjadinya hal ini adalah pihak pemberi kerja tidak menerapkan metode estimasi biaya perangkat lunak, melainkan memberikan alokasi budget berdasarkan ketersediaan dana. Hasil dari penelitian ini dapat juga digunakan oleh pihak pemberi kerja dalam mempertimbangkan budget proyek perangkat lunak.

D. PENUTUP

Simpulan dan Saran

Penelitian ini memberikan gambaran bahwa perhitungan SEE merupakan hal yang penting dalam sebuah proyek perangkat lunak. Metode yang cukup akurat untuk menghitung SEE adalah pendekatan use case based effort estimation. Hasil dari penggunaan metode use-case based effort estimation lebih akurat dibandingkan dengan metode *non-algoritmic*, seperti *guesstimating*, *expert judgement*, atau *budget oriented estimation*. Metode use case-based effort estimation dapat digunakan oleh kedua belah pihak (pemberi kerja dan pelaksana proyek) untuk memperkirakan biaya proyek dengan lebih akurat.

Saran untuk penelitian berikutnya adalah menggunakan metode use case-based effort estimation lainnya seperti Fuzzy Use Case Point, Ucp Sizing, Bayesian UCP, dan lainnya. Dan juga mengembangkan penelitian ke domain yang lebih spesifik misalnya pada proyek pemerintah atau swasta.

DAFTAR PUSTAKA

Aljohani, M., & Qureshi , R. (2017). Comparative Study of Software Estimation Techniques. *International Journal of Software Engineering & Applications (IJSEA), 8*(6), 39-53.

Anda, B., & Sjøberg, D. I. (2002). Towards an Inspection Technique for Use Case Models. *Proceedings of the 14th international conference on Software engineering and knowledge engineering*, (hal. 127-134).

Azzeh, M., & Nassif, A. (2018). Project productivity evaluation in early software effort estimation. *Journal of Software : Evolution and Process, 30*(12).

Boehm, B. (1981). *Software Engineering Economics.* Englewood Cliffs: Prentice-Hall.

Karner, G. (1993). Resource estimation for objector projects. *Object. Syst.*

Khatibi, V., & Jawawi, D. (2010). Software Cost Estimation Methods: A Review. *Journal of Emerging Trends in Computing and Information Sciences, 2*(1), 21-29.

Kirmani, M. M., & Wahid, A. (2015). Revised Use Case Point (Re-UCP) Model for Software Effort Estimation. *Oriental Journal of Computer Science and Technology*, 65-71.

Marchewka, J. (2003). *Information Technology Project Management: Providing Measurable Organizational Value* (5th ed.). Wiley.

Neill, C. J., & Laplante, P. A. (2003). Requirements Engineering: The State of the Practice. *IEEE Software, 20*(6), 40-45.

Nhung, H., Hoc, H., & Hai, V. (2019). A Review of Use Case-Based Development Effort Estimation Methods in the System Development Context. Dalam S. Radek , S. Petr, & P. Zdenka (Penyunt.), *3rd Computational Methods in Systems and Software 2019. 1046*, hal. 484-499. Zlin: Springer, Cham. doi:https://doi.org/10.1007/978-3-030-30329-7_44.

Periyasamy, K., & Ghode, A. (2009). Cost estimation using extended use case point. *2009 International Conference on Computational Intelligence and Software Engineering*, (hal. 3-7).

Primandari A, P., & Sholiq, S. (2015). Effort Distribution to Estimate Cost in Small to Medium Software Development Project with Use Case Points. *Procedia Computer Science, 72*, 78-85.

Project Management Institute. (2013). *A guide to the project management body of knowledge (PMBOK® guide)* (5th ed.). Pennsylvania: Project Management Institute, Inc.

Saleh, K. (2011). Effort and Cost Allocation in Medium to Large Software Development Projects. *INTERNATIONAL JOURNAL OF COMPUTERS, 5*(1), 74-79.

Schwalbe, K. (2011). *Information Technology Project Management* (REVISED 6th ed.). Boston, USA: Course Technology.

Silhavy, R., Silhavy, P., & Prokopova, Z. (2018). Evaluating subset selection methods for use case points estimation. *Information and Software Technology, 97*, 1-9.

Somerville, I. (2011). *Software Engineering* (9 ed.). Pearson.

The Standish Group. (2018). *CHAOS Chronicles. Technical Report.* The Standish Group International, Inc.

Trendowicz A., M. J. (2008). State of the Practice in Software Effort Estimation: A Survey and Literature Review. Dalam K. R. Huzar Z. (Penyunt.), *Third IFIP TC 2 Central and East European Conference* (hal. 232-245). Brno: Springer.