Estimasi Ukuran Perangkat Lunak Menggunakan Function Point Analysis - Studi Kasus Aplikasi Pengujian dan Pembelajaran Berbasis Web

Nur Rachmat
STMIK Global Informatika MDP
Magister Teknik Informatika, Universitas Sriwijaya
Jl. Rajawali 14 Palembang, Indonesia
rachmat.nur91@mdp.ac.id

Saparudin
Fakultas Ilmu Komputer, Universitas Sriwijaya
Palembang, Indonesia
saparudin@unsri.ac.id

***Abstrak*— Pengembangan perangkat lunak yang sukses salah satunya diawali dengan perancangan. Perancangan yang baik diawali dengan estimasi ukuran perangkat lunak yang akan dibangun. Salah satu metode pengukuran yang banyak digunakan adalah *Function Point Analysis* (FPA). Ukuran perangkat lunak bisanya disajikan dalam satuan *Lines of Code* (LOC). Dalam study kali ini, ukuran perangkat lunak berbasis web telah diestimasi dengan metode FPA. Hasil estimasi telah dibandingkan dengan ukuran aslinya setelah proses pengembangan.**

Kata Kunci*— *Software Size Estimation; Web Based; FPA; LOC;

I. PENDAHULUAN

Faktor keberhasilan proyek pengembangan perangkat lunak adalah keakurasian estimasi dan perencanaan dari aktivitas yang akan dilakukan, estimasi waktu, biaya dan kualitasnya. Estimasi dan perencanaan proyek perangkat lunak dapat dilakukan dengam mengukur perangkat lunak yang akan dibangun dan dikembangkan[1].

Salah satu metode yang banyak digunakan dalam mengukur estimasi ukuran dari perangkat lunak adalah Function Point Analysis (FPA) [2][3]. FPA pertama kali dikenalkan oleh Allan Albrecht pada tahun 1979 dan sekarang terus diperbaharui oleh *International Function Point User Group* (IFPUG) [4][5]. FPA berbasis pada jumlah fungsionalitas dan sekumpulan faktor dalam proyek perangkat lunak. FPA adalah metode standar dalam mengukur pengembangan perangkat lunak dari sisi pengguna.

Beberapa penelitian yang telah dilakukan berbasis *function point.* Metode yang diadopsi berdasarkan standard ISO untuk pengukuran fungsionalitas perangkat lunak dari IFPUG. Penelitian juga dilakuan untuk meningkatkan akurasi estimasi dengan pendekatan sistem cerdas logika fuzzy[5][6][7]. Hasil yang didapat menjadi lebih baik.

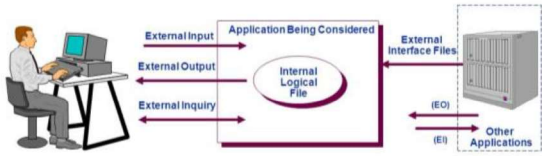
Penelitain berbasis FPA terdahulu telah dilakukan untuk mengukur estimasi ukuran perangkat lunak berbasis *mobile*[8]. Hasil yang didapatkan sangat akurat walaupun tanpa pendekatan sistem cerdas. Namun, aplikasi yang diukur adalah aplikasi mobile skala kecil dan pada bahasa pemrograman Java. Pada penelitian ini akan dilakukan penghitungan estimasi ukuran perangkat lunak pada aplikasi berbasis web dengan bahasa pemrograman PHP.

II. FUNCTION POINT ANALYSIS

FPA merupakan metode pengukuran perangkat lunak yang paling banyak digunakan di seluruh dunia. FPA pertama kali dikenalkan oleh Allan Albrecht pada tahun 1979 dan sekarang terus diperbaharui oleh *International Function Point User Group* (IFPUG). Terdapat beberapa metode pengukuran perangkat lunak selain FPA, sebagai contoh adalah *Lines of Code* (LOC)[2][9], [10].

A. Identifikasi User Function (UF)

FPA mengukur ukuran dari sebuah perangkat lunak dengan menghitung jumlah fungsionalitas dan kompleksitas dalam perangkat lunak dari sudut pandang pengguna seperti pada Gambar 1.



Gambar 1. Fungsionalitas dilihat dari sudut pendang pengguna

Dalam metode FPA terdapat 5 fungsi sebagai parameter pengukuran sebuah perangkat lunak, yaitu *External Input* (EI), *External Output* (EO), *Internal Logical File* (ILF), *External Interface File* (EIF) dan *External Inquiry* (EQ)[1].

- *External Input* (EI) adalah proses dasar yang memproses data dan informasi kontrol yang datang dari luar batasan aplikasi.

- *External Output* (EO) adalha sebuah proses dasar dimana hasil data dilewatkan dari dalam ke keluar dari batasan aplikasi.
- *Internal Logical File* (ILF) adalah kelompok data atau kelompok informasi kontrol yang digunakan dalam aplikasi.
- *External Interface* File (EIF) adalah kelompok data berelasi atau informasi kontrol yang dirujuk oleh aplikasi, tapi dipelihara oleh aplikasi lain.
- *External Inquiry* (EQ) fungsi utamanya adalah menyediakan informasi ke user melalui pengambilan/pemrosesan data atau informasi kontrol dari ILF/EIF.

B. *Kompleksitas Klasifikasi dari User Function*

Setelah proses identifikasi, setiap UF dalam perangkat lunak harus dihitung kompleksitasnya[8]. Setiap UF diklasifikasikan berdasarkan tingkat kompleksitas antara lain *height*, *medium* (*average*) dan *low*. Nilai dari setiap UF pada masing masing kategori kompleksitas dikalikan dengan *complexity weight* yang telah ditentukan pada Tabel I.

TABEL I. FUNCTION POINT COMPLEXITY[11]

User Function Types	Complexity Weight		
	Low	Medium	Height
External Input (EI)	3	4	6
External Output (EO)	4	5	7
Internal Logical File (ILF)	7	10	15
External Interface File (EIF)	5	7	10
External Inquiry (EQ)	3	4	6

C. *Mengitung Total Nilai Unadjusted Function Point (UFP)*

Setelah tingkat kompleksitas dari setiap jenis UF diidentifkasikan maka selanjutnya adalah menghitung Nilai *Unadjusted Function Point (UFP)* dengan formula (1).

$$UFP = \sum_{i=1}^{5} \sum_{j=1}^{3} x_{ij} \text{ x } w_{ij} \quad (1)$$

UFP dihitung dengan mejumlahkan *weight* dari sejumlah UF dimana x_{ij} adalah sejumlah UF jenis *i* dengan kompleksitas level *j*, dan w_{ij} adalah nilai *weight* untuk jenis *i* dengan kompleksitas level *j*[8]. Dengan kata lain UFP didapatkan dengan mengalikan nilai *weight* dengan jumlah fitur yang ada di setiap UF yang berbeda. Kemudian hasil yang didapat kelima fungsi dijumlahkan untuk mendapatkan nilai *Total Unadjusted Function Point (TUFP).*

D. *Menghitung Technical Complexity Adjustment*

Pada FPA diperlukan perhitungan tidak hanya jumlah kompleksitas dari beberapa fitur yang diberikan kepada pengguna, tetapi juga kepada opersional dari lingkungan sistem. Ada 14 faktor yang dibuat mempengaruhi tingkat kesulitan yang berhubungan dengan implementasi sistem [6]. Setiap faktor diberi nilai dari 0 sampai 5. Nilai 0 jika faktor tersebut tidak menimbulkan efek apapun dan 5 jika faktor tersebut sangat penting di perangkat lunak yang diukur. Dalam FPA, faktor tersebut disebut *Value Adjustment Factor* seperti terlihat pada Tabel II.

TABEL II. VALUE ADJUSTMENT FACTORS[11]

ID	Factor
C1	Data communications
C2	Distributed function
C3	Performance objectives
C4	Heavily used configuration
C5	Transaction rate
C6	On-line data entry
C7	End-user efficiency
C8	On-line update
C9	Complex processing
C10	Reusability
C11	Installation ease
C12	Operational ease
C13	Multiple sites
C14	Facilitate change

Ketika keseluruhan faktor dan skor masing-masing telah ditentukan, selanjutnya adalah menghitung nilai *Total Degree of Influence* (TDI) dengan menjumlahkan ke 14 faktor tersebut. Lalu TDI diubah menjadi nilai akhir *Technical Complexity Adjustment* dengan formula (2).

$$TCA = 0.65 + 0.01 \times TDI \quad (2)$$

E. *Menghitung Nilai Function Point dan Estimasi Ukuran*

Nilai *Function Point (FP)* dari perangkat lukan dihitung dengan mengalikan UFP dengan TCA sesuai formula (3).

$$FP = UFP \times TCA \quad (3)$$

Nilai FP dapat digunakan untuk mengukur LOC perangkat lunak, *cost*, *effort*, *resource* yang dibutuhkan dan lama pengerjaan perangkat lunak [2]. Untuk memperoleh estimasi ukuran perangkat lunak dalam satuan *Lines of Code* (LOC), nilai FP dikalikan dengan *Productivity Factor* berdasarkan bahasa pemrograman yang akan digunakan. *Productivity Factor* adalah jumlah kode logis per *function point* dan nilainya bervariasi untuk setiap bahasa pemrogramanan, seperti yang dibuat oleh Capers Jones pada Tabel III.

TABEL III. PRODUCTIVITY FACTOR PROGRAMMING LANGUAGE[11]

Programming Language	Productivity Factor
SQL	37
PHP	53
HTML/Javascript	58

III. STUDI KASUS

A. *Aplikasi Berbasis Web : Sistem Pengujian dan Pembelajaran*

Dalam penelitian ini, kasus yang diambil adalah sebuah aplikasi berbasis web. Sebuah sistem pengujian online berbasis komputer atau sering disebut dengan *Computer Based Test* (CBT) untuk pendidikan Sekolah Menengah Atas (SMA) di kota Palembang. Aplikasi ini secara spesifik dibangun agar dapat diakses melalui jaringan komputer baik lokal maupun internet. Aplikasi harus dapat dibuka melalui *browser* baik di *desktop*, *tablet*, ataupun *smartphone*.

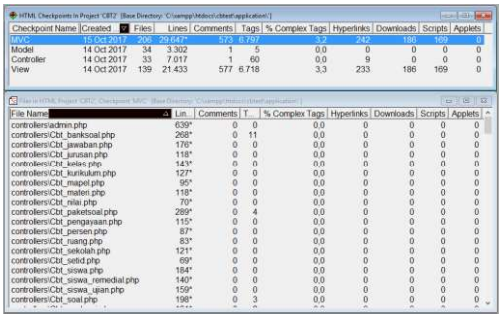
Melalui aplikasi ini, siswa dan guru dapat terhubung secara *realtime*. Guru dapat memberikan ujian, mengatur jadwal ujian, membuat soal ujian, menyampaikan materi pelajaran dan memproses nilai ujian siswa. Begitu juga siswa sebagai peserta didik dapat mengerjakan soal ujian sesuai jadwal yang telah ditentukan, membaca materi, serta melihat hasil ujian yang telah dilakukannya. Beberapa jenis ujian yang dapat dilakukan siswa melalui sistem ini adalah ulangan harian, ujian sekolah, simulasi ujian nasional, mengerjakan latihan soal, serta mengerjakan soal remedial yang bersifat pilihan ganda maupun esai. Soal ujian yang dikerjakan siswa dalam satu mata pelajaran harus saling berbeda untuk menghindari praktik kerja sama antar siswa.

Aplikasi ini dibangun dengan bahasa pemrograman PHP berbasis *Model View Controller* (MVC) dan berjalan pada server dengan sistem operasi Windows.

Ukuran asli dari proyek ini adalah 29647 baris kode dalam bahasa pemrograman PHP dari total jumlah 206 file MVC. Termasuk baris komentar pada kode sumber juga dihitung dikarenakan komentar sangat penting sebagaimana kode program. Gambar 2 menampilkan metrik dari kode sumber yang dikumpulkan dengan perangkat lunak *Source Monitor Software* [12].

B. *Penerapan FPA pada Proyek yang Dibangun*

Pertama-tama, *User Function* (UF) dari proyek harus dibuat dan diklasifikasikan sesuai tingkat kompleksitasnya. Terdapat 23 *External Input*, 6 *External Output*, 14 *Internal Logical File*, 6 *External Interface File* dan 5 *External Inquiry* dalam proyek ini yang tertera pada Tabel IV, V, VI, VII dan IX.



Gambar 2. Example of a figure caption. *(figure caption)*

TABEL IV. EXTERNAL INPUT

No	Deskripsi	Kompleksitas
1	Halaman Login	Low
2	Konfigurasi Sistem	Low
3	Manajemen Pengguna	Low
4	Upload Materi	Medium
5	Import Data Siswa	Medium
6	Import Data Guru	Medium
7	Import Data Mata Pelajaran	Medium
8	Input Nilai Soal Essay	Medium
9	Import Bank Soal Ujian	Height
10	Import Bank Soal Pengayaan	Height
11	Add/Edit/Delete Data Pengguna	Height
12	Add/Edit/Delete Data Siswa	Height
13	Add/Edit/Delete Data Guru	Height
14	Add/Edit/Delete Data Soal	Height
15	Add/Edit/Delete Data Materi Pelajaran	Height
16	Add/Edit/Delete Kelas	Height
17	Add/Edit/Delete Pertemuan	Height
18	Add/Edit/Delete Soal Pilihan Ganda	Height
19	Add/Edit/Delete Soal Esai	Height
20	Add/Edit/Delete Jadwal Ujian	Height
21	Halaman Pengerjaan Soal Ujian	Height
22	Halaman Pengerjaan Soal Pengayaan	Height
23	Halaman Pengerjaan Soal Remedial	Height

TABEL V. EXTERNAL OUTPUT

No	Deskripsi	Kompleksitas
1	Cetak Presensi Ujian	Low
2	Cetak Soal Ujian	Medium
3	Cetak Soal Pengayaan	Medium
4	Cetak Nilai	Height
5	Cetak Analisis Butir Soal	Height
6	Cetak Jawaban Ujian	Height

TABEL VI. INTERNAL LOGICAL FILE

No	Deskripsi	Kompleksitas
1	File Materi	Low
2	Gambar Soal	Medium
3	Gambar Jawaban	Medium
4	Suara Soal	Medium
5	Video Soal	Medium
	Database tabel User	Height
6	Database tabel Siswa	Height
7	Database tabel Guru	Height
8	Database tabel Mata Pelajaran	Height
9	Database tabel Kelas	Height
10	Database tabel Pertemuan	Height
11	Database tabel Nilai	Height
12	Database tabel Soal Pilihan Ganda	Height
13	Database tabel Soal Esai	Height
14	Database tabel Jawaban	Height

TABEL VII. EXTERNAL INTERFACE FILE

No	Deskripsi	Kompleksitas
1	File Excel Data Siswa	Medium
2	File Excel Data Guru	Medium
3	File Excel Data Jadwal Ujian	Medium
4	File Excel Data Mata Pelajaran	Medium
5	File Excel Soal Ujian	Height
6	File Excel Soal Pengayaan	Height

TABEL VIII. EXTERNAL INQUIRY

No	Deskripsi	Kompleksitas
1	Acak Soal Ujian	Medium
2	Proses Jawaban	Height
3	Proses Nilai	Height
4	Enkripsi Jawaban dan Kunci Jawaban	Height
5	Monitoring Ujian	Height

Selanjutnya penghitungan nilai UFP dilakukan dengan mengalian UF dan nilai kompleksitas masing-masing lalu menjumlahkan keseluhan nilan yang menghasilkan angka 427. Perhitungan UFP dapat dilihat pada Tabel IX.

TABEL IX. PENGHITUNGAN NILAI UFP

User Function Types	Complexity Weight x Counts			
	Low	*Medium*	*Height*	*Total*
External Input (EI)	3x3	4x5	6x15	119
External Output (EO)	4x1	5x2	7x3	35
Internal Logical File (ILF)	7x1	10x4	15x10	197
External Interface File (EIF)	5x0	7x4	10x2	48
External Inquiry (EQ)	3x0	4x1	6x4	28
Unadjusted Function Point (UFP)				427

Setelah menghitung UFP, selanjutnya adalah menghitung nilai TDI yang hasil perhitungannya adalah 64 dengan menggunakan *Value Adjustment Factor* yang dapat dilihat pada Tabel X.

TABEL X. PENGHITUNGAN NILAI TDI

ID	Factor	DI
C1	Data communications	5
C2	Distributed function	5
C3	Performance objectives	5
C4	Heavily used configuration	2
C5	Transaction rate	5
C6	On-line data entry	5
C7	End-user efficiency	5
C8	On-line update	5
C9	Complex processing	4
C10	Reusability	4
C11	Installation ease	4
C12	Operational ease	5
C13	Multiple sites	5
C14	Facilitate change	5
Total Degree of Inluence		64

Hasil perhitungan TDI dimasukkan ke dalam formula (2) dan menghasilkan nilai TCA sebesar 1.29. Akhirnya FP dari sistem didapatkan dengan mengalikan nilai UFP dengan TCA seperti pada formula (5).

$$TCA = 0.65 + 0.01 \times 64 = 1.29 \quad (4)$$

$$FP = 427 \times 1.08 = 550.83 \quad (5)$$

Untuk mendapatkan nilai estimasi ukuran dari sistem dalam *Lines of Codes* (LOC), maka nilai FP dikalikan dengan 53 yang merupakan *productivity factor* dari bahasa pemrograman PHP, sehingga LOC didapatkan seperti pada formula (6).

$$LOC = 550.83 \times 53 \cong 29194 \quad (6)$$

Ukuran proyek telah diestimasi dengan nilai estimasi 29194 baris kode dengan menggunakan metode FPA. Setelah proyek selesai, ukuran asli dari proyek adalah 29647 baris kode. Jika nilai estimasi dibandingkan dengan nilai ukuran asli, maka perbedaannya adalah sekitar 1,5%. Estimasi yang dilakukan hampir mendekati hasil penelitian sebelumnya yang mencapai 1.2% mengingat proyek

yang dihitung merupakan aplikasi yang berbeda dan skala besar. Bagaimanapun juga akan selalu didapatkan perbedan yang besar antara estimasi dan ukuran sesunguhnya dikarenakan proses identifikasi jenis fungsi dan klasifikasi kompleksitas adalah hal yang sulit. Sehingga perlu pendekatan yang lebih baik untuk mengantisipasi perbedaan antara nilai estimasi dan ukuran aslinya.

IV. SIMPULAN

Dalam penelitian ini telah dijelaskan secara detail metode pengukuran estimasi perangkat lunak berbasis *Function Point Analysis* (FPA). Metode FPA telah diaplikasikan untuk menghitung estimasi ukuran perangkat lunak pada aplikasi berbasis web yang dikembangkan oleh penulis serta hasil perhitungan telah dievaluasi. Aplikasi ini termasuk aplikasi berukuran besar layaknya sebuah proyek sistem informasi. Hasil yang diperoleh menunjukkan keakurasian yang tinggi jika dibandingkan dengan penelitian sebelumnya.

Penelitian ini menjadi pembelajaran bagi penulis untuk mengembangkan aplikasi-aplikasi lain dikemudian hari. Dengan menghitung nilai estimasi terlebih dahulu maka memungkinkan untuk menghitung estimasi sumber daya yang dibutuhkan, lama pengerjaan serta biaya pengembangan sehingga kegagalan dan kerugian proses pengembangan dapat dihindari. Sebagai penelitian lanjutan, dapat dilakukan pengukuran dengan metode estimasi lain yang berbasis *class* atau *use case* seperti Class Point[13] dan Use Case Point[14].

Referensi

[1] K. Sangeetha and P. P. Dalal, "A Review paper on Software Effort Estimation Methods," no. 3, pp. 163–169, 2015.

[2] N. Balaji, N. Shivakumar, and V. V. Ananth, "Software Cost Estimation using Function Point with Non Algorithmic Approach," vol. 13, no. 8, 2013.

[3] E. Ng'ang'a and I. Tonui, "A Survey on Software Sizing for Project Estimation International Journal of Advanced Research in," no. January, 2015.

[4] IFPUG, "International Function Point User Group." [Online]. Available: http://www.ifpug.org. [Accessed: 16-Oct-2017].

[5] F. Ferrucci, C. Gravino, and F. Sarro, "Conversion from IFPUG FPA to COSMIC: Within-vs Without-Company Equations," in *2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications*, 2014, pp. 293–300.

[6] R. Saptono and G. D. Hutama, "Peningkatan Akurasi Estimasi Ukuran Perangkat Lunak dengan Menerapkan Logika Samar Metode Mamdani," no. March, 2017.

[7] S. wisnu Murti, R. Saptono, and E. Suryani, "Comparison Analysis of Weight Value Changing in Function Point Analysis Between Fuzzy Inference System Mamdani and Tsukamoto for Software Size Estimation," 2016.

[8] V. Tunalı, "Software Size Estimation Using Function Point Analysis – A Case Study for a Mobile Application," *7. Mühendislik ve Teknol. Sempozyumu*, no. May, pp. 73–76, 2014.

[9] P. Ochieng, W. Mwangi, and S. M. Mwgha, "Software Size Estimation in Incremental Software Development based on Improved Pairwise Comparison Matrices," *Int. J. Comput. Appl.*, vol. 93, no. 5, pp. 975–8887, 2014.

[10] A. Arshad, "A Critical Review of Software Size and Effort Estimation," vol. 3, no. 2, pp. 96–99, 2014.

[11] C. Jones, *Applied Software Measurement: Global Analysis of Productivity and Quality*, 3rd ed. McGraw-Hill Education Group, 2008.

[12] Campwood Software, "Source Monitor Version 3.5." [Online]. Available: http://www.campwoodsw.com/sourcemonitor.html. [Accessed: 14-Oct-2017].

[13] G. Costagliola, F. Ferrucci, G. Tortora, and G. Vitiello, "Class point: an approach for the size estimation of object-oriented systems," *IEEE Trans. Softw. Eng.*, vol. 31, no. 1, pp. 52–74, Jan. 2005.

[14] K. Periyasamy and A. Ghode, "Cost Estimation Using Extended Use Case Point (e-UCP) Model," in *2009 International Conference on Computational Intelligence and Software Engineering*, 2009, pp. 1–5.