Jurnal Sistem Informasi (JSI), VOL. 1, NO. 2, Agustus 2009 *Halaman 79-87*
ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

PENGUKURAN KUALITAS PERANGKAT LUNAK BERDASARKAN KOMPLEKSITAS MENGGUNAKAN METODE FUNCTION POINT

Fathoni
Fakultas Ilmu Komputer, Universitas Sriwijaya
Email : Fathoni@Unsri.ac.id

Abstrak

Before 1969, there was a crisis related to software development conducted. The crisis occurred because most of developed software was likely to produce a new problem rather than giving benefit to its user. In fact, the developed software didn't meet with what user need. Moreover, the management and developer had difficulties to measure software quality during software development process. Metode Function Point can be used to solve this problem. It has a set of capability to list important information to predict what resource should be provided. This prediction gives a basic thing for Software Company to prepare their proposal or project plan. As well, the method could prevent or, at least, reduce a risk of managerial mistake due to underestimate in project cost planning

Key words: Quality, Software, Funtion Point

1. PENDAHULUAN

Pada pertengahan tahun 60 sampai 70-an banyak dikembangkan sistem-sistem perangkat lunak yang besar. Sistem-sistem yang dikembangkan ini banyak yang dipandang tidak efisien, kurang berhasil, bahkan banyak yang gagal. Kegagalan ini disebabkan karena tidak tersedianya teknik pengembangan perangkat lunak yang baik. Pada awal tahun 70-an mulai muncul metodologi-metodologi pengembangan perangkat lunak yang cukup baik.

Pengembangan perangkat lunak dapat diartikan sebagai proses membuat suatu perangkat lunak baru untuk menggantikan perangkat lunak lama secara keseluruhan atau memperbaiki perangkat lunak yang telah ada. Agar lebih cepat dan tepat dalam mendeskripsikan solusi dan mengembangkan perangkat lunak, juga hasilnya mudah dikembangkan dan dipelihara, maka pengembangan perangkat lunak memerlukan suatu metodologi khusus. Metodologi pengembangan perangkat lunak adalah suatu proses pengorganisasian kumpulan metode dan konvensi notasi yang telah didefinisikan untuk mengembangkan perangkat lunak. Secara prinsip bertujuan untuk membantu menghasilkan perangkat lunak yang berkualitas. Penggunaan suatu metodologi sesuai dengan persoalan yang akan dipecahkan dan memenuhi kebutuhan pengguna akan menghasilkan suatu produk perekayasaan yang berkualitas dan terpelihara serta dapat menghindari masalah-masalah yang sering terjadi seperti estimasi penjadwalan dan biaya, perangkat lunak yang tidak sesuai dengan keinginan pengguna dan sebagainya.

Metodologi pengembangan perangkat lunak (atau disebut juga model proses atau paradigma rekayasa perangkat lunak) adalah suatu strategi pengembangan yang

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

memadukan proses, metode, dan perangkat (*tools*). Disamping ketiga hal tersebut diperlukan juga suatu model untuk pengukuran kualitas perangkat lunak yang akan dibangun. Kualitas perangkat lunak dapat dihitung pada saat proses rekayasa perangkat lunak ataupun setelah diserahkan kepada pemakai. Satuan ukuran kualitas perangkat lunak pada saat proses rekayasa dapat meliputi; Kompleksitas program, modularitas yang efektif dan besarnya program.

2. PERMASALAHAN

Sebelum tahun 1969 terjadi krisis terhadap pengembangan perangkat lunak yang dilakukan. Krisis ini disebabkan karena sebagian besar perangkat lunak yang dikembangkan ternyata lebih banyak menimbulkan masalah baru daripada menyelesaikan permasalah yang sudah ada. Hal ini disebabkan oleh perangkat lunak yang dibuat ternyata tidak sesuai dengan kebutuhan yang diharapkan. Selain itu, fihak Manajemen dan pengembang mengalami kesulitan dalam mengukur kualitas perangkat lunak yang dibangun selama tahap pengerjaan pembuatan.

3. TUJUAN PENULISAN

Memberikan pengertian dan pemahaman terhadap model pengukuran kualitas perangkat lunak yang dikembangkan menggunakan **Motode Function Point**

4. MANFAAT PENULISAN

Bagi pengembang, mengukur Kualitas Perangkat Lunak bermanfaat untuk merencanakan sumber daya, biaya dan durasi yang diperlukan untuk membangun software. Selain itu, pengembang juga dapat mengevaluasi kualitas produk dengan cara membandingkan volume sistem dengan banyaknya error (*error-count*) dalam software yang dikerjakan. Sementara dari perspektif bisnis, Kualitas Perangkat Lunak dapat menjadi dasar untuk menentukan nilai harga dari produk software yang bersangkutan.

Sedangkan bagi manajemen, pengukuran kualitas perangkat lunak dapat digunakan untuk menilai apakah proses pengerjaan pembangunan perangkat lunak yang dilakukan telah sesuai dengan apa yang diharapkan.

5. STUDY PUSTAKA

Dahulu orang banyak mengukur volume dari suatu software menggunakan LOC (Lines Of Code), yaitu suatu teknik pengukuran besar software dengan cara menghitung baris kode program yang ada. Teknik ini mempunyai sifat yang menjadi kekurangannya yaitu :

1. Relatif terhadap bahasa/tool pemrograman dan gaya pengkodean programer.
   LOC sangat tergantung pada karakteristik tool pemrograman yang digunakan dan gaya pengkodean programer. Sebagai contoh dalam bahasa BASIC kode sebagai berikut :

   ```
   a = a + 1
   ```

   hanya membutuhkan 1 baris kode. Sedangkan untuk mendapatkan hasil yang sama dalam bahasa PASCAL kode tersebut dikonversi sebagai berikut :

   ```
   program x;
   var
     a : integer;
   ```

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

```
begin
  a := a + 1;
end.
```

yang membutuhkan 6 baris kode. Juga perhatikan contoh perbedaan gaya pengkodean dari 2 skrip program berikut yang mengakibatkan perbedaan LOC.

Adjustment factore

Kode 1 (2 baris)

```
a := a + 1; b :=5;
if a = 2 then a=1;
```

Kode 2 (4 baris):

```
a := a + 1;
b := 5;
if a = 2 then
  a := 1;
```

2. LOC tidak bisa ditentukan sebelum proyek pengembangan menyelesaikan tahapan implementasi (pengkodean). Oleh karena itu, LOC tidak dapat dimanfaatkan untuk merencanakan proses pengembangan dan tidak pula dapat digunakan untuk memperkirakan harga produk. Selesainya tahapan implementasi adalah suatu fase yang sangat terlambat untuk menyusun estimasi sumber daya.

Dari kekurangan tersebut maka timbul keinginan untuk mendapatkan suatu teknik pengukuran volume software yang tidak hanya berdasar pada banyaknya baris kode program, namun lebih kearah sesuatu yang dapat diukur lebih awal pada *software development life cycle* sehingga kemudian munculah gagasan metode Function Point.

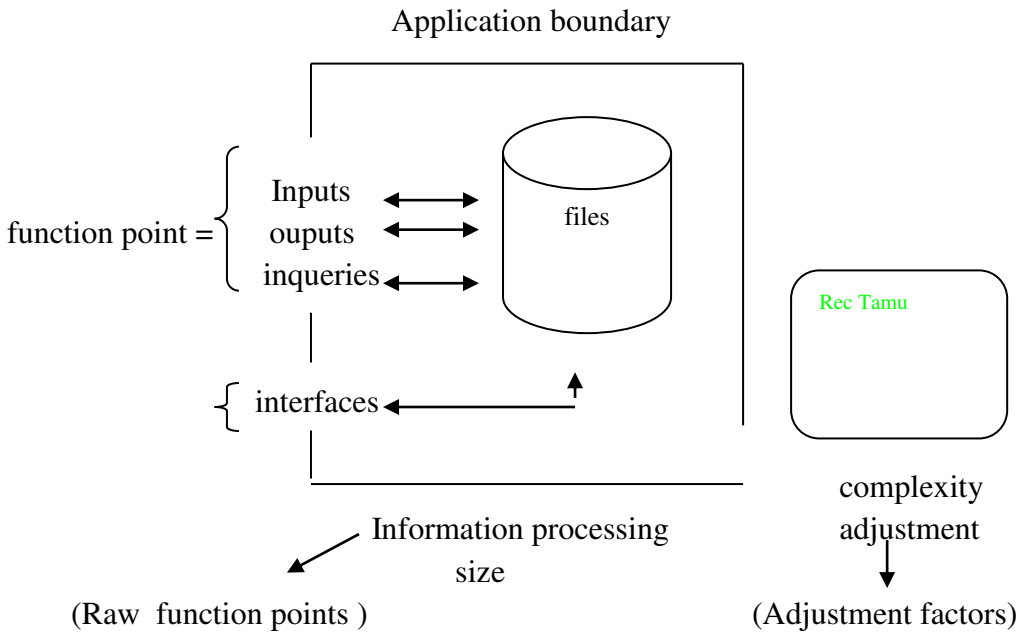
Keunggulan dari metode Function Point adalah kemampuannya untuk menyediakan perkiraan volume proyek dalam bentuk sumber daya pengembangan yang dibutuhkan, sebelum proyek tersebut jauh melangkah. Perkiraan ini memberikan dasar penting untuk perkiraan sumber daya yang dibutuhkan perusahaan software dalam mempersiapkan proposal tender dan *project plan*. Penggunaan metode seperti ini juga dapat mencegah atau setidaknya mengurangi secara substansial risiko dari kesalahan manajerial karena *underestimate* pada perencanaan biaya proyek.

Metode function point perhitungannya didasarkan pada ukuran banyak dan kompleksitas fungsi yang diinginkan dalam proyek software. Function point dapat dihitung melalui telah dokumentasi *requirement* fungsional sistem oleh seorang profesional. Metode function point diperkenalkan pertama kali oleh Albrecht pada tahun 1979. Perkembangan selanjutnya metode ini digunakan secara luas untuk keperluan komersial oleh banyak pihak namun masih dipandang sebagai eksperimental oleh banyak ilmuwan atau profesional. Memandang pentingnya metode ini, maka riset-riset dilakukan untuk memvalidasi, meningkatkan dan mengadaptasikan metode ini ke dalam beberapa jenis sistem software seperti sistem software real-time dan sistem software berorientasi obyek.

6. METODE FUNCTION POINT

Pendekatan metode function point diusulkan oleh Albrecht yang disebut sebagai matrik function point, matrik ini diperoleh dari keterhubungan dasar antara domain informasi software dan kompleksitas software (pada gambar 1) Juga model yang dikembangkan dalam kaitan ini meliputi model estimasi besaran usaha pengembangan proyek dengan pendekatan function point dan alat bantu berupa software untuk memasukkan nilai parameter function point tersebut dan menampilkan model yang dihasilkan.

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index



Gambar 1 . Analisis function point

Perhitungan dengan metode Function Point menuntut untuk dilakukan oleh seorang profesional yang berpengalaman karena memiliki tingkat subyektifitas yang cukup tinggi. Metode ini sendiri terdiri dari banyak varia. Variasi yang adalah pada langkah/tahapan yang ada maupun pada isi dari tiap tahapan. Varian-varian ini timbul karena metode ini dapat diubah sesuai dengan kebijakan perusahaan pengembang software. Namun apapun varian yang digunakan oleh pengembang, hendaknya digunakan dengan konsisten agar tercipta komparasi yang benar antara software-software yang dinilai.

Pada tulisan ini penulis memberikan contoh berdasarkan publikasi varian yang populer seperti Gramus dan Herron (1996), IEEE (2000), Caldiera dkk (1998) yang menghasilkan manual penggunaan function point seperi IFPUG 3, IFPUG 4 dan Mark II. Contoh berikut penulis buat dengan beberapa penyesuaian sesuai dengan pengalaman dan pengamatan penulis. Tahapan-tahapan yang ada dalam menentukan function point adalah sebagai berikut :

Langkah 1 : *Menghitung crude function points (CFP)*. Jumlah dari komponen fungsional sistem pertama kali diidentifikasi dan dilanjutkan dengan mengevaluasi kuantitasi bobot kerumitan dari tiap komponen tersebut. Pembobotan tersebut kemudian dijumlahkan dan menjadi angka CFP.

Perhitungan CFP melibatkan 5 tipe komponen sistem software berikut :

1. Jumlah macam aplikasi input
2. Jumlah macam aplikasi output
3. Jumlah macam aplikasi query online – aplikasi ini berhubungan dengan query terhadap data yang tersimpan.
4. Jumlah macam file/tabel logic yang terlibat
5. Jumlah macam interface eksternal – output atau input yang dapat berhubungan dengan komputer lewat komunikasi data, CD, disket, dan lain-lain.

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

Kemudian diberikan faktor bobot pada tiap komponen di atas berdasarkan kompleksitasnya. Tabel 1 di bawah ini merupakan contoh blanko pembobotan tersebut.

Tabel 1: Blanko penghitungan CFP

Komponen Sistem Software	**Level kompleksitas**									**Total CFP**
	Sederhana			**Menengah**			**Kompleks**			
	Count	Faktor Bobot	Point	Count	Faktor Bobot	Point	Count	Faktor Bobot	Point	
	A	B	C= AxB	D	E	F= DxE	G	H	I= GxH	J=C+F+I
Input		3			4			6		
Output		4			5			7		
Query Online		3			4			6		
File logic		7			10			15		
Interface Eksternal		5			7			10		
Total CFP										

Langkah 2 : *Menghitung faktor pengubah kompleksitas relatif/relative complexity adjustment factor (RCAF)* untuk proyek tersebut.

RCAF berfungsi untuk menghitung kesimpulan kompleksitas dari sistem software dari beberapa subyek karakteristik. Penilaian berskala 0 sampai 5 diberikan pada tiap subyek yang paling berpengaruh terhadap usaha pengembangan yang dibutuhkan. Blanko penilaian yang diusulkan penulis diberikan seperti Tabel 2.

Tabel 2: Blanko penghitungan RCAF

No	Subyek	Nilai
1	Tingkat kompleksitas kehandalan backup/recovery	0 1 2 3 4 5
2	Tingkat kompleksitas komunikasi data	0 1 2 3 4 5
3	Tingkat kompleksitas pemrosesan terdistribusi	0 1 2 3 4 5
4	Tingkat kompleksitas kebutuhan akan kinerja	0 1 2 3 4 5
5	Tingkat kebutuhan lingkungan operasional	0 1 2 3 4 5
6	Tingkat kebutuhan knowledge pengembang	0 1 2 3 4 5
7	Tingkat kompleksitas updating file master	0 1 2 3 4 5
8	Tingkat kompleksitas instalasi	0 1 2 3 4 5
9	Tingkat kompleksitas aplikasi input, output, query online dan file	0 1 2 3 4 5
10	Tingkat kompleksitas pemrosesan data	0 1 2 3 4 5
11	Tingkat ketidakmungkinan penggunaan kembali dari kode (reuse)	0 1 2 3 4 5

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

12	Tingkat variasi organisasi pelanggan	0 1 2 3 4 5
13	Tingkat kemungkinan perubahan/fleksibilitas	0 1 2 3 4 5
14	Tingkat kebutuhan kemudahan penggunaan	0 1 2 3 4 5
	Total = RCAF	

Langkah 3 : *Menghitung Function Point (FP)* dengan rumus :

$$FP = CFP \times (0.65 + 0.01 \times RCAF)$$

Nilai function point untuk sistem software tersebut kemudian dihitung berdasarkan hasil dari tahap 1 dan 2 yang dimasukkan ke dalam formula

7. STUDI KASUS PENGGUNAAN – *ATTEND MASTER*

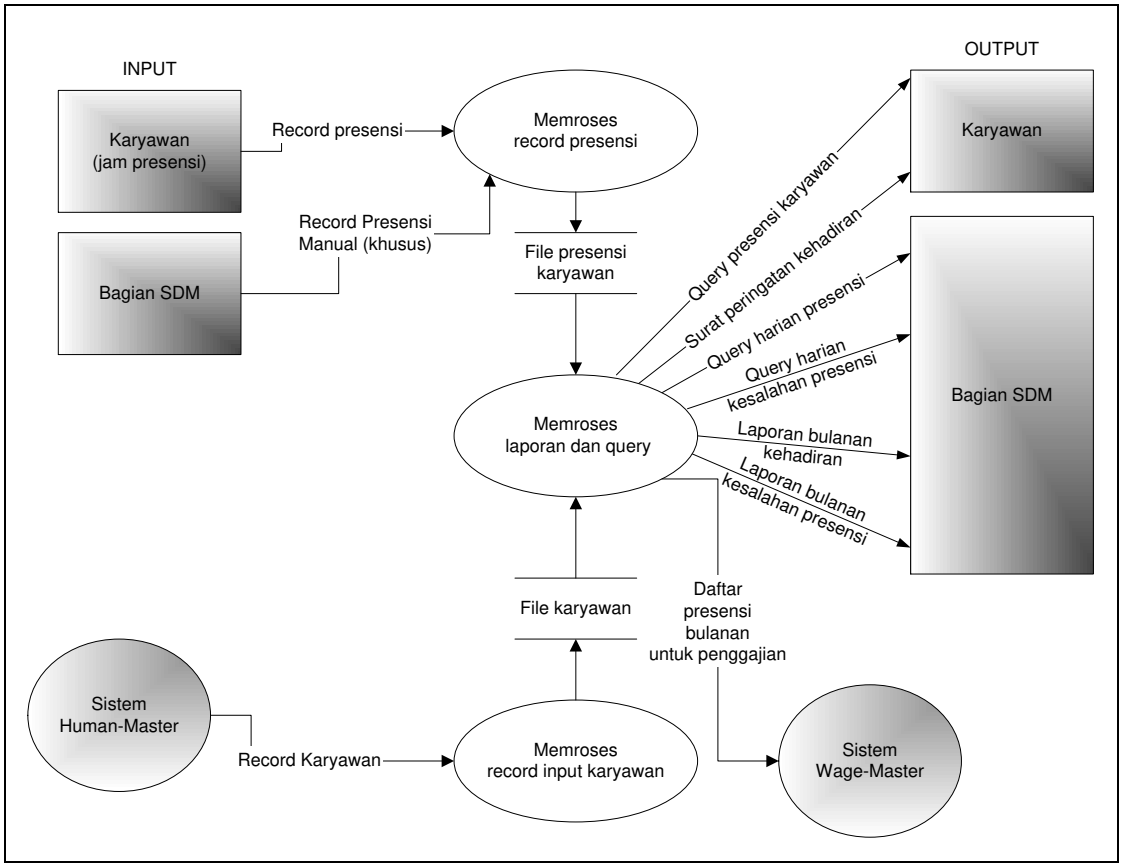
Akan dibangun sebuah sistem presensi karyawan bernama *Attend-Master* yang direncanakan dapat melayani bisnis kelas kecil sampai menengah dengan karyawan sebanyak 10-100 orang. Sistem tersebut direncanakan akan memiliki interface dengan paket software dari perusahaan lain yaitu *Human-Master*, yang melayani sumber daya manusia dan *Wage-Master* yang melayani penggajian. Attend-Master direncanakan dapat menghasilkan beberapa laporan dan query online. Dari dokumentasi requirement sistem software yang direncanakan ini, didapatkan Data Flow Diagram (DFD) yang ditunjukkan pada Gambar 1. Dari gambar tersebut dapat dihitung nilai function point untuk sistem software Attend-Master yang diajukan.

Langkah 1 : ***Menghitung crude function points (CFP)***

- Jumlah aplikasi input – 2
- Jumlah aplikasi output – 3
- Jumlah query online – 3
- Jumlah file logic – 2
- Jumlah interface eksternal – 2

Derajat kompleksitas (sederhana, menengah atau kompleks) kemudian dievaluasi untuk tiap komponen untuk mendapatkan nilai CFP seperti contoh pada Tabel 3

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index



Gambar 1: Data Flow Diagram dari *Attend-Master*

Tabel 3: Contoh penghitungan CFP untuk *Attend Master*

Komponen Sistem Software	**Level kompleksitas**									**Total CFP**
	Sederhana			**Menengah**			**Kompleks**			
	Count	Faktor Bobot	Point	Count	Faktor Bobot	Point	Count	Faktor Bobot	Point	
	A	B	C= AxB	D	E	F= DxE	G	H	I= GxH	J=C+F+I
Input	**1**	3	**3**	**---**	4	**---**	**1**	6	**6**	**9**
Output	**---**	4	**---**	**2**	5	**10**	**1**	7	**7**	**17**
Query Online	**1**	3	**3**	**1**	4	**4**	**1**	6	**6**	**13**
File logic	**1**	7	**7**	**---**	10	**---**	**1**	15	**15**	**22**
Interface Eksternal	**---**	5	**---**	**---**	7	**---**	**2**	10	**20**	**20**
Total CFP										**81**

Langkah 2 : ***Menghitung relative complexity adjustment factor (RCAF)***

Evaluasi terhadap karakteristik kompleksitas dari *Attend-Master* dan perhitungan dari RCAF digambarkan pada Tabel 4.

Tabel 4: Contoh penghitungan RCAF

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

No	Subyek	Nilai
1	Tingkat kompleksitas kehandalan backup/recovery	0 1 2 3 **4** 5
2	Tingkat kompleksitas komunikasi data	0 **1** 2 3 4 5
3	Tingkat kompleksitas pemrosesan terdistribusi	0 **1** 2 3 4 5
4	Tingkat kompleksitas kebutuhan akan kinerja	0 1 2 3 **4** 5
5	Tingkat kebutuhan lingkungan operasional	0 **1** 2 3 4 5
6	Tingkat kebutuhan knowledge pengembang	0 1 2 3 4 **5**
7	Tingkat kompleksitas updating file master	0 1 **2** 3 4 5
8	Tingkat kompleksitas instalasi	0 1 **2** 3 4 5
9	Tingkat kompleksitas aplikasi input, output, query online dan file	0 1 2 3 4 **5**
10	Tingkat kompleksitas pemrosesan data	0 1 2 3 **4** 5
11	Tingkat ketidakmungkinan penggunaan kembali dari kode (reuse)	0 1 2 **3** 4 5
12	Tingkat variasi organisasi pelanggan	0 **1** 2 3 4 5
13	Tingkat kemungkinan perubahan/fleksibilitas	0 1 2 3 **4** 5
14	Tingkat kebutuhan kemudahan penggunaan	0 1 2 3 **4** 5
	Total = RCAF	**41**

Langkah 3 : ***Menghitung Function Point (FP)***
Dengan menggunakan rumus FP maka didapat :

$$FP = CFP \times (0.65 + 0.01 \times RCAF) = 81 \times (0.65 + 0.01 \times 41) = 85.86$$

8. KELEBIHAN DAN KEKURANGAN METODE FUNCTION POINT

Kelebihan utama :

1. Perkiraan dapat disiapkan pada tahap pra-proyek dan oleh karena itu maka manajemen dapat didukung dalam usaha persiapan proyek.
2. Karena metode ini berbasiskan pada dokumen spesifikasi requirement (tidak berdasarkan pada tool pengembangan atau gaya pengkodean programer), kehandalan metode ini relatif tinggi.

Kekurangan utama:

1. Hasil perhitungan FP tergantung pada manual penggunaan function point yang digunakan.
2. Terkadang ada beberapa proyek yang tidak memiliki dokumen spesifikasi requirement mendetail pada tahap pra-proyek.
3. Seluruh proses penghitungan memerlukan profesional yang berpengalaman
4. Banyaknya evaluasi yang dibutuhkan berdampak pada hasil yang terlalu subyektif
5. Penghitungan FP dilakukan hanya didasarkan pada sistem pemrosesan data. Padahal aspek-aspek lain dari pengembangan sistem software juga ikut berpengaruh terhadap pengembangan itu sendiri. Oleh karena itu metode FP tidak dapat diterapkan secara universal atau masih membutuhkan dukungan perhitungan faktor lainn untuk memperkuat perkiraan

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

9. KESIMPULAN

Dari uraian di atas dapat ditarik beberapa kesimpulan :

1. Metode function point dapat dijadikan salah satu alternatif untuk menghitung volume software berdasarkan kompleksitasnya.
2. Penggunaan metode function point memerlukan campur tangan profesional yang berpengalaman karena perhitungannya sangat subyektif.
3. Karena perhitungannya hanyak berdasarkan pada gambaran pemrosesan data, metode function point harus pula didukung data-data tambahan untuk memperkuat perkiraan volume sistem software yang dihasilkan.

10. DAFTAR PUSTAKA

Caldiera, G., Antoniol, G., Fiuterm, R. dan Lokan, C., 1998, *Definition and Experimental Evaluation of Function Points for Object-Oriented Systems*, Proceedings of The Fifth International Software Metrics Symposium, California, US

Galin, Daniel, 2004, *Software Quality Assurance*, Pearson Education Limited, Addison Wesley,
Inggris

Gramus, D. dan Herron, D., 1996, *Measuring the Software Process – A Practical Guide to Functional Measurements*, Yourdon Press, Prentice Hall, New Jersey, US

IEEE, 2000, *IEEE Std 1061-1998 – Standard for Software Quality Metrics Methodology*, The Institure of Electrical and Electronics Engineers, New York, US.
Agustus 2009.