

# Manajemen Proyek Perangkat Lunak

Disiapkan oleh: Umi Proboyekti, S.Kom, MLIS

## Pengantar

Manajemen proyek perangkat lunak merupakan bagian yang penting dalam pembangunan perangkat lunak. Sekalipun tidak bersifat teknis seperti pengkodean, hal-hal dalam manajemen proyek PL ini mampu menentukan apakah proyek akan berjalan dengan baik sehingga menghasilkan produk yang baik. Hal-hal yang berkaitan dengan manajemen adalah pengelolaan personel dan koordinasi tim, proses, pengukuran proyek-termasuk menentukan harga dari PL, penjadwalan dan sebagainya. Dalam pembahasan berikut, hanya sebagian kecil dari manajemen yang akan dibahas untuk memberi gambaran tentang hal-hal manajemen yang berlaku dan diterapkan dalam pembangunan PL.

## Manajemen Personel, Produk dan Proses

Manajemen proyek perangkat lunak mengatur 4 hal penting: personel, produk, proses dan proyek. Empat hal ini berurutan mulai dari yang paling penting. Personel merupakan mendapat tempat paling penting karena tanpa personel yang baik dan tepat maka 3 hal lain tidak bisa berjalan dengan baik.

### Kategori Personel

Proses pembangunan PL melibatkan banyak personel. Personel-personel ini digambarkan seperti pemain, dan dikategorikan dalam 5 kategori pemain:

1. manajer senior : yang menentukan usaha yang dikerjakan, dan pemegang keputusan dalam proyek.
2. manajer proyek (teknis)– pemimpin tim: yang membuat rencana, memotivasi, mengatur dan mengendalikan praktisi yang mengerjakan PL
3. praktisi : yang mengerjakan PL
4. klien : yang menentukan kebutuhan PL dan pihak lain yang berkaitan dengan hasil produk
5. pengguna PL : yang berinteraksi langsung dengan PL yang dibangun.

Efektifitas kerja masing-masing personel di atas harus diusahakan oleh pemimpin tim. Pemimpin tim ini yang mengatur tim proyek agar dapat memberikan yang terbaik dari masing-masing personel.

### Pemimpin Tim

Pemimpin Tim PL disini adalah manager proyek. Seorang pemimpin tim diharuskan mempunyai ketrampilan memimpin yang cukup. Seseorang tidak menjadi pemimpin tim secara kebetulan tapi sungguh-sungguh karena punya kemampuan. Kemampuan yang dibutuhkan dalam kepemimpinan seperti:

- mampu memotivasi

- mampu berorganisasi : mengatur proses yang ada atau membuat yang baru dalam rangka mewujudkan ide/konsep menjadi produk
- mampu mendorong keluarnya ide-ide baru: memberi dorongan, menciptakan situasi yang kondusif untuk lahirnya ide baru
- mencari penyelesaian masalah (problem solving): mampu menganalisa masalah-masalah teknis ataupun manajemen/organisasi kemudian mendapatkan jalan keluar atau memotivasi anggota untuk mampu menyelesaikan masalah. Akomodatif terhadap perubahan yang mungkin terjadi
- mampu menjadi manajer: menggunakan wewenangnya pada saat yang tepat, atau memberikan kebebasan pada anggota timnya jika diperlukan
- mampu menghargai kerja: menghargai hasil yang dicapai, ide yang dilontarkan dan pendapat yang diajukan oleh anggota timnya
- mampu mengenali tim: mampu "membaca" dan memahami anggota timnya. Mampu memenuhi kebutuhan tim dan bertahan dalam tekanan yang tinggi.

### **Tim Perangkat Lunak (Software Team)**

Struktur organisasi dalam tim ini bisa mengadaptasi dari banyak struktur organisasi yang sudah ada. Berikut beberapa pilihan pembagian tugas/penugasan yang bisa diterapkan untuk tim perangkat lunak yang terdiri dari  $n$  personel yang bekerja selama  $k$  tahun:

- $n$  personel ditugaskan untuk sejumlah  $m$  tugas yang berbeda dengan sedikit tugas gabungan → koordinasi adalah tugas dari manajer yang mungkin saja punya 6 proyek lainnya.
- $n$  personel di tugaskan untuk sejumlah  $m$  tugas yang berbeda dengan  $m < n$  sehingga terbentuk tim informal. Pemimpin tim khusus perlu ada → koordinasi antar tim adalah tanggung jawab manajer
- $n$  personel dibagi menjadi sejumlah  $t$  tim. Tiap tim ditugaskan mengerjakan satu atau lebih tugas. Tiap tugas mempunyai struktur yang ditentukan sebelumnya bagi semua tim → koordinasi dikendalikan oleh tim dan manager

Sekalipun masing-masing pilihan punya argumentasi sendiri-sendiri, namun dari pengamatan yang dilakukan, pilihan no 3 dianggap lebih produktif.

Cara atau gaya manajemen, jumlah personel, tingkat kemampuan para personel dan masalah-masalah yang dihadapi tim menentukan bentuk struktur organisasi yang bisa diterapkan. Contoh struktur organisasi tim adalah:

1. **Democratic Decentralized (DD)** : Tidak ada pemimpin yang permanen, koordinator ditunjuk untuk jangka waktu yang pendek, keputusan diambil

berdasarkan konsensus bersama, komunikasi horizontal antar anggota tim (posisi sejajar semua) → cocok untuk masalah yang sulit/rumit, cocok untuk proyek besar, tim cenderung awet dan bertahan lama, pekerjaan memuaskan, cocok untuk masalah yang modularitasnya rendah, perlu banyak waktu untuk menyelesaikan proyek,

2. **Controlled decentralized (CD)** : Pemimpin tim ditentukan, ada wakil pemimpin dan mereka berbagi tugas, penyelesaian masalah adalah tugas tim dan implementasinya dibagi di antara beberapa sub-tim oleh pemimpin, komunikasi horizontal di antara sub-tim dan di antara personel, komunikasi vertikal berdasarkan struktur hirarki → sentralisasi untuk penyelesaian masalah, cocok untuk masalah yang sederhana, cukup cocok untuk proyek besar, masalah dengan modularitas tinggi, menghasilkan sedikit kesalahan
3. **Controlled Centralized (CC)**: penyelesaian masalah dikerjakan oleh pemimpin, pemimpin melakukan koordinasi internal tim, komunikasi lebih banyak vertikal antara pemimpin dan anggota tim → cocok untuk masalah yang sederhana, melakukan penyelesaian, masalah lebih cepat, masalah dengan modularitas tinggi, menghasilkan sedikit kesalahan

### **Pengukuran PL**

Metric dalam software engineering didefinisikan oleh IEEE Glossary of SE sebagai " a quantitative measure of the degree to which a system, component, or process possesses a given attribute" atau artinya pengukuran secara kuantitatif pada tingkat sistem, komponen atau proses berdasarkan katagori yang ditetapkan.

a. pengukuran berdasarkan ukuran

Pengukuran berdasarkan PL-PL yang sudah diproduksi/dibuat sebelumnya, lengkap dengan karakteristik lain seperti line of code (LOC), harga, waktu yang diperlukan pada tiap fungsi atau proyek yang dibangun, kesalahan (error) yang ditemukan. Dari total LOC, harga dan lama waktu dapat diperoleh misalnya :

- harga per KLOC (seribu baris kode)
- kesalahan per KLOC

Functions	estimated LOC	LOC/pm	\$/LOC	Cost	Effort (months)
UICF	2340	315	14	32,000	7.4
2DGA	5380	220	20	107,000	24.4
3DGA	6800	220	20	136,000	30.9
DSM	3350	240	18	60,000	13.9
CGDF	4950	200	22	109,000	24.7
PCF	2140	140	28	60,000	15.2
DAM	8400	300	18	151,000	28.0
<b>Totals</b>	<b>33,360</b>			<b>655,000</b>	<b>145.0</b>

Gambar 1 : Contoh tabel pengukuran berdasarkan ukuran

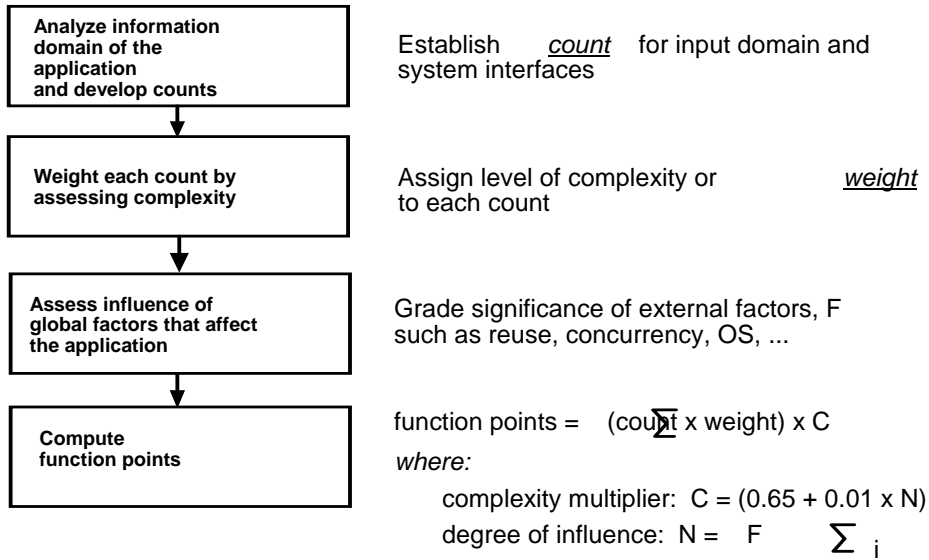
Cara ini kurang diterima secara universal karena penggunaan LOC untuk kunci ukuran bergantung pada bahasa pemrograman yang digunakan.

b. pengukuran berdasarkan fungsi (Function Point – FP)

Function point ditentukan berdasarkan bagian-bagian software yang bisa dihitung seperti :

- jumlah input dari pengguna
- jumlah output untuk pengguna
- jumlah user *inquiry*: inquiry didefinisikan sebagai online input yang menghasilkan respon langsung dari software dalam bentuk online output
- jumlah file: baik file yang terpisah dari database, atau bagian dari file
- jumlah external interface: misalnya data file pada storage media yang digunakan untuk mengirimkan informasi ke sistem lain.

Gambar 2 menggambarkan proses penghitungan Function Point. Yang Kurang jelas dalam proses ini dan kurand detil adalah bagaimana menentukan berat (weight)



Gambar 2: Alur pengukuran dengan Function Point

<u>measurement parameter</u>	<u>count</u>	<u>weighting factor</u>			=	
		<u>simple</u>	<u>avg.</u>	<u>complex</u>		
number of user inputs	<input type="text"/>	X 3	4	6	=	<input type="text"/>
number of user outputs	<input type="text"/>	X 4	5	7	=	<input type="text"/>
number of user inquiries	<input type="text"/>	X 3	4	6	=	<input type="text"/>
number of files	<input type="text"/>	X 7	10	15	=	<input type="text"/>
number of ext.interfaces	<input type="text"/>	X 5	7	10	=	<input type="text"/>
count-total	→					<input type="text"/>
complexity multiplier						<input type="text"/>
function points	→					<input type="text"/>

Gambar 3: Penghitungan Function Point

Gambar 3 menjelaskan contoh penghitungan Function point berdasarkan parameter yang sudah ditentukan.

Factors are rated on a scale of 0 (not important) to 5 (very important):

data communications	on-line update
distributed functions	complex processing
heavily used configuration	installation ease
transaction rate	operational ease
on-line data entry	multiple sites
end user efficiency	facilitate change

Gambar 4: Faktor-faktor yang dianggap penting

c. ukuran untuk organisasi kecil (DRE = Defect Removal efficiency)

Untuk organisasi yang kecil mungkin bisa menggunakan ukuran seperti :

- waktu (hari atau jam) mulai dari permintaan/request samai evaluasi lengkap  $\rightarrow t_{\text{queue}}$
- usaha (personel-waktu) untuk melakukan evaluasi  $\rightarrow W_{\text{eval}}$
- waktu (jam atau hari) dari selesainya evaluasi sampai penugasan lain ke personel  $\rightarrow t_{\text{eval}}$
- usaha (personel – jam) yang dibutuhkan untuk membuat perubahan  $\rightarrow W_{\text{change}}$
- waktu (jam atau hari ) untuk melakukan perubahan,  $\rightarrow t_{\text{change}}$
- kesalahan yang terjadi selama pengerjaan untuk melakukan perubahan  $\rightarrow E_{\text{change}}$
- cacat yang terjadi setelah perubahan diserahkan ke klien  $\rightarrow D_{\text{change}}$

Setelah ukuran-ukuran tersebut dikumpulkan bisa beberapa hal bisa dihitung seperti total waktu dari permintaan perubahan sampai implementasi dari perubahan. Persentase usaha yang dibutuhkan untuk evvaluasi dan implementasi bisa ditetapkan. Defect Removal Efficiency (DRE) bisa dihitung dengan:  $DRE = E_{\text{change}} / (E_{\text{change}} + D_{\text{change}})$ .

Diadaptasi dari:

1. Pressman, Roger.S. "Software Engineering : A Practioner's Approach." 5th . McGrawHill. 2001.