Topik 2 : Perangkat Lunak

2.1 Produk Perangkat Lunak

Tujuan utama rekayasa perangkat lunak adalah menghasilkan suatu produk perangkat lunak. Produk Perangkat Lunak menurut Ian Sommerville [SOM00] didefinisikan sebagai berikut:

> *Software Products are Software Systems delivered to a customer with the documentation which describes how to install and use the system.*

Produk perangkat lunak adalah sistem perangkat lunak beserta dokumentasinya yang menjelaskan prosedur penyiapan dan penggunaan perangkat lunak tersebut.

Pada definisi lain yang dikutip oleh Pressman [PRE01] dalam bukunya Software Engineering A Practioner's Approach, perangkat lunak didefinisikan lebih riinci lagi yaitu sebagai:

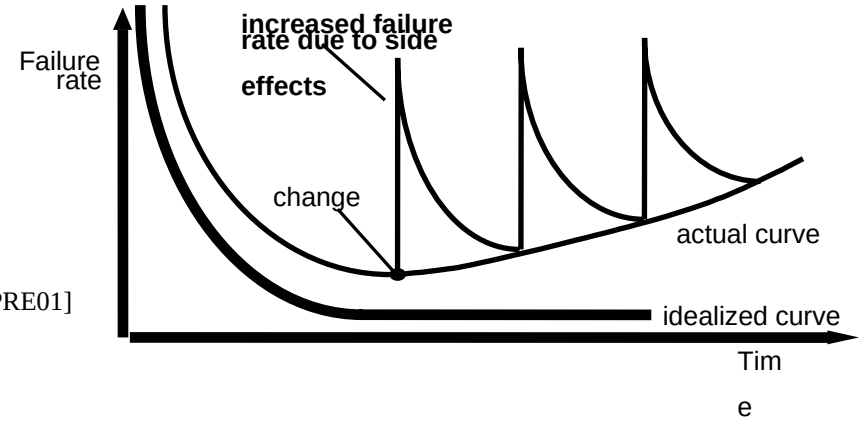
a. instruksi-instruksi yang jika dieksekusi akan memberikan layanan-layanan atau fungsi seperti yang diinginkan
b. struktur data yang diperlukan oleh suatu program untuk memanipulasi informasi
c. dokumen-dokumen yang mendeskripsikan penggunaan suatu program.

2.2 Karakteristik Perangkat Lunak

Menurut Pressman [PRE01], ada beberapa karakteristik perangkat lunak yang membedakan dengan perangkat keras :

Karakteristik pertama *: Software is developed or engineered, it is not manufactured in the classical sense.*
Perangkat lunak adalah suatu produk yang lebih menekankan pada kegiatan rekayasa (*engineering)* dibandingkan kegiatan *manufacturing* (rancang bangun di pabrik). rumit.

Karakteristik kedua *: Software doesn't "wear out"*
Perangkat lunak bukanlah produk yang dapat usang atau rusak untuk kemudian dibuang, seperti halnya produk perangkat keras. Yang dapat terjadi adalah produk-produk perangkat lunak tersebut tidak dapat melayani beberapa kebutuhan yang dikehendaki pemakainya, disebabkan berkembangnya kebutuhan-kebutuhan baru. Sehingga perlu dilakukan perubahan-perubahan pada perangkat lunak tersebut seperti yang ditunjukkan Gambar 2.4



Sumber : [PRE01]

Gambar 2.4 Kondisi Aktual dan Ideal Perangkat Lunak [PRE01]

Karakteristik ketiga : *Most software is custom-built rather than being assembled from existing components*
Kebanyakan perangkat lunak tidak dibangun dari perangkat lunak-perangkat lunak yang sudah ada. Pembangunan aplikasi baru kebanyakan dimulai dari awal, dari tahap analisis sampai tahap pengujian. Namun demikian, kini paradigma baru mulai dikembangkan, yaitu konsep *reuseability*. Dengan konsep ini suatu aplikasi baru dapat dikembangkan dari aplikasi yang sudah ada yang menerapkan konsep *reusability* tersebut.

2.3 Kualitas Perangkat Lunak

Ukuran kualitas perangkat lunak dilihat dari atribut antara lain :

- *Maintainability* , yaitu tingkat kemudahan perangkat lunak tersebut dalam mengakomodasi perubahan-perubahan
- *Dependability*, ketidakbergantungan perangkat lunak dengan elemen-elemen sistem lainnya atau sistem secara keseluruhan.Artinya kegagalan elemen lain tidak mempengaruhi performansi perangkat lunak
- *Efficiency* , menyangkut waktu eksekusi
- *Usability* , yaitu atribut yang menunjukkan tingkat kemudahan pengoperasian perangkat lunak

2.4 Aplikasi Perangkat Lunak

Pressman [PRE01] mendefinisikan aplikasi perangkat lunak sebagai berikut:

Perangkat lunak sistem
Suatu perangkat lunak yang berfungsi melayani perangkat lunak lain, seperti : kompilator, editor, sistem operasi, utilitas, da lain-lain.

Perangkat lunak Real time (waktu nyata)
Suatu perangkat lunak yang berfungsi mengendalikan, memonitor atau menganalisis kejadian (event) yang terjadi pada keadaan nyata. Perangkat lunak ini mempunyai komponen sebagai berikut :

- komponen pengumpul data : mengumpulkan & memformat informasi dari lingkungan eksternal
- komponen analisis: melakukan transformasi informasi yang dibutuhkan aplikasi
- komponen control/output: merespon lingkungan eksternal
- komponen pemonotoran : mengkoordinasi seluruh komponen sehingga respon real time yang diinginkan dapat tercapai

Perangkat lunak bisnis
Perangkat lunak yang mengakses satu atau lebih basisdata besar yang berisi informasi bisnis, sebagai contoh sistem *payroll*, sistem invetori, dan lain-lain.

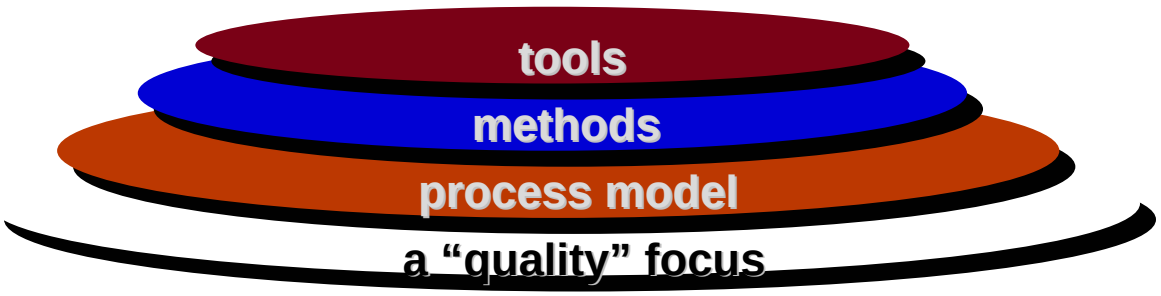
Perangkat lunak keteknikan dan keilmuan
Perangkat lunak ini banyak membantu memecahkan permasalahan di bidang astronomi, vulkanologi, automotive stress analysis, molecular biology, automotive manufacturing, Computer Aided Design (CAD), dan lain-lain

Embedded software
Perangkat lunak yang ditanam pada suatu *chip* (EEPROM). Perangkat lunak ini terintegrasi dengan perangkat keras dan berfungsi mengatur kinerja dari perangkat keras tersebut. Sebagai contoh : microwave oven, telpon genggam, pengaturan avionik pesawat udara, dan lain-lain.
Perangkat lunak Komputer personal (PC)
Perangkat lunak yang dioperasikan di PC, seperti : pengolah kata, multimedia, DBMS.

Perangkat lunak Kecerdasan Buatan
Perangkat lunak yang menerapkan algoritma nonnumerik untuk memecahkan permasalahan yang kompleks. Contoh : perangkat lunak kecerdasan buatan, sistem pakar, dan lain-lain.

2.5 PROSES PERANGKAT LUNAK

Proses perangkat lunak adalah sebuah kerangka kerja untuk membangun perangkat lunak yang **berkualitas tinggi.** Gambar 2.5 menunjukkan lapisan teknologi pada rekayasa Perangkat lunak.



Gambar 2.5 Lapisan-lapisan Rekayasa Perangkat Lunak [PRE01]

Dari Gambar 2.2 tersebut dapat dilihat bahwa tujuan utama rekayasa perangkat lunak adalah pencapaian kualitas (**"Quality Focus")**. Kualitas ini diterjemahkan ke dalam ukuran-ukuran (*metrics*), meliputi *maintaiability, dependability, usability,* dan *efificientcy* yang sudah diterangkan di atas.
.
Proses : mendefinisikan kerangka kerja (*frame work*) , sehingga pembangunan perangkat lunak dapat dilakukan secara sistematis.

Metode : mendefinisikan bagaimana perangkat lunak dibangun, meliputi metode-metode yang digunakan dalam melakukan analisis kebutuhan, perancangan, implementasi dan pengujian. Sebagai contoh : metode terstruktur, metode berorientasi objek, dan lain-lain.

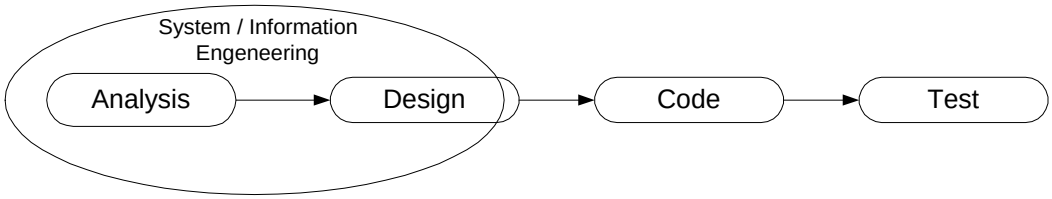
Alat Bantu : perangkat yang bersifat otomatis maupun semi otomatis yang berfungsi mendukung tiap tahap pembangunan perangkat lunak. Contoh : CASE, CAD, dan lain-lain.

2.6 MODEL PROSES

Fungsi utama model proses pengembangan perangkat lunak adalah :
- menentukan tahap-tahap yang diperlukan untuk pengembangan perangkat lunak.

- menentukan urutan pelaksanaan dari tahap-tahap tersebut dalam rangka pengembangan perangkat lunak.
- menentukan kriteria transisi/perpindahan dari satu tahap ke tahap berikutnya.

a. Model Linier Sekuensial



Gambar 2.6 Model Linier Sekuensial [PRE01]

Analisis : merupakan tahap untuk menganalisis hal-hal yang diperlukan dalam pelaksanaan pembangunan perangkat lunak . Hasil analisis didokumentasikan dan dikaji ulang oleh *customer*.

Perancangan (*Design*) : tahap ini merupakan tahap penerjemahan dari kebutuhan fungsional dan data yang telah dianalisis ke dalam bentuk yang mudah untuk dimengerti oleh *programmer*

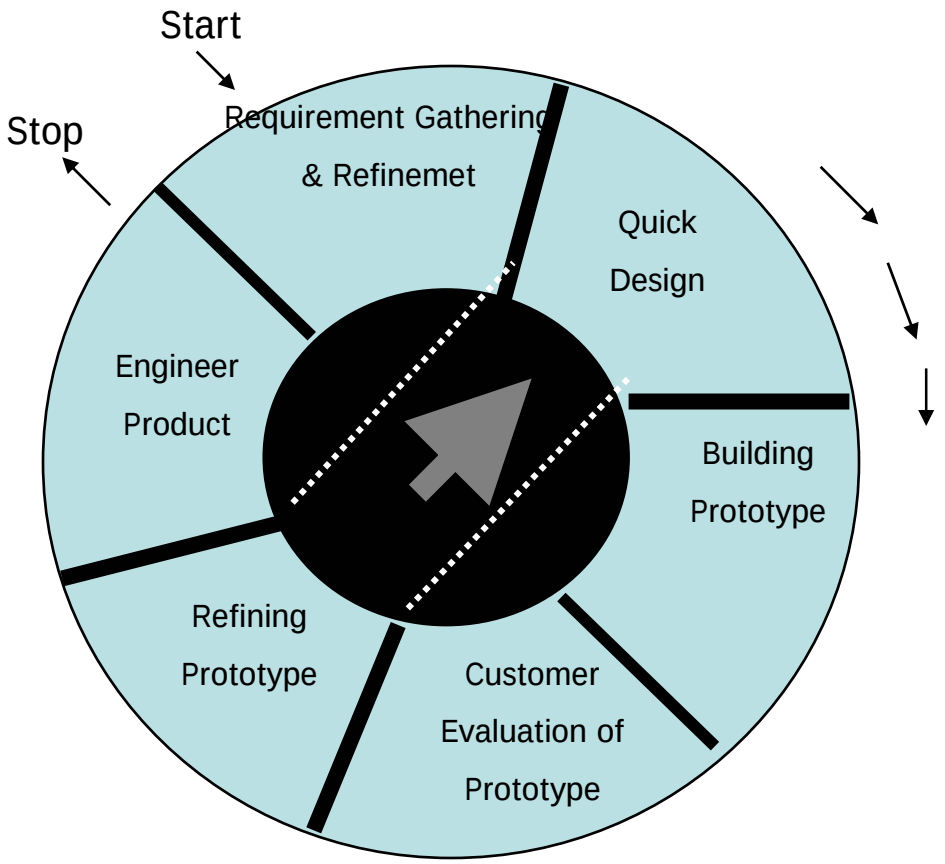
Implementasi (*Code Generation*) : mengimplementasikan hasil rancangan ke dalam bahasa pemrogramamn komputer yang telah ditentukan

Pengujian (*Testing*) : Uji coba perangkat lunak yang terfokus pada logika internal dari perangkat lunak dan kesesuaian perangkat lunak yang dibangun dengan kebutuhan fungsional yang didefinisikan.

Model proses ini sering disebut juga dengan model proses *waterfall*. Tiap tahap pada model proses ini diakhir dengan dokumentasi. Oleh karena itu model ini sering juga disebut model *Document Driven Software Process.*

b. Model Prototyping

Tidak semua *customer* mampu mendefinisikan kebutuhannya secara detil. Oleh karena itu diperlukan *prototype* untuk mengeksplorasi kebutuhan *customer*. Dengan melihat *prototype customer* dapat mengevaluasinya dan memberikan masukan-masukan kepada pengembang sehingga *prototype* yang dibangun semakin mendekati kebutuhan yang sebenarnya. Gambar 2.7 menunjukkan model prototyping.



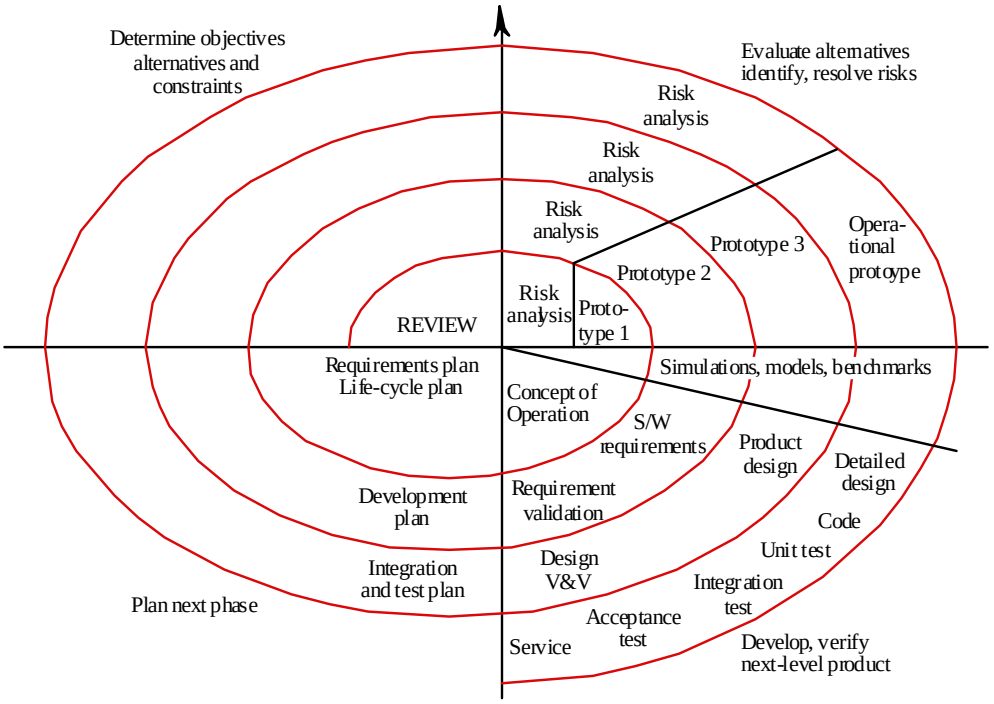
Gambar 2.7 Model Prototyping

Model ini sering dikenal sebagai "*Code Driven Software Process*"dan merupakan salah satu proses perangkat lunak yang mulai banyak digunakan saat ini. Model Prototyping banyak memanfaatkan 4GL dan *Application Generator*.

Dibandingkan dengan model linier sekuensial model ini memiliki produktivitas lebih baik namun kelengkapan fungsi dari sistem dan keterpaduan (integrasi) sistem kurang baik.

c. Model Spiral

Model ini sering dikenal sebagai *Risk Driven Software Process*. Model ini sesuai untuk pengembangan proyek yang berskala besar, dengan memperhatikan pengaruh resiko dilihat dari segi finansial maupun keamanan (jiwa manusia). Model ini merupakan kombinasi linier sekuensial, *Prototyping* dan *Risk Analysis*. Pada tiap akhir tahap dibuat dokumen hasil analisis resiko. Gambar 2.8 menunjukkan model proses Spiral.



Gambar 2.8 Model Proses Spiral

Beberapa model proses lain antara lain Inkremental, Rapid Application Development dan 4GT . Silakan anda eskplorasi sendiri !