

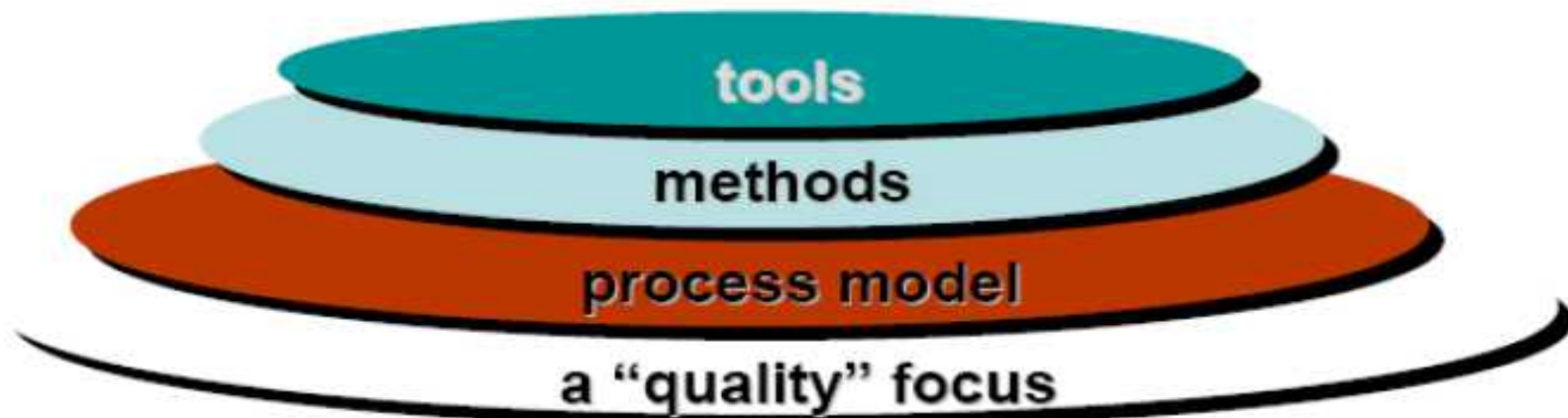
# **BAB 3.**

## **MACAM DARI SIKLUS HIDUP PERANGKAT LUNAK (SWDLC/SOFTWARE DEVELOPMENT LIFE CYCLE)**

**Disusun Oleh :  
Elisawati, M.Kom**

# TEKNOLOGI BERLAPIS PENGEMBANGAN PERANGKAT LUNAK

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



Lapisan-lapisan Rekayasa Perangkat Lunak

- Dari Gambar tersebut dapat dilihat bahwa tujuan utama rekayasa perangkat lunak adalah pencapaian kualitas ( "Quality Focus"). Kualitas ini diterjemahkan ke dalam ukuran-ukuran (metrics), meliputi maintainability, dependability, usability, dan eficiency 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.



# DEFINISI SOFTWARE PROCESS

Software process dapat didefinisikan sebagai berikut :

- Merupakan suatu deskripsi proses yang dijadikan panduan kerja bagi para software engineer dengan memetakan peran dan tanggung jawab mereka masing-masing.
- Merupakan sekumpulan aktifitas yang ditujukan untuk melakukan pengembangan maupun evolusi software.
- Merupakan suatu urutan langkah yang diperlukan dalam pengembangan atau pemeliharaan software.



- Merupakan kerangka teknis dan manajemen untuk menerapkan metode, alat bantu(tool) serta komponen SDM dalam pengerjaan software.
- (Menurut Ian Sommerville) : merupakan sekumpulan proses dan hasil yang terkait dengan proses tersebut dalam rangka pengembangan produk software.



# AKTIFITAS UMUM DALAM SOFTWARE PROCESS

Untuk mengembangkan perangkat lunak secara memadai, proses pengembangan perangkat lunak harus didefinisikan terlebih dahulu. Usaha yang berhubungan dengan rekayasa perangkat lunak dapat dikategorikan ke dalam beberapa aktifitas dengan tanpa mempedulikan area aplikasi, ukuran proyek atau kompleksitasnya. Berikut adalah aktifitas umum yang terdapat dalam proses pengembangan perangkat lunak :



## 1. Requirement

Merupakan aktifitas dimana didefinisikan mengenai "apa" (what) yang akan dibangun terkait dengan produk perangkat lunak yang akan dihasilkan. Kebutuhan dari persepsi pelanggan (requirement) didefinisikan dan disepakati. Dari aktifitas ini akan diperoleh pernyataan global mengenai kegunaan sistem serta ketersediaan sumber daya yang akan mendukung pembangunan sistem seperti : kebutuhan sumber daya waktu, biaya dan tenaga (manusia).





## 2. Specification

Merupakan aktifitas dimana kebutuhan pelanggan (requirement) yang telah ditetapkan ditransformasikan ke dalam kebutuhan sistem. Dari aktifitas ini akan diperoleh spesifikasi detil mengenai produk perangkat lunak yang akan dibangun antara lain seputar fungsionalitasnya (mengidentifikasi informasi apa yang akan diproses, fungsi dan unjuk kerja apa yang dibutuhkan, tingkah laku sistem seperti apa yang diharapkan), kebutuhan perangkat keras dan perangkat lunak pendukung dalam pembangunannya, dan lain-lain.



### 3. Design


Merupakan aktifitas dimana hasil analisis kebutuhan dan spesifikasi sistem dibentuk dalam suatu model. Pengembang harus mendefinisikan bagaimana data dikonstruksikan, bagaimana fungsi-fungsi diimplementasikan sebagai sebuah arsitektur perangkat lunak, bagaimana detail prosedur akan diimplementasikan, bagaimana interface ditandai (dikarakterisasi). Dalam membuat pemodelan, pengembang dapat menggambarkan pemodelan berdasarkan perilaku sistem ataupun secara struktural. Dari aktifitas ini akan diperoleh penggambaran sistem dalam bentuk model semacam use case diagram, data flow diagram, sequence diagram, entity relationship diagram, dan lain-lain.

## 4. Code

Merupakan aktifitas dimana hasil rancangan (model) dari tahapan sebelumnya diterjemahkan dalam bentuk coding program pada sebuah bahasa pemrograman.

## 5. Test (verification & validation)

Setelah rancangan diterjemahkan ke dalam bentuk coding program selanjutnya akan dilakukan proses pengujian untuk memastikan apakah aplikasi yang dibangun sudah sesuai dengan spesifikasi dan kebutuhan pelanggan yang ditahapan awal ditetapkan.



## 6. Debug

Merupakan tahapan dimana akan dilakukan proses perbaikan yang diperlukan apabila pada fase pengujian masih ditemukan adanya kesalahan.

## 7. Maintenance


Aktifitas ini berfokus pada perubahan (change), yang dihubungkan dengan koreksi kesalahan, penyesuaian yang dibutuhkan ketika lingkungan perangkat lunak berkembang, serta perubahan sehubungan dengan perkembangan yang disebabkan oleh perubahan kebutuhan pelanggan. Fase pemeliharaan mengaplikasikan lagi langkah-langkah pada fase definisi dan fase pengembangan, tetapi semuanya tetap bergantung pada konteks perangkat lunak yang ada. Ada empat tipe perubahan yang terjadi selama masa fase pengembangan yaitu :

### a. Koreksi (corrective)

Meskipun dengan jaminan kualitas yang terbaik, sepertinya pelanggan akan tetap menemukan cacat pada perangkat lunak. Pemeliharaan korektif mengubah perangkat lunak, membetulkan cacat atau rusak.

### b. Adaptasi (adaptive)

Dari waktu ke waktu, lingkungan original (contohnya CPU, sistem operasi, aturan-aturan bisnis, karakterisasi produk eksternal) dimana perangkat lunak dikembangkan akan terus berubah. Pemeliharaan adaptif menghasilkan modifikasi kepada perangkat lunak untuk mengakomodasi perubahan pada kebutuhan fungsional original.



### c. Pengembangan (perfective)

Ketika perangkat lunak dipakai, pelanggan akan mengenali fungsi-sungsi tambahan yang memberi mereka keuntungan. Pemeliharaan perfektif memperluas perangkat lunak sehingga melampaui kebutuhan fungsi originalnya.

### d. Pencegahan (preventive)

Keadaan perangkat lunak semakin memburuk sehubungan dengan waktu, dan karena itu preventive maintenance yang sering juga disebut software engineering (rekayasa perangkat lunak), harus dilakukan untuk memungkinkan perangkat lunak melayani kebutuhan para pemakainya. Pada dasarnya preventive maintenance melakukan perubahan pada program komputer sehingga bisa menjadi lebih mudah untuk dikoreksi, disesuaikan dan dikembangkan.

Aktifitas dasar dalam software process menurut Ian Sommerville antara lain :

### 1. Specification

Aktifitas dimana pelanggan dan pengembang membuat definisi mengenai software yang akan dibangun dan juga batasan pada pengembangannya.

### 2. Development

Aktifitas dimana software diproduksi ( didisain dan diprogram)

### 3. Validation

Aktifitas dimana dilakukan proses pengecekan apakah software yang dibangun sudah sesuai dengan keinginan pelanggan.



## 4. Evolution

Aktifitas dimana dilakukan perubahan/modifikasi terhadap software untuk diadaptasikan terhadap perubahan kebutuhan dari pelanggan.





Aktifitas atau langkah-langkah yang dijabarkan dalam pengembangan perangkat lunak tersebut harus diimbangi dengan sejumlah aktifitas pelindung (umbrella activities). Kegiatan-kegiatan khusus di dalam kategori ini menyangkut :

- Manajemen proyek PL : melindungi agar PL yang ada hasilnya bagus
- Formal technical review, contoh : menemui user dan mengecek kebutuhannya untuk analisis. Jika tidak dilakukan, nanti kita buat PL yang sesuai pikiran kita, bukan sesuai dengan kebutuhan user.
- Software quality assurance (jaminan kualitas PL) : langkah supaya PL berkualitas

- Manajemen konfigurasi PL : bagaimana PL bisa dikonfigurasi atau dibuat
- Pembuatan dan penyiapan dokumen : sebagai senjata jika user tiba-tiba meminta tambahan fungsi PL
- Reusability management (manajemen reusabilitas): komponen PL bisa dipakai ulang
- Measurement (pengukuran) : harus ada dalam setiap tahap
- Risk Management (manajemen resiko) : risiko harus diantisipasi supaya tidak gagal
- Aktivitas pelindung diaplikasikan ke seluruh proses perangkat lunak

# KARAKTERISTIK SOFTWARE PROCESS YANG BAIK

- Understandability

Proses secara eksplisit didefinisikan sehingga mudah dipahami bagi siapapun yang terlibat di dalam proses pengembangan.

- Visibility

Aktifitas proses memberi hasil yang jelas sehingga kemajuan proses dapat terlihat dari luar pihak pengembang.

- Supportability

Proses dapat didukung oleh teknologi semacam CASE tools.

- Acceptability

Penerimaan atas proses yang terdefinisi dan yang digunakan oleh software engineer selama pembangunan produk perangkat lunak.

- Reliability

Proses didisain dengan suatu metode untuk menghindari dari kesalahan, dan apabila ada kesalahan dapat terdeteksi sedini mungkin sebelum mengakibatkan cacat pada produk akhir.

- Robustness

Proses dapat terus dilanjutkan meskipun terdapat masalah yang tidak diharapkan muncul.

- Maintainability

Proses dapat mengadaptasi terhadap permintaan perubahan ataupun perbaikan

- Rapidity

Proses dapat diselesaikan dalam waktu yang relatif cepat



# MODEL PROSES RPL

Model proses disebut juga dengan aliran kerja (workflow), yakni tata cara bagaimana elemen-elemen proses berhubungan satu dengan lainnya. Aliran kerja ini dapat juga disebut dengan siklus hidup (life-cycle) sistem yang dimulai dari sejak sistem diajukan untuk dibangun hingga saat ia ditarik dari peredaran.

Fungsi utama model proses pengembangan perangkat lunak adalah :

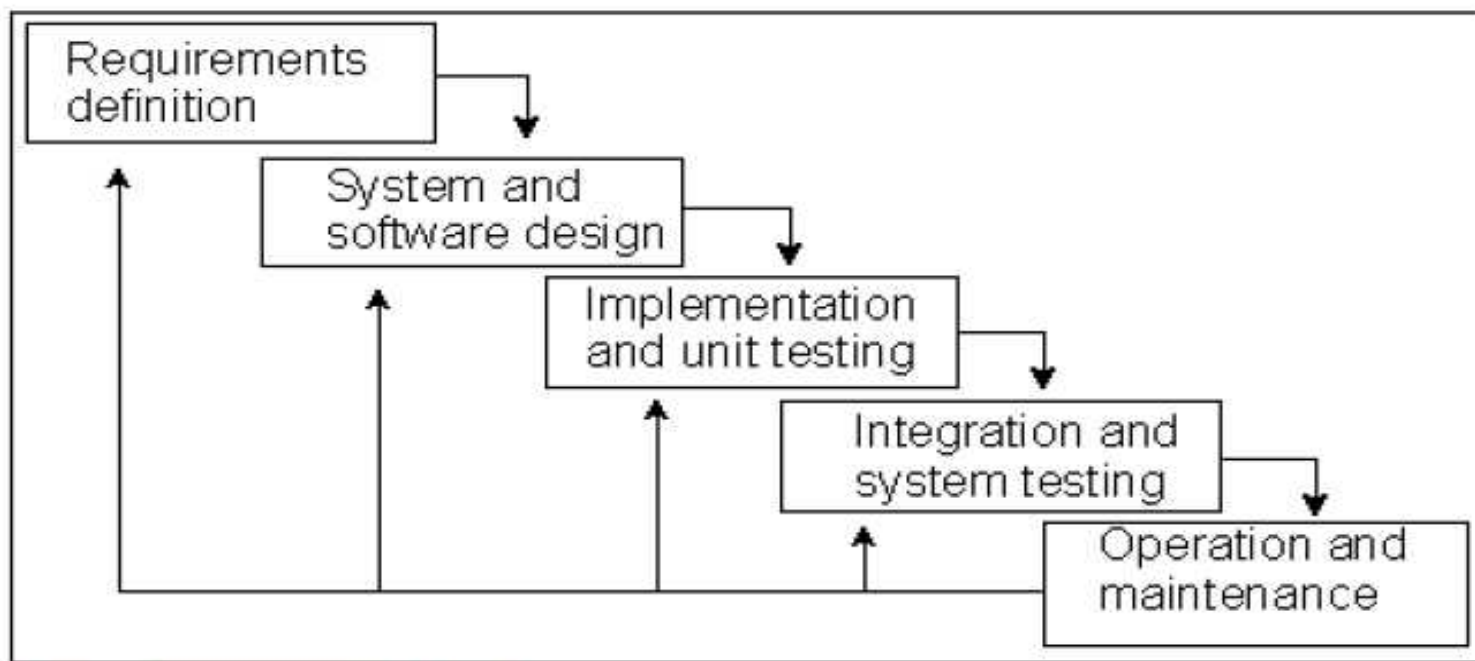
- menentukan tahap-tahap yang diperlukan untuk pengembangan perangkat lunak.
- menentukan urutan pelaksanaan dari tahap-tahap tersebut dalam rangka pengembangan perangkat lunak.
- menentukan kriteria transisi/perpindahan dari satu tahap ke tahap berikutnya.

Untuk menentukan mana model yang terbaik, kita harus tahu apa kelebihan dan kekurangan model-model proses tersebut. Akan tetapi, model proses biasanya bukan ditentukan mana yang terbaik atau tidak, tetapi ditentukan oleh karakteristik dari berbagai macam faktor, misalnya tim SE-nya, atau software-nya sendiri, waktu untuk melakukan SE, kebijakan-kebijakan dari perusahaan, dan sebagainya. Dengan menggunakan model proses yang terbaru pun, ketika diaplikasikan ke dalam perusahaan misalnya, tetapi kalau tim SE-nya tidak siap dengan kondisi yang mengharuskan menggunakan model proses tersebut, tentunya tidak mungkin bisa dilakukan. Kalaupun dipaksakan tentu saja hasilnya tidak akan maksimal. Akan tetapi di perusahaan lain, bisa jadi menerapkan model proses yang sama, tetapi hasilnya bagus, karena tim SE-nya siap atau scope software-nya berbeda.

# MODEL WATERFALL

Nama model ini sebenarnya adalah "Linear Sequential Model". Model ini sering disebut dengan "classic life cycle" atau model waterfall. Model ini adalah model yang muncul pertama kali yaitu sekitar tahun 1970 sehingga sering dianggap kuno, tetapi merupakan model yang paling banyak dipakai didalam Software Engineering (SE). Model ini melakukan pendekatan secara sistematis dan urut mulai dari level kebutuhan sistem lalu menuju ke tahap analisis, desain, coding, testing / verification, dan maintenance. Disebut dengan waterfall karena tahap demi tahap yang dilalui harus menunggu selesainya tahap sebelumnya dan berjalan berurutan. Sebagai contoh tahap desain harus menunggu selesainya tahap sebelumnya yaitu tahap requirement. Secara umum tahapan pada model waterfall (menurut Ian Sommerville) dapat dilihat pada gambar berikut :





Gambar di atas adalah tahapan umum dari model proses ini menurut Ian Sommerville. Penjelasan dari tiap tahapan tersebut adalah :

### 1. Requirements analysis and definition:

Di tahapan ini dilakukan Analisa kebutuhan

## 2. System and software design:

Pada tahapan ini, desain dikerjakan setelah kebutuhan selesai dikumpulkan secara lengkap.

## 3. Implementation and unit testing:

4. Desain program diterjemahkan ke dalam kode-kode dengan menggunakan bahasa pemrograman yang sudah ditentukan. Program yang dibangun langsung diuji baik secara unit.

## 5. Integration and system testing:

Penyatuan unit-unit program kemudian diuji secara keseluruhan (system testing).

## 6. Operation and maintenance:

Mengoperasikan program dilingkungannya dan melakukan pemeliharaan, seperti penyesuaian atau perubahan karena adaptasi dengan situasi sebenarnya

## SOAL :

1. Buatlah Kelompok dan cari model proses dari rekayasa perangkat lunak, Berikan penjelasan serta sebutkan kelebihan dan kelemahan dari setiap model tersebut.



## REFERENSI

- Roger S. Pressman, Software Engineering Apratitional Approach, edisi ketiga
- Simarmata Janner, Rekayasa Perangkat Lunak, Penerbit Andi, Yogyakarta, 2010
- Rosa A.S, M. Shalahuddin, Modul Pembelajaran Rekayasa Perangkat Lunak (Terstruktur dan Beroriented Objek), Penerbit Modula, Bandung, 2011

