REKAYASA PERANGKAT LUNAK
(SOFTWARE ENGINEERING)

I. PENDAHULUAN

Rekayasa perangkat lunak telah berkembang sejak pertama kali ddiciptakan pada tahun 1940-an hingga kini. Focus utama pengembangannya adalah untuk mengembangkan praktek dan teknologi untuk meningkatkan produktivitas para praktisi pengembang perangkat luank dan kualitas aplikasi yang dapat digunakan oleh pemakai.

I.1 Sejarah Software Engineering

Istilah software engineering digunakan pertama kali pada akhir 1950-an dan awal 1960-an. Saat itu, masih terdapat perdebatan tajam mengenai aspek engineering dari pengembangan perangkat lunak. Pada tahun 1968 dan 1969, komite sains NATO mensponsori dua konferensi tentang rekayasa perangkat lunak, yang memberikan dampak kuat terhadap pengembangan rekayasa perangkat lunak. Banyak yang menganggap dua konferensi inilah yang menandai awal resmi profesi rekayasa perangkat lunak.

Pada tahun 1960-an hingga 1980-an, banyak masalah yang ditemukan para praktisi pengembangan perangkat lunak. Banyak project yang gagal, hingga masa ini disebut sebagai krisis perangkat lunak. Kasus kegagalan pengembangan perangkat lunak terjadi mulai dari project yang melebihi anggaran, hingga kasusu yang mengakibatkan kerusakan fisik dan kematian. Salah satu kasus yang terkenal antara lain meledaknya roket Ariane akibat kegagalan perangkat lunak. Selama bertahun-tahun, para peneliti memfokuskan usahanay untuk menemukan teknik jitu untuk memecahkan masalah krisi perangkat lunak. Berbagai teknik, metode, alat, proses diciptakan dan diklaim sebagai senjata pamungkas untuk memecahkan kasus ini. Mulai dari pemrograman terstruktur, pemrograman berorientasi objek, pernagkat pembantu pengembangan perangkat lunak (CASE tools), berbagai standar, UML hingga metode formal diagung-agungkan sebagai senjaat pamungkas untuk menghasilkan software yang benar, sesuai anggaran dan tepat

waktu. Pada tahun 1987, Fred Brooks menulis artikel No Silver Bullet, yang berproposisi bahwa tidak ada satu teknologi atau praktek yang sanggup mencapai 10 kali lipat perbaikan dalam produktivitas pengembanan perngkat lunak dalam tempo 10 tahun.

Sebagian berpendapat, no silver bullet berarti profesi rekayasa perangkat lunak dianggap telah gagal. Namun sebagian yang lain justru beranggapan, hal ini menandakan bahwa bidang profesi rekayasa perangkat lunak telah cukup matang, karena dalam bidang profesi lainnya pun, tidak ada teknik pamungkas yang dapat digunakan dalam berbagai kondisi.

I.2 Pengertian Dasar

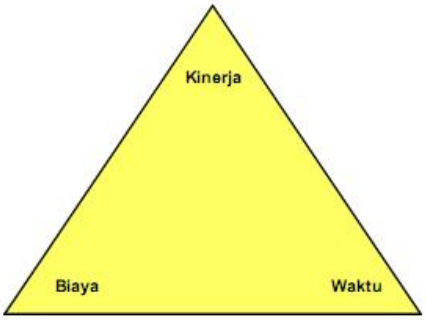
Istilah Reakayasa Perangkat Lunak (RPL) secara umum disepakati sebagai terjemahan dari istilah Software engineering. Istilah Software Engineering mulai dipopulerkan pada tahun 1968 pada software engineering Conference yang diselenggarakan oleh NATO. Sebagian orang mengartikan RPL hanya sebatas pada bagaimana membuat program komputer. Padahal ada perbedaan yang mendasar antara perangkat lunak (software) dan program komputer.

Perangkat lunak adalah seluruh perintah yang digunakan untuk memproses informasi. Perangkat lunak dapat berupa program atau prosedur. Program adalah kumpulan perintah yang dimengerti oleh komputer sedangkan prosedur adalah perintah yang dibutuhkan oleh pengguna dalam memproses informasi (O'Brien, 1999).

Pengertian RPL sendiri adalah suatu disiplin ilmu yang membahas semua aspek produksi perangkat lunak, mulai dari tahap awal yaitu analisa kebutuhan pengguna, menentukan spesifikasi dari kebutuhan pengguna, disain, pengkodean, pengujian sampai pemeliharaan sistem setelah digunakan. Dari pengertian ini jelaslah bahwa RPL tidak hanya berhubungan dengan cara pembuatan program komputer. Pernyataan "semua aspek produksi" pada pengertian di atas, mempunyai arti semnua hal yang berhubungan dengan proses produksi seperti manajemen proyek, penentuan personil, anggaran biaya, metode, jadwal, kualitas sampai dengan pelatihan pengguna merupakan bagian dari RPL.

II. TUJUAN REKAYASA PERANGKAT LUNAK

Secara umunmm tujuan RPL tidak berbeda dengan bidang rekayasa yang lain. Hal ini dapat kita lihat pada Gambar di bawah ini.



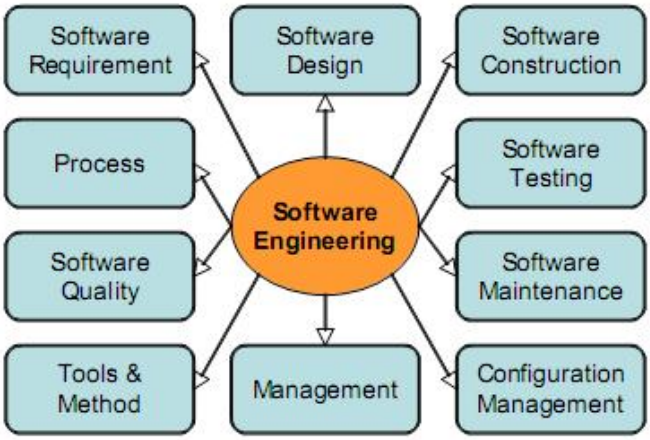
Gambar 1. Tujuan RPL

Dari Gambar di atas dapat diartikan bahwa bidang rekayasa akan selalu berusaha menghasilkan output yang kinerjanya tinggi, biaya rendah dan waktu penyelesaian yang tepat. Secara leboih khusus kita dapat menyatakan tujuan RPL adalah:

a. memperoleh biaya produksi perangkat lunak yang rendah
b. menghasilkan pereangkat lunak yang kinerjanya tinggi, andal dan tepat waktu
c. menghasilkan perangkat lunak yang dapat bekerja pada berbagai jenis platform
d. menghasilkan perangkat lunak yang biaya perawatannya rendah

III. RUANG LINGKUP

Sesuai dengan definisi yang telah disampaikan sebelumnya, maka ruang lingkup RPL dapat digambarkan sebagai berikut:



Gambar 2. Ruang lingkup RPL (Abran et.al., 2004).

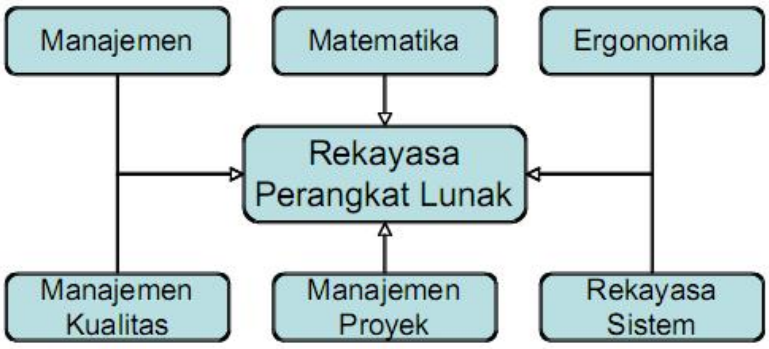
- software Requirements berhubungan dengan spesifikasi kebutuhan dan persyaratan perangkat lunak
- software desain mencakup proses penampilan arsitektur, komponen, antar muka, dan karakteristik lain dari perangkat lunak
- software construction berhubungan dengan detail pengembangan perangkat lunak, termasuk algoritma, pengkodean, pengujian dan pencarian kesalahan
- software testing meliputi pengujian pada keseluruhan perilaku perangkat lunak
- software maintenance mencakup upaya-upaya perawatan ketika perangkat lunak telah dioperasikan
- software configuration management berhubungan dengan usaha perubahan konfigurasi perangkat lunak untuk memenuhi kebutuhan tertentu
- software engineering management berkaitan dengan pengelolaan dan pengukuran RPL, termasuk perencanaan proyek perangkat lunak
- software engineering tools and methods mencakup kajian teoritis tentang alat bantu dan metode RPL

- software engineering process berhubungan dengan definisi, implementasi pengukuran, pengelolaan, perubahan dan perbaikan proses RPL
- software quality menitik beratkan pada kualitas dan daur hidup perangkat lunak

IV. REKAYASA PERANGKAT LUNAK DAN DISIPLIN ILMU LAIN

Cakupan ruang lingkup yang cukup luas, membuat RPL sangat terkait dengan disiplin dengan bidang ilmu lain. tidak saja sub bidang dalam disiplin ilmu komputer namun dengan beberapa disiplin ilmu lain diluar ilmu komputer.

Hubungan keterkaitan RPL dengan ilmu lain dapat dilihat pada gambar dibawah ini



Gambar 3. Keterkaitan RPL dengan bidang ilmu lain.

- bidang ilmu manajemen meliputi akuntansi, finansial, pemasaran, manajemen operasi, ekonomi, analisis kuantitatif, manajemen sumber daya manusia, kebijakan, dan strategi bisnis
- bidang ilmu matematika meliputi aljabar linier, kalkulus, peluang, statistik, analisis numerik, dan matematika diskrit
- bidang ilmu manajemen proyek meliputi semua hal yang berkaitan dengan proyek, seperti ruang lingkup proyek, anggaran, tenaga kerja, kualitas, manajemen resiko dan keandalan, perbaikan kualitas, dan metode-metode kuantitatif

- bidang ilmu ergonomika menyangkut hubungan (interaksi) antar manusia dengan komponen-komponen lain dalam sistem komputer
- bidang ilmu rekayasa sistem meliputi teori sistem, analisis biaya-keuntungan, pemodelan, simulasi, proses, dan operasi bisnis

V. PERKEMBANGAN REKAYASA PERANGKAT LUNAK

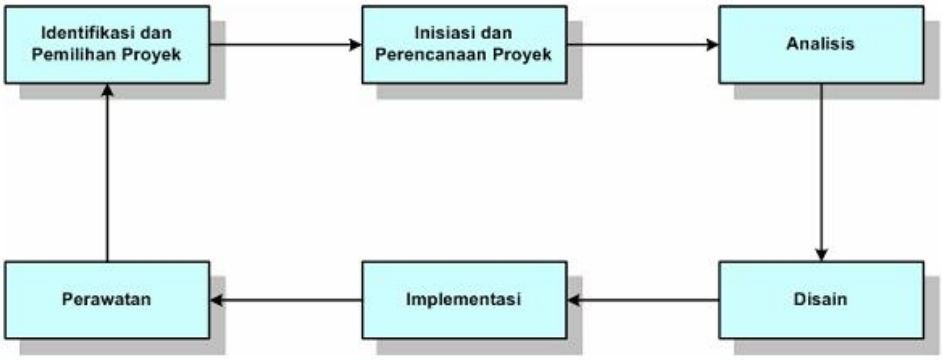
Meskipun baru dicetuskan pada tahun 1968, namun RPL telah memiliki sejarah yang cukup yang panjang. Dari sisi disiplin ilmu, RPL masih reklatif muda dan akan terus berkembang.

Arah perkembangan yang saat ini sedang dikembangkan antara lain meliputi :

Tahun	Kejadian
1940an	Komputer pertama yang membolehkan pengguna menulis kode program langsung
1950an	Generasi awal interpreter dan bahasa macro Generasi pertama compiler
1960an	Generasi kedua compiler Komputer mainframe mulai dikomersialkan Pengembangan perangkat lunak pesanan Konsep Software Engineering mulai digunakan
1970an	Perangkat pengembang perangkat lunak Perangkat minicomputer komersial
1980an	Perangkat Komputer Personal (PC) komersial Peningkatan permintaan perangkat lunak
1990an	Pemrograman berorientasi obyek (OOP) Agile Process dan Extreme Programming Peningkatan drastis kapasitas memori Peningkatan penggunaan internet
2000an	Platform interpreter modern (Java, .Net, PHP, dll) Outsourcing

VI. METODE REKAYASA PERANGKAT LUNAK

Pada rekayasa perangkat lunak, banyak model yang telah dikembangkan untuk membantu proses pengembangan perangkat lunak. Model-model ini pada umumnya mengacu pada model proses pengembangan sistem yang disebut ***System Development Life Cycle (SDLC)*** seperti terlihat pada Gambar berikut ini.



Gambar 4. System Development Life Cycle (SDLC).

- Kebutuhan terhadap definisi masalah yang jelas. Input utama dari setiap model pengembangan perangkat lunak adalah pendefinisian masalah yang jelas. Semakin jelas akan semakin baik karena akan memudahkan dalam penyelesaian masalah. Oleh karena itu pemahaman masalah seperti dijelaskan pada Bab 1, merupakan bagian penting dari model pengembangan perangkat lunak.
- Tahapan-tahapan pengembangan yang teratur. Meskipun model-model pengembangan perangkat lunak memiliki pola yang berbeda-beda, biasanya model-model tersebut mengikuti pola umum analysis – design – coding – testing - maintenance
- Stakeholder berperan sangat penting dalam keseluruhan tahapan pengembangan. Stakeholder dalam rekayasa perangkat lunak dapat berupa pengguna, pemilik, pengembang, pemrogram dan orang-orang yang terlibat dalam rekayasa perangkat lunak tersebut.

- Dokumentasi merupakan bagian penting dari pengembangan perangkat lunak. Masing-masing tahapan dalam model biasanya menghasilkan sejumlah tulisan, diagram, gambar atau bentuk-bentuk lain yang harus didokumentasi dan merupakan bagian tak terpisahkan dari perangkat lunak yang dihasilkan.
- Keluaran dari proses pengembangan perangkat lunak harus bernilai ekonomis. Nilai dari sebuah perangkat lunak sebenarnya agak susah di-rupiah-kan. Namun efek dari penggunaan perangkat lunak yang telah dikembangkan haruslah memberi nilai tambah bagi organisasi. Hal ini dapat berupa penurunan biaya operasi, efisiensi penggunaan sumberdaya, peningkatan keuntungan organisasi, peningkatan "image" organisasi dan lain-lain.

VII. TAHAPAN REKAYASA PERANGKAT LUNAK

Meskipun dalam pendekatan berbeda-beda, namun model-model pendekatan memiliki kesamaan, yaitu menggunaka pola tahapan analysis – design – coding(construction) – testing – maintenance.

1. **Analisis sistem** adalah sebuah teknik pemecahan masalah yang menguraikan sebuah sistem menjadi komponen-komponennya dengan tujuan mempelajari seberapa bagus komponen-komponen tersebut bekerja dan berinteraksi untuk meraih tujuan mereka.

   Analisis mungkin adalah bagian terpenting dari proses rekayasa perangkat lunak. Karena semua proses lanjutan akan sangat bergantung pada baik tidaknya hasil analisis. Ada satu bagian penting yang biasanya dilakukan dalam tahapan analisis yaitu pemodelan proses bisnis.

2. **Model proses** adalah model yang memfokuskan pada seluruh proses di dalam sistem yang mentransformasikan data menjadi informasi (Harris, 2003). Model proses juga menunjukkan aliran data yang masuk dan keluar pada suatu proses. Biasanya model ini digambarkan dalam bentuk Diagram Arus Data

(Data Flow Diagram / DFD). DFD meyajikan gambaran apa yang manusia, proses dan prosedur lakukan untuk mentransformasi data menjadi informasi.

3. **Disain perangkat lunak** adalah tugas, tahapan atau aktivitas yang difokuskan pada spesifikasi detil dari solusi berbasis computer (Whitten et al, 2004).

   Disain perangkat lunak sering juga disebut sebagai physical design. Jika tahapan analisis sistem menekankan pada masalah bisnis (business rule), maka sebaliknya disain perangkat lunak fokus pada sisi teknis dan implementasi sebuah perangkat lunak (Whitten et al, 2004).

   Output utama dari tahapan disain perangkat lunak adalah spesifikasi disain. Spesifikasi ini meliputi spesifikasi disain umum yang akan disampaikan kepada stakeholder sistem dan spesifikasi disain rinci yang akan digunakan pada tahap implementasi. Spesifikasi disain umum hanya berisi gambaran umum agar stakeholder sistem mengerti akan seperti apa perangkat lunak yang akan dibangun. Biasanya diagram USD tentang perangkat lunak yang baru merupakan point penting dibagian ini. Spesifikasi disain rinci atau kadang disebut disain arsitektur rinci perangkat lunak diperlukan untuk merancang sistem sehingga memiliki konstruksi yang baik, proses pengolahan data yang tepat dan akurat, bernilai, memiliki aspek user friendly dan memiliki dasar-dasar untuk pengembangan selanjutnya.

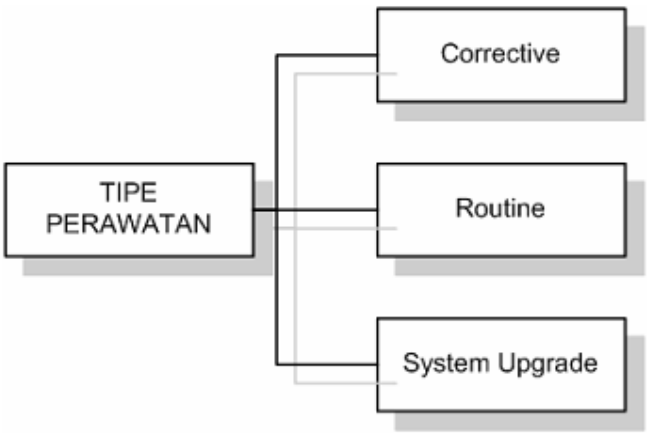
   Desain arsitektur ini terdiri dari desain database, desain proses, desain user interface yang mencakup desain input, output form dan report, desain hardware, software dan jaringan. Desain proses merupakan kelanjutan dari pemodelan proses yang dilakukan pada tahapan analisis.

4. **Konstruksi** adalah tahapan menerjemahkan hasil disain logis dan fisik ke dalam kode-kode program komputer.

5. **Pengujian** sistem melibatkan semua kelompok pengguna yang telah direncanakan pada tahap sebelumnya. Pengujian tingkat penerimaan terhadap

perangkat lunak akan berakhir ketika dirasa semua kelompok pengguna menyatakan bisa menerima perangkat lunak tersebut berdasarkan kriteria-kriteria yang telah ditetapkan.

6. **Perawatan dan Konfigurasi.** Ketika sebuah perangkat lunak telah dianggap layak untuk dijalankan, maka tahapan baru menjadi muncul yaitu perawatan perangkat lunak. Ada beberapa tipe perawatan yang biasa dikenal dalam dunia perangkat lunak seperti terlihat pada diagram di Gambar di bawah ini :



Gambar 5. Tipe-tipe perawatan.

- Tipe perawatan corrective dilakukan jika terjadi kesalahan atau biasa dikenal sebagai bugs. Perawatan bisa dilakukan dengan memperbaiki kode program, menambah bagian yang dirasa perlu atau malah menghilangkan bagian-bagian tertentu.
- Tipe perawatan routine biasa juga disebut preventive maintenance dilakukan secara rutin untuk melihat kinerja perangkat lunak ada atau tidak ada kesalahan.
- Tipe perawatan sistem upgrade dilakukan jika ada perubahan dari komponen-komponen yang terlibat dalam perangkat lunak tersebut. Sebagai contoh perubahan platform sistem operasi dari versi lama ke versi baru menyebabkan perangkat lunak harus diupgrade.

DAFTAR PUSTAKA

IEEE Xplore - *Software Engineering, IEEE Transactions on*. http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=32. Diakses pada tanggal 25 Mei 2009 jam 23.05 WIB

Mulyanto, Aunur R. 2008. *Rekayasa Perangkat Lunak Jilid 1 untuk SMK*. Direktorat Pembinaan Sekolah Menengah Kejuruan, Direktorat Jenderal Manajemen Pendidikan Dasar dan Menengah, Departemen Pendidikan Nasional : Jakarta

Pengertian Software Engineering. http://www.total.or.id/info.php?kk=Software%20Engineering. Diakses pada tanggal 25 Mei 2009 jam 22.50 WIB

Rekayasa Perangkat Lunak, http://www.irfante06.blog.unsoed.ac.id/files/2009/

Wikipedia, the free encyclopedia - *Software engineering* . http://en.wikipedia.org/wiki/Software_engineering. Diakses pada tanggal 25 Mei 2009 jam 23.00 WIB