Perangkat lunak dan Pengembangannya
(Software and Software Engineering)

Apa yang dipelajari ???

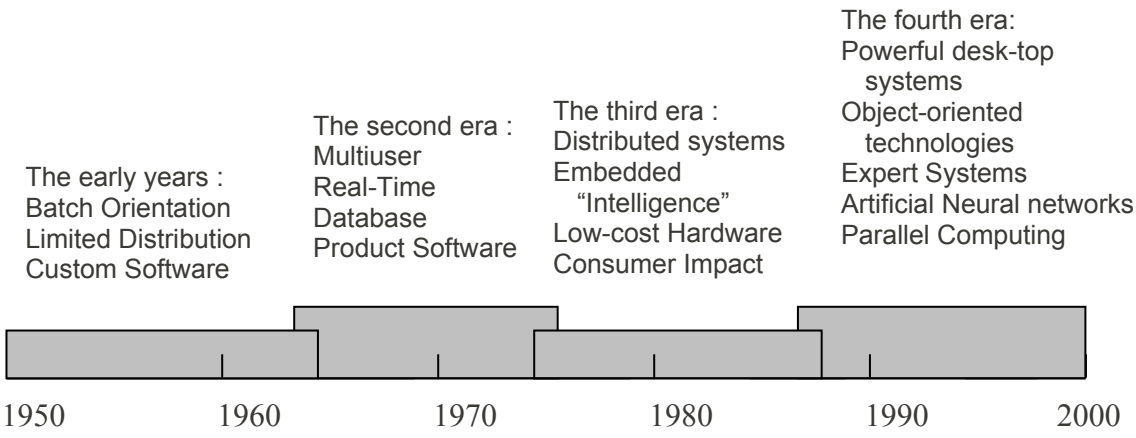
- Mata kuliah ini memperkenalkan prinsip-prinsip ujicoba dan implementasi perangkat lunak komputer.
- Topik implementasi meliputi : bentuk-bentuk peng-kodean, prinsip-prinsip *packaging*, penggunaan ulang *source code* (*reuse*), membuat sourecode yang dapat diujikan dan dipelihara (*testable and maintainable code*).
- Topik-topik dalam ujicoba meliputi : peninjauan ulang kode (*code reviews*), Ujicoba berbasis spesifikasi (*black box testing*), Ujicoba struktural (clear box testing), tes ketercukupan (*adequacy test*), dan tes administrasi.

Software dan Evolusi Software

Software adalah program komputer dan dokumentasi yang terkait dengannya. Produk software yang dikembangkan untuk *customer* tertentu atau untuk kebutuhan general market. Produk software terdiri dari:

1. Generic – dikembangkan untuk dijual ke berbagai *customer*
2. Bespoke (custom) – dikembangkan untuk memenuhi kebutuhan *customer* yang tertentu

Software mengalami evolusi dalam beberapa dekade, terlihat dalam gambar berikut :



Software Evolution

Selama awal periode, hardware dengan fungsi umum memegang peranan penting, sedangkan software merupakan tambahan untuk setiap aplikasi dan mempunyai distribusi yang terbatas. Program dikembangkan dan digunakan oleh beberapa orang atau organisasi, mobilitas masih sangat rendah. Perancangan program/sistem merupakan proses yang terpisah dan dokumentasi tidak tersedia. Selama awal periode lebih ditekankan pada implementasi sistem berbasis komputer dari pada pengembangan sistem.

Era kedua mulai pertengahan 1960 sampai 1970-an. Sistem multiprogramming dan multi user mulai diperkenalkan sebagai konsep baru dalam interaksi manusia-komputer. Sistem Real-time mampu mengumpulkan, menganalisa dan memindahkan data dari berbagai sumber dengan proses pengawasan dan menghasilkan output dalam hitungan milisecond. Karakteristik lainnya dari era kedua adalah penggunaan produk-produk software yang disediakan oleh "software house", dimana program dikembangkan untuk distribusi yang lebih luas dan pangsa pasar yang lebih beragam. Program digunakan oleh ratusan orang bahkan ribuan, sehingga banyak muncul permintaan baru terhadap program. Ketika banyak permintaan dan kesalahan yang ditemukan dalam sebuah program, maka dilakukan modifikasi sesuai keinginan user, aktifitas ini secara kolektif disebut sebagai *software maintenance.*

Era ketiga mulai pertengahan 1970 hingga akhir 1980-an. Sistem terdistribusi dengan multiple komputer, dan masing-masing menjalankan fungsi secara serentak, serta berkomunikasi antar komputer, menambah rumit sistem berbasis komputer. Era ketiga juga ditandai dengan penggunaan mikroprosesor, komputer personal dan stasiun kerja yang handal. Mikroprosesor menghasilkan produk dengan kemampuan

tinggi seperti oven microwave, peralatan diagnostik darah. Dalam banyak kasus, teknologi software diintegrasikan di produk-produk oleh para staf teknikal.

Era keempat baru saja dimulai. Teknologi berorientasi objek diterapkan dalam berbagai area aplikasi. *Expert system* dan *artificial intelligence* mulai berubah dari laboratorium menjadi aplikasi praktikal untuk area permasalahan yang lebih luas.

Karakteristik Software

Penting untuk memeriksa karakteristik software, yang membedakannya dari hal lainnya yang dapat dibangun/dibuat oleh manusia. Software merupakan elemen sistem yang bersifat logikal, karakteristik software diantaranya :

1. *Software dibangun/direncanakan, bukan barang pabrikan dalam artian umum*
   Walaupun sepertinya ada kesamaan antara Pengembangan software (*software development*) dan pembuatan hardware (*hardware manufacture*), tetapi kedua aktivitas ini secara mendasar sangat berbeda. Dalam kedua aktifitas, kualitas yang baik didapat melalui perancangan yang baik, dan melakukan konstruksi untuk produk tetapi dengan pendekatan yang berbeda.
2. *Software tidak lekang oleh waktu*
   Hardware dapat habis dimakan waktu (*wear out*), dan ketika terjadi kerusakan atau kesalahan pada hardware, yang dapat dilakukan adalah dengan mengganti bagian yang rusak dengan yang baru, tetapi hal tersebut tidak berlaku dalam software. Jika terjadi kerusakan atau kesalahan pada software maka kesalahan tersebut dapat mengindikasikan kesalahan pada saat perancangan
3. *Kebanyakan software merupakan `custom-builts` dari pada dibuat untuk suatu kepentingan tertentu*
   Jika dalam perancangan suatu hardware seorang perancang cukup menggambarkan rancangannya, kemudian membangun/membuat hardware yang dimaksud, tetapi tidak demikian dengan software, tidak ada gambar yang dapat mendeskripsikan komponen software.

Aplikasi-aplikasi software

1. **System Software**, merupakan kumpulan program-program yang dibuat untuk menjalankan program lainnya. Beberapa contoh *system software* diantaranya : *compiler, editor,* dan *file management utilities*, jenis lainnya adalah *OS, component, drivers* dan *telecommunications processors*. Dari kesemuanya, area sistem software dapat ditandai dengan interaksi yang kuat dengan hardware komputer, penggunaan yang banyak oleh banyak user, operasi konkuren yang memerlukan penjadwalan, penggunaan bersama sumberdaya yang ada, manajemen proses yang baik, struktur data yang rumit, dan interface eksternal yang banyak
2. **Real time software**, adalah software yang mengawasi/menganalisa/ mengatur kejadian nyata. Elemen dari real-time software termasuk komponen pengumpulan data yang mengumpulkan dan memformat informasi dari lingkungan luar, komponen analisis merubah informasi yang dibutuhkan oleh aplikasi, komponen input/output yang memberikan respon pada lingkungan luar dan komponen pengawasan yang mengkoordinasikan seluruh komponen sehingga respon real-time (dengan kisaran waktu 1 milidetik s/d 1 menit) dapat dilaksanakan.
3. **Business software,** pemrosesan informasi bisnis merupakan area aplikasi software terbesar. Sistem terbatas (seperti program penggajian, program inventory dsb) mengalami perubahan menjadi sistem informasi manajemen (MIS) yang mengakses satu atau lebih database yang berisikan informasi bisnis. Aplikasi dalam area ini merestrukturisasi data yang telah ada dengan suatu cara yang bertujuan untuk menfasilitasi operasi bisnis dan pembuat keputusan manajemen.
4. **Engineering and scientific software**, karakteristik *engineering* dan *scientific software* adalah dengan digunakannya sejumlah algoritma yang rumit. Lingkup aplikasi mulai dari astronomi hingga vulkanologi, dari biologi molekuler hingga pabrikan otomatis. Aplikasi barunya berupa sistem simulasi dan sistem interaksi lainnya yang bersifat real-time.
5. **Embedded software**, produk-produk pintar mulai menguasai konsumen, *embedded software* digunakan untuk mengatur produk dan sistem untuk konsumen dan pasar industri, misalnya fungsi-fungsi digital pada kendaraan seperti alat ukur tangki bensin, tampilan dashboard, sistem rem dan lain-lain.
6. **Personal Computer software**, meliputi *word processing, spreadsheet, computer graphic, entertainment, database management,* aplikasi bisnis keuangan dsb. PC software merepresentasikan rancangan interface manusia-komputer yang paling inovatif.
7. **Artificial intelligence software**, menggunakan algoritma nonnumerik untuk mengatasi masalah yang rumit, contohnya adalah *expert system*. Juga dikenal dengan istilah *knowledge based system.* Area aplikasi lainnya adalah *pattern recognitions* (gambar dan suara), pembuktian teorema dan permainan.

Software Engineering

Software engineering adalah bidang disiplin ilmu rekayasa yang terkait dengan aktifitas produksi software. Perekayasa software semestinya mengadopsi pendekatan yang sistematik dan terorganisir dalam menyelesaikan pekerjaannya dengan memanfaatkan tools dan teknik yang tepat bergantung pada permasalahan yang akan dipecahkan, batasan pengembangan dan sumberdaya yang tersedia.

Software process adalah sekumpulan aktifitas yang terstruktur yang dibutuhkan untuk pengembangan atau evolusi software. Aktifitas umum dalam *software process* diantaranya :

1. *Specification* – Apa yang harus dikerjakan sistem dan batasan pengembangannya
2. *Development* – Produksi dari sistem software
3. *Validation* – Pemeriksaan apakah software memenuhi kebutuhan customer
4. *Evolution* – Perubahan software terhadap perubahan kebutuhan

Software process model adalah representasi sederhana dari *software process*, yang dipresentasikan dari sudut pandang tertentu. Contoh dari sudut pandang proses diantaranya :

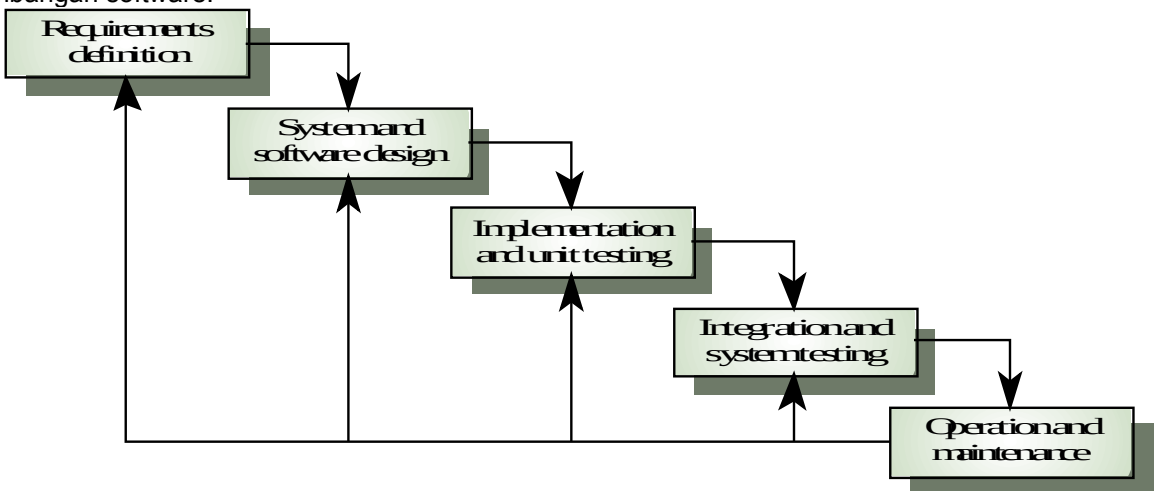
1. *Workflow perspective - sequence of activities*
2. *Data-flow perspective - information flow*
3. *Role/action perspective - who does what*

Model proses secara umum :

1. *The waterfall model*, memisahkan dan membedakan tahapan-tahapan spesifikasi dan pengembangan.
2. *Evolutionary development*, specifikasi dan pengembangan saling berselakan
3. *Formal systems development*, model sistem matematika yang secara formal diubah untuk di implementasikan
4. *Reuse-based* development, sistem dibangun dengan menggunakan komponen-komponen yang sudah ada

The Waterfall Model

Dalam *software lifecycle* (waterfall model) terdapat beberapa tahapan utama yang menggambarkan aktivitas pengembangan software.



Software lifecycle (*Sommerville, 2001*)

- *Requirements analysis and definition*, merupakan layanan, batasan dan tujuan dari sistem yang dibuat dengan mengkonsultasikannya bersama para pengguna sistem. Hal tersebut didefinisikan secara detail dan ditampilkan sebagai spesifikasi dari sistem.
- *System and software design*, proses desain sistem membagi kebutuhan sistem akan software dan hardware. Hal tersebut membangun arsitektur sistem keseluruhan. Desain software meliputi identifikasi dan penjabaran abstrasi sistem software dasar dan keterhubungannya.

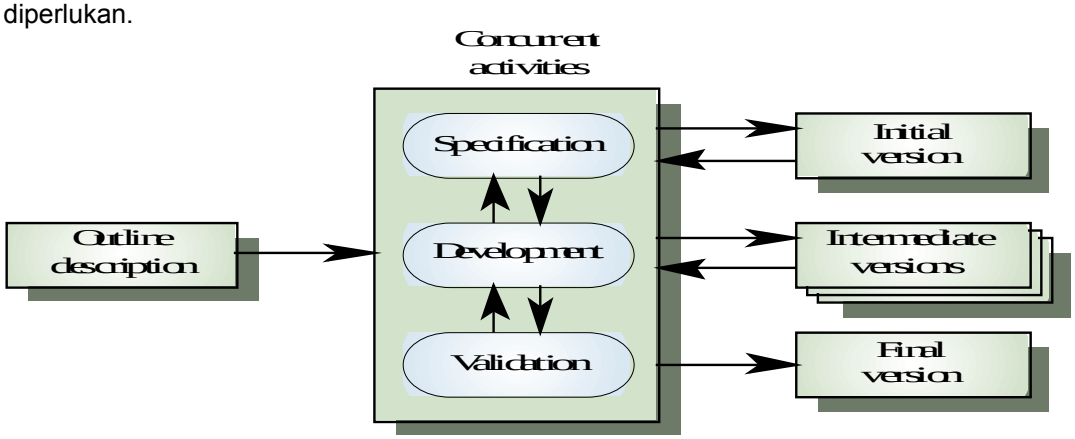
- *Implementation and unit testing,* selama tahapan ini, desain software direalisasikan sebagai sekumpulan program atau unit program. Unit testing meliputi verifikasi bahwa setiap unit telah memenuhi spesifikasinya.
- *Integration and system testing,* unit-unit program individual digabungkan (*integrated*) dan diujicoba (*tested*) sebagai sebuah sistem lengkap untuk memastikan bahwa kebutuhan-kebutuhan software telah terpenuh. Setelah pengujian, sistem software desampaikan pada pelanggan.
- *Operation and maintenance,* biasanya tahapan ini merupakan tahapan terpanjang dalam *lifecycle.* Sistem di-install dan digunakan secara praktikal. Pemeliharaan meliputi perbaikan kesalahan yang tidak diketahui pada tahapan sebelumnya, memperbaiki implementasi unit sistem dan meningkatkan layanan sistem ketika terdapat kebutuhan baru.

Masalah-masalah yang muncul dalam model *waterfall* ini diantaranya :
1. Pembagian proyek yang tidak flexibel dalam bentuk tahapan yang berbeda.
2. Hal ini mengakibatkan kesulitan saat merespon perubahan kebutuhan klien.
3. Dengan demikian, model ini hanya akan sesuai apabila kebutuhan telah disepakati dan dipahami dengan baik antara klien dan pengembang

Evolutionary development

Terdapat 2 macam pendekatan :

1. *Exploratory development* (Pengembangan dengan penyelidikan)
   Bertujuan untuk bekerja sama dengan klien untuk membangun sebuah sistem dari spesifikasi awal. Harus dimulai dengan pemahaman kebutuhan yang memadai
2. *Throw-away prototyping*
   Bertujuan untuk mengerti akan kebutuhan sistem. Dimulai dengan pemahanan kebutuhan yang sangat minim, karena pada umumnya konsumen mendefinisikan sekumpulan tujuan secara umum untuk software, tetapi tidak mengidentifikasikan secara detail mengenai input, proses dan output yang diperlukan.



Evolutionary development Model

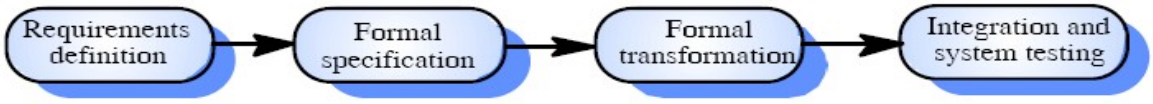
Masalah yang mungkin muncul diantaranya :
1. Ketidakjelasan dari alur proses
2. Struktur sistem yang amat buruk
3. Dibutuhkan kemampuan khusus (misalnya : penguasaan pemrograman untuk *rapid prototyping*)

Model ini dapat diterapkan pada :
1. Sistem interaktif berukuran kecil atau sedang
2. Untuk bagian/subsistem dari sebuah sistem yang besar (misalnya User interface)
3. Untuk sistem dengan siklus hidup/penggunaan yang pendek

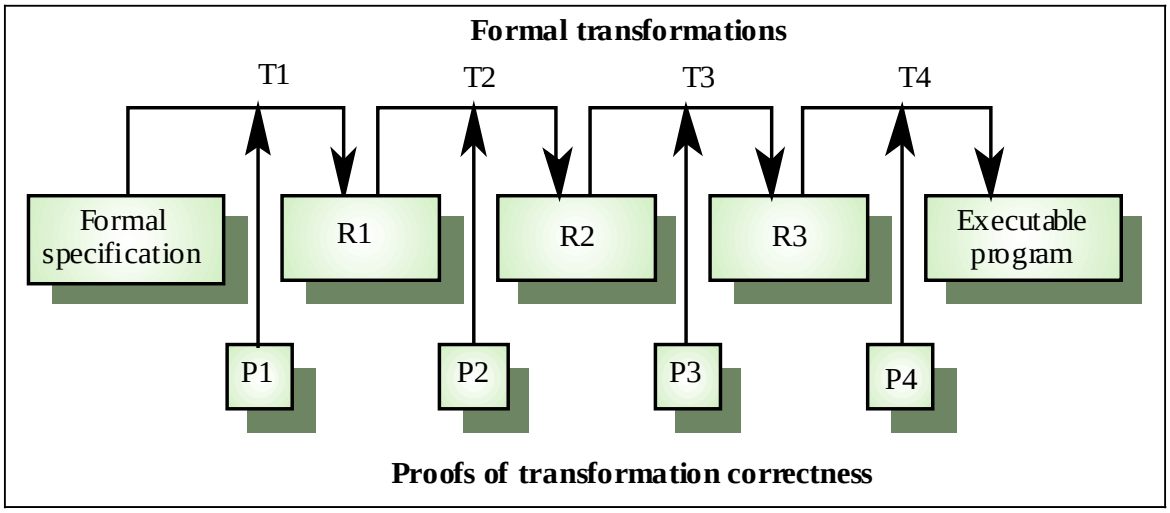
Formal systems development

1. Didasarkan pada perubahan bentuk spesifikasi matematika melalui representasi yang berbeda menjadi program *executable*.

2. Perubahan bentuk bersifat '*correctness-preserving*', sehingga secara lasung dapat menunjukkan bahwa program telah sesuai dengan spesifikasinya.



Formal systems development

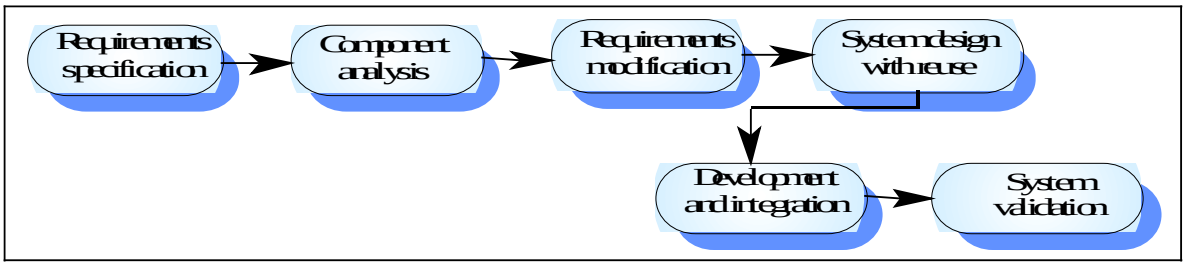


Masalah yang mungkin muncul diantaranya :
1. Dibutuhkan ketrampilan dan pelatihan khusus untuk mengaplikasikan teknik ini
2. Kesulitan dalam menspesifikasikan beberapa aspek ke dalam sistem misalnya dalam penentuan user interface

Model ini dapat diterapkan pada sistem-sistem kritis khususnya sistem yang mengutamakan faktor keselamatan dan keamanan sebelum sistem utamanya dioperasikan

***Reuse-based* development**

1. Berdasarkan pada pendekatan pakai-ulang yang sistematik dimana sistem dintegrasikan dari komponen-komponen yang telah tersedia atau disebut COTS *(Commercial-off-the-shelf) systems.*
2. Tahapan proses :
   - *Component analysis*
   - *Requirements modification*
   - *System design with reuse*
   - *Development and integration*
3. Pendekatan ini menjadi sangat penting namun kajian dan pengalaman masih sangat terbatas.



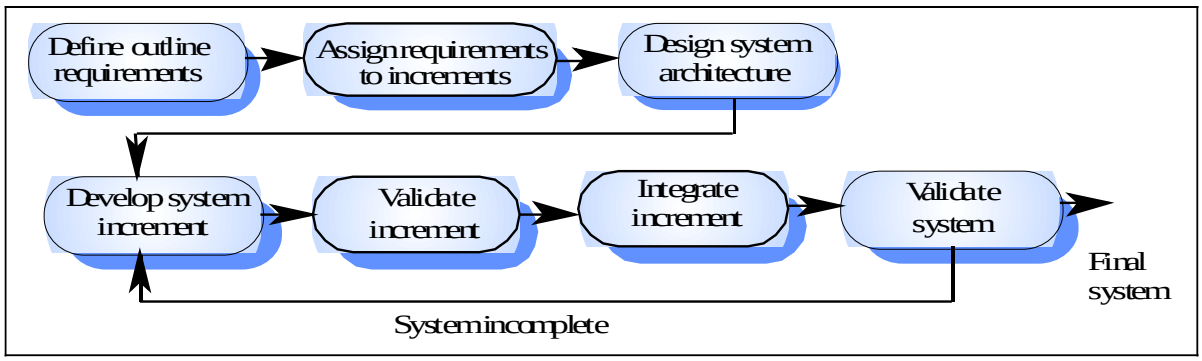
Reuse-based development

Iterasi proses :

1. Dalam suatu *project system requirements* SELALU mengalami perubahan, itulah sebabnya iterasi proses pada tahapan awal yang selalu dikerjakan berulang-ulang menjadi bagian dari proses pada sistem yang lebih besar
2. Iterasi dapat di aplikasikan pada model proses generik manapun
3. Terdapat 2 pendekatan :
    a. Incremental development
    b. Spiral development

A. Incremental development

1. Bukan sekedar menghantarkan sistem sebagai "single delivery", *development* dan *delivery* dipecah menjadi beberapa tahapan dimana tiap tahap akan men-*deliver* bagian dari kebutuhan fungsionalitas sistem
2. Kenutuhan User diberi urutan prioritas dan kebutuhan dengan prioritas tertinggi akan disertakan pada pengembangan awal.
3. Ketika pengembangan pada suatu tahapan dimulai, maka kebutuhan akan di bekukan, walaupun kebutuhan untuk tahapan berikutnya dapat melanjutkan untuk berubah.



Incremental development process

Keuntungan menggunakan pendekatan *Incremental development :*

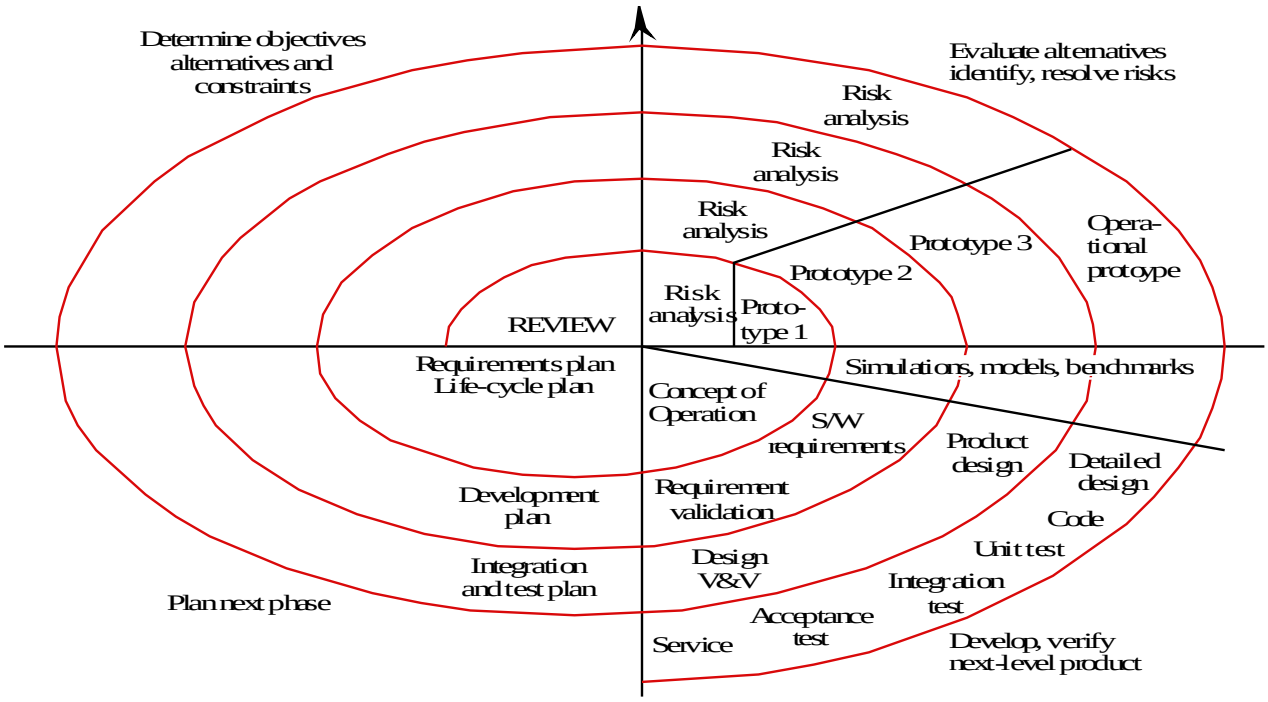
1. Kebutuhan klien dapat dikirimkan di tiap-tiap *increment* sehingga sistem akan tersedia lebih awal
2. *increment* awal bertindak sebagai prototype untuk membantu menentukan kebutuhan untuk *increment* selanjutnya
3. Resiko yang lebih rendah terjadinya kesalahan projek secara keseluruhan
4. Layanan sistem dengan prioritas tertinggi yang akan paling di uji

B. Spiral development

1. Proses direpresentasikan sebagai spiral bukan aktifitas berurutan dengan *backtracking*
2. setiap *loop*/perulangan dalam spiral merepresentasikan sebuah tahapan dalam suatu proses.
3. Tidak terdapat tahapan yang tetap seperti spesifikasi atau desain, perulangan-perulangan (*loops*) dalam spiral dipilih bergantung pada apa yang dibutuhkan
4. Secara eksplisit resiko dikenali dan diselesaikan selama proses berlangsung

Spiral development terbagi menjadi 4 sektor :

1. *Objective setting*, menentukan tujuan untuk phase yang diidentifikasi
2. *Risk assessment and reduction,* resiko dikenali dan aktifitas dilakukan untuk mengurangi sejumlah resiko
3. *Development and validation,* model pengembangan sistem ditentukan yang memungkinkan penggunaan berbagai model secara umum
4. *Planning,* projek ditinjau ulang dan tahapan berikutnya dari spiral direncanakan

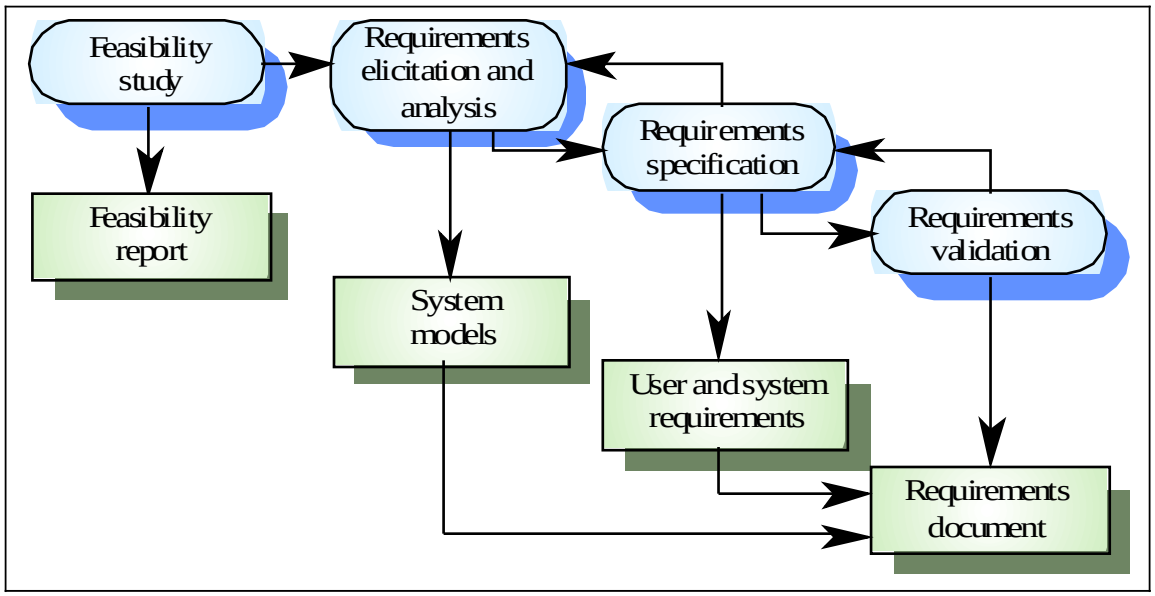


Spiral development process

Software Specification

Spesifikasi software adalah suatu proses yang bertujuan untuk menentukan layanan-layanan apa yang dibutuhkan dari sebuah sistem dan batasan pada tahapan pengoperasian dan pengembangan sistem Aktivitas ini biasa disebut perencanaan kebutuhan-kebutuhan (*requirements engineering*), yang merupakan tahapan kritis dari proses pembuatan software, karena kesalahan pada tahap ini akan berpengaruh/ menjadi masalah dalam desain dan implementasi sistem. Proses perencanaan kebutuhan-kebutuhan (*requirements engineering*) dibagi menjadi empat tahap utama.

1. *Feasibility study* (Studi kelayakan), sebuah perkiraan/tafsiran ditetapkan untuk menetapkan apakah kebutuhan user yang diidentifikasikan telah terpenuhi oleh teknologi software dan hardware saat ini. Studi ini akan menetapkan apakah sistem yang akan dibuat akan efektif secara biaya dan dapat dibangun dengan batasan biaya yang ada. Studi kemungkinan ini harus relatif murah dan cepat. Hasilnya harus melaporkan keputusan apakah akan dilanjutkan dengan analisis yang lebih detail.
2. *Requirements elicitation and analysis* (Pembentukan kebutuhan dan analisis), merupakan proses untuk mendapatkan kebutuhan-kebutuhan sistem melalui observasi dari sistem yang ada, diskusi bersama pengguna potensial dan analisis tugas. Dapat meliputi pembentukan satu atau lebih model sistem dan prototype yang berbeda. Hal ini membantu analis untuk mengerti sistem yang dispesifikasikan.
3. *Requirements specification* (Spesifikasi kebutuhan), merupakan aktifitas penerjemahan informasi yang didapat selama aktifitas analisis kedalam dokumen yang mendefinisikan kumpulan kebutuhan. Terdapat 2 tipe kebutuhan, yaitu : (1) Kebutuhan pengguna, yang merupakan pernyataan abstrak dari kebutuhan sistem untuk *costumer* dan *end user,* (2) Kebutuhan sistem, merupakan deskripsi yang lebih detail dari fungsi-fungsi yang harus disediakan.
4. *Requirements validation* (Validasi kebutuhan), merupakan aktifitas yang memeriksa kebutuhan untuk direalisasikan, konsisten dan lengkap. Selama proses ini kesalahan dalam dokumen kebutuhan dapat terjadi, dan selanjutnya diubah untuk perbaikan.

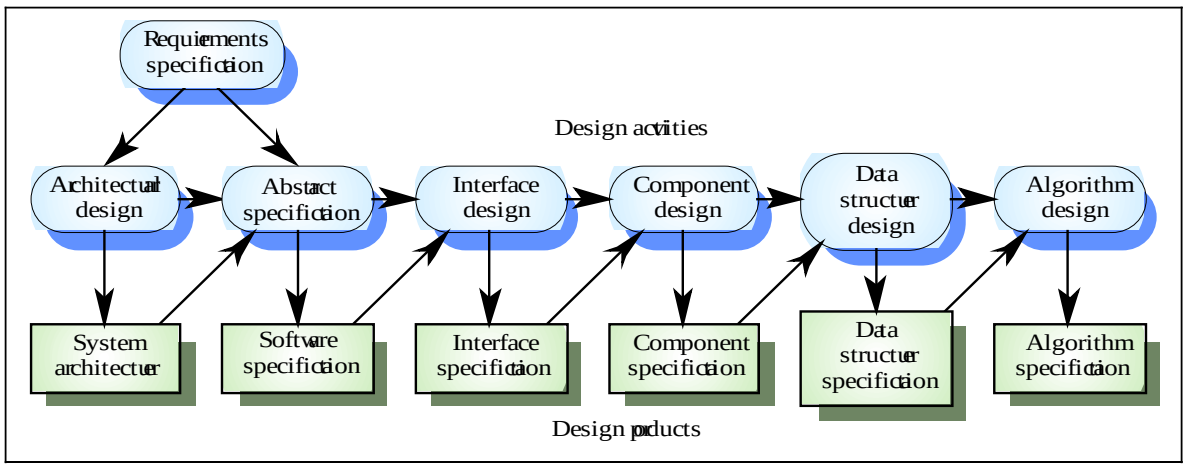


The Requirements engineering process(*Sommerville, 2001*)

Software Implementation and Design

Tahapan implementasi dari pembuatan software merupakan proses mengkonversi spesifikasi sistem kedalam *executable* sistem. Meliputi proses-proses dari desain dan pemrograman software, tetapi jika digunakan pendekatan *evolutionary,* menyertakan perbaikan dari spesifikasi software.

Desain Software, yaitu proses mendesain struktur software yang mengacu pada dokumen spesifikasi. Implementasi adalah menterjemahkan struktur yang telah didapat kedalam *executable program.* Aktifitas desain dan implementasi sangatlah berdekatan dan kadang saling tumpang tindih.



Software Design Process/Activity

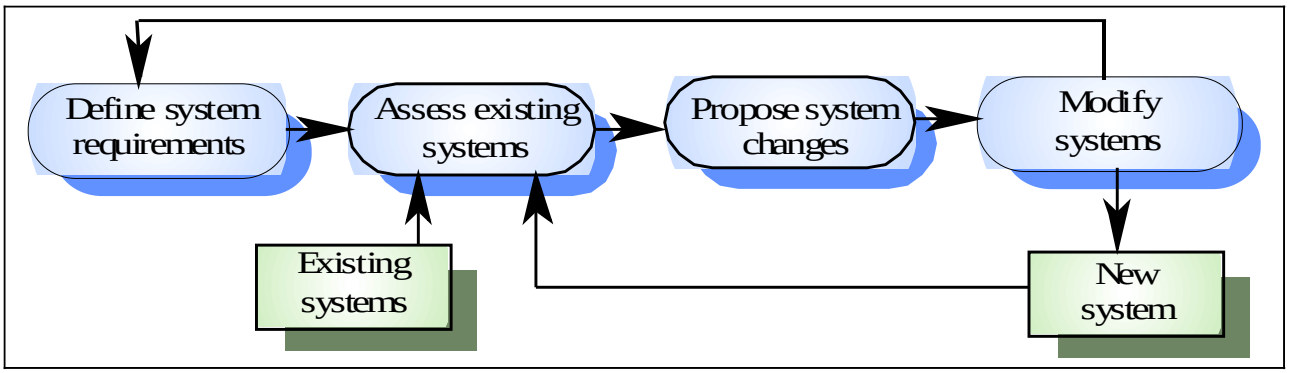
Metode-metode desain :

Metode desain yaitu pendekatan sistematik dalam pengembangan desain software. Hasil desain biasanya didokumentasikan dalam bentuk model grafikal. Model-model yang dapat digunakan diantaranya :

1. Data-flow model
2. Entity-relation-attribute model
3. Structural model
4. Object models

Software Evolution

Pada dasarnya software bersifat flexible dan dapat dirubah, seiring dengan perubahan kebutuhan dan perubahan kondisi bisnis. Dengan pengembangan dan pemeliharaan yang terus meningkat, maka hanya akan terdapat sedikit sistem yang benar-benar baru.



Software Evolution