

# Rekayasa Perangkat Lunak

# Atribut Produk

- Kinerja
- Reliability
- Pelayanan
- Maintanability
- Garansi
- Mudah digunakan
- Penampilan
- Merek
- Kemasan
- Model terakhir

# Definisi

- Rekayasa perangkat lunak adalah penetapan dan penggunaan prinsip-prinsip rekayasa yang tangguh/teruji dalam upaya memperoleh perangkat lunak secara ekonomis, handal dan bekerja efisien di mesin nyata, dan berkaitan dengan metode dan kaidah yang diperlukan dalam mengembangkan perangkat lunak untuk computer.
- Sedangkan pengertian rekayasa perangkat lunak menurut IEEE :*Rekayasa perangkat lunak adalah pendekatan sistematis untuk pengembangan, operasi, pemeliharaan dan pemberhentian pemakaian perangkat lunak.*

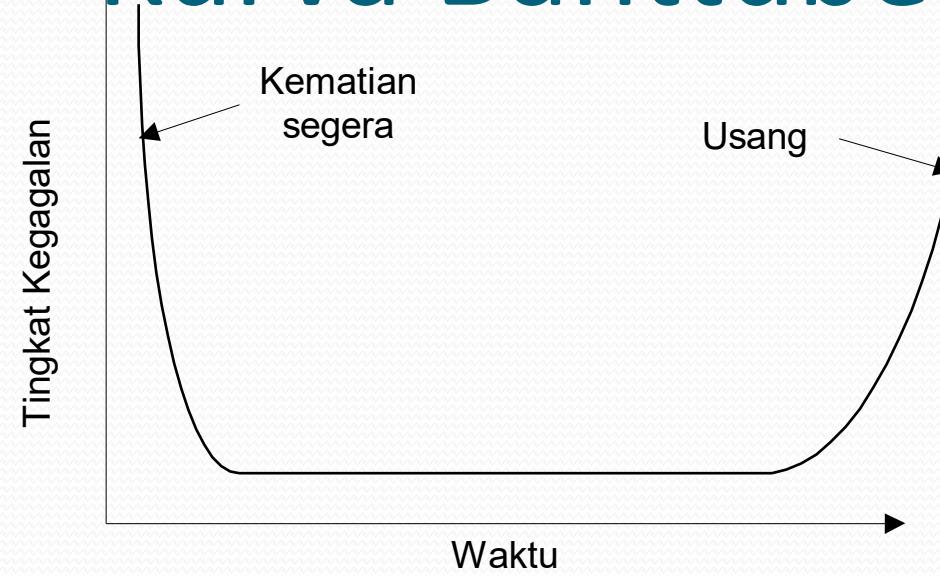
# Prinsip Perangkat Lunak

- Kekakuan (*Rigor*),  
Rekayasa yang dilakukan harus sesuai dengan keinginan user, walupun terkadang diperlukan kreativitas perekayasa untuk membuat perangkat lunak.
- Resmi (formal)  
Pemilihan salah satu metodologi/pendekatan perangkat lunak, berdampak pada harus dilaksanakannya aktivitas rekayasa sesuai dengan metodologi yang dipilih, serta notasi yang dipilih harus selalu konsisten digunakan
- Pemisahan kepentingan  
Berkaitan dengan apek-aspek persoalan : melebarnya focus kerja, kompleksitas sistem.
- Abstraksi  
Menggambarkan keseluruhan sistem dalam bentuk yang sederhana
- Modularitas  
Mendekomposisikan persoalan menjadi modul-modul independent sehingga memisahkan perhatian mengenai persoalan internal modul dan interaksi modul-modul dengan lingkungan luarnya.

# Karakteristik Perangkat Lunak

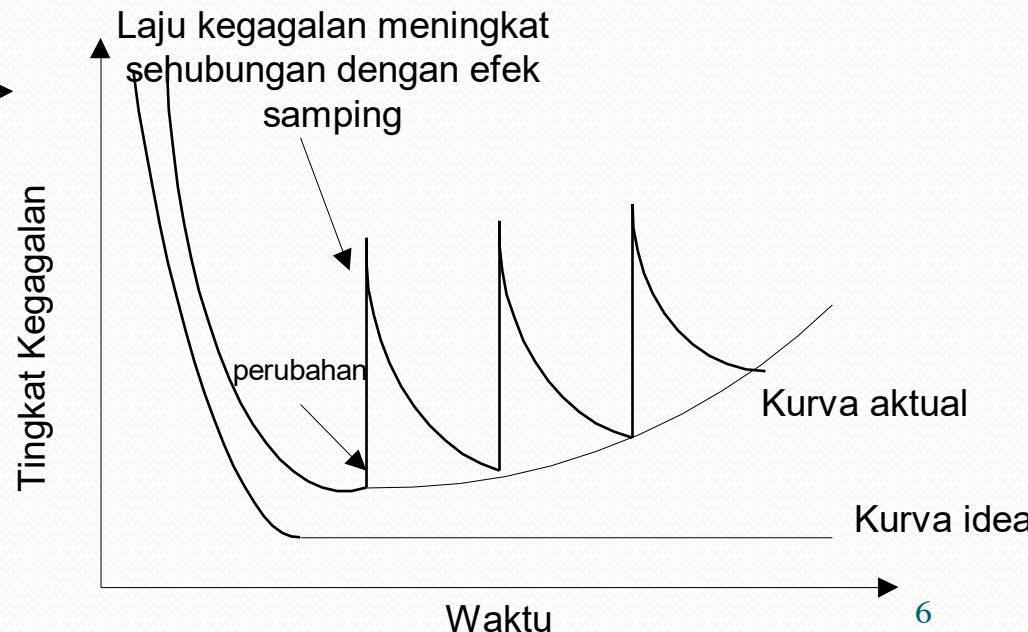
- Perangkat lunak dibangun dan dikembangkan, tidak dibuat dalam bentuk yang klasik. Walaupun perkembangan antara perangkat keras dan perangkat lunak sangat ekuivalen, namun aktivitas diantara keduanya sangat berbeda.
- Perangkat lunak tidak pernah usang,
- Sebagian besar perangkat lunak dibuat secara custom built, serta tidak dapat dirakit dari komponen yang sudah ada

# Kurva Bahttube



- Proses Umur Perangkat Keras

- Proses Umur Perangkat Lunak



# Tahapan Umur Perangkat Lunak

Periode	Simbolisasi	Penyebab	Solusi
Pembuatan	DFR (Decreasing Failure Rate)	Defect, rendahnya control kualitas,	Quality control, Pengujian penerimaan,
Pemakaian	CFR (Constant Failure Rate)	Human error	Redudancy, User friendly,
Kadaluarsa	IFR (Increasing Failure Rate)	Peningkatan kebutuhan, prosedur kerja baru,	Teknologi, Modifikasi

# Pemodelan Sistem

- Asumsi, digunakan untuk mengurangi jumlah kemungkinan (permutasi) dan variasi yang mungkin.
- Penyederhanaan, digunakan untuk menciptakan model dengan waktu yang tepat.
- Pembatasan (*Boundaries*), digunakan untuk membatasi lingkup sistem.
- Batasan (*Constraint*), digunakan untuk menunjukkan cara dimana model tersebut diciptakan dan pendekatan yang dilakukan pada saat model diimplementasikan.
- Preferensi, digunakan untuk menunjukkan arsitektur yang dipilih untuk semua data, fungsi dan teknologi.

# Ciri-ciri software yang baik

- **Maintainability** (dapat dipelihara)
  - Software bisa menangani perubahan spek kebutuhan
- **Dependability** (dapat diandalkan)
  - Aman, selamat, tidak menyebabkan keruksakan fisik
- **Efficiency** (Efisien)
  - Software mampu mengoptimalkan resource
- **Acceptability** (Kemampupakaian)
  - Software bisa diterima user sebagaimana rancangan. Mudah dimengerti, digunakan and compatible dengan sistem yang lain

# Rekayasa Sistem

## Rekayasa Perangkat Lunak

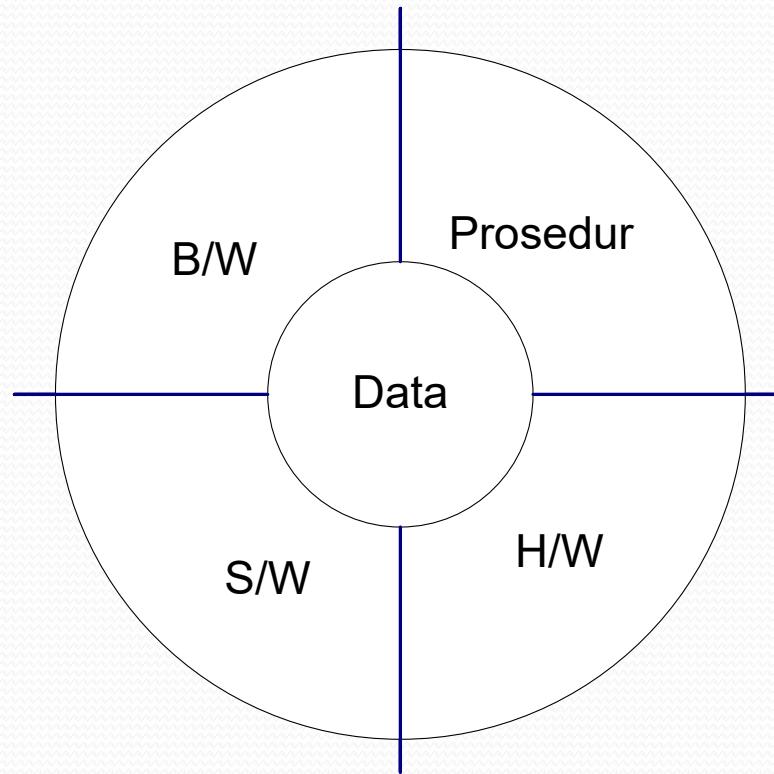
# Definisi

- Rekayasa sistem adalah kegiatan untuk melakukan perekayasaan (manipulasi /membuat/mengembangkan /modifikasi) sistem, dengan berfokus pada berbagai elemen analisis, perancangan dan pengorganisasian elemen tersebut menjadi produk, jasa atau teknologi untuk mentransformasi informasi atau control.
- Proses rekayasa sistem disebut *rekayasa informasi* bila konteks kerja rekayasa berfokus pada perusahaan bisnis. Pada tahapan pembuatan produk maka proses pembuatannya disebut *rekayasa produk*.

# (CBIS – Computer Based on Information System)

- Sistem berbasis computer adalah serangkaian atau tatanan elemen-elemen yang diatur untuk mencapai tujuan yang telah ditentukan melalui pemrosesan informasi.
- Tujuan tersebut dapat berupa upaya mendukung fungsi bisnis atau untuk mengembangkan suatu produk yang dapat dijual untuk menghasilkan keuntungan bisnis.

# Elemen CBIS



# Rekayasa Informasi

- Tujuan global rekayasa informasi adalah untuk mengaplikasikan teknologi informasi dengan cara tertentu yang melayani dengan baik kebutuhan bisnis secara keseluruhan.
- Tujuan khusus dari rekayasa informasi (information engineering – IE) adalah untuk menentukan arsitektur yang memungkinkan suatu bisnis menggunakan informasi secara efektif, dan mengimplementasikan semua arsitektur tersebut.
- IE membuat transisi ke dalam domain RPL yang teknis – proses dimana sistem informasi, aplikasi dan program dianalisis, didesain dan dibangun.

# Arsitektur Sistem Informasi

## 1. Arsitektur data (database)

Arsitektur data memberikan kerangka kerja untuk kebutuhan informasi dari bisnis atau fungsi bisnis

## 2. Arsitektur aplikasi

Arsitektur aplikasi melingkupi elemen-elemen dari suatu sistem yang mentrasformasikan objek ke dalam arsitektur data untuk keperluan bisnis. (perangkat lunak atau penggabungan peran manusia dengan prosedur bisnis yang belum diotomatisasi)

## 3. Infrastruktur teknologi

Infrastruktur teknologi menyangkut penggunaan perangkat keras dan perangkat lunak untuk mendukung aplikasi dan data. (Berupa computer, jaringan, telekomunikasi, storage).

# Lapisan Perangkat Lunak

- Lapisan fondasi berupa Proses.

Lapisan fondasi adalah lapisan proses pengembangan perangkat lunak yang merupakan perekat bagi lapisan-lapisan teknologi yang lain, serta yang memungkinkan pengembangan perangkat lunak yang rasional dan tepat.

Lapisan proses mendefinisikan kerangka kerja untuk sekumpulan proses pokok (KPA –Key Process Area)

- Lapisan Metode

Lapisan Metode memberikan cara teknis dalam membangun perangkat lunak pada kegiatan penetapan kebutuhan, analisis, perancangan, pembangunan program, pengujian.

- Lapisan Alat Bantu

Lapisan yang mendefinisikan alat Bantu yang digunakan untuk mendukung pelaksanaan proses dan metode.

# Perencanaan Strategi Informasi

- Langkah pertama rekayasa Informasi adalah perencanaan strategi informasi (*Information Strategic Planning - ISP*).
- Sasaran ISP :
  1. Menentukan sasaran dan tujuan bisnis strategis
  2. Mengisolasi faktor sukses kritis (Critical Success Factor – CSF) yang memungkiskan bisnis mencapai tujuan dan sasaran tersebut
  3. Menganalisis pengaruh teknologi dan otomasi terhadap tujuan dan sasaran
  4. Menganalisis informasi yang ada untuk menentukan peranan data dalam mencapai sasaran dan tujuan

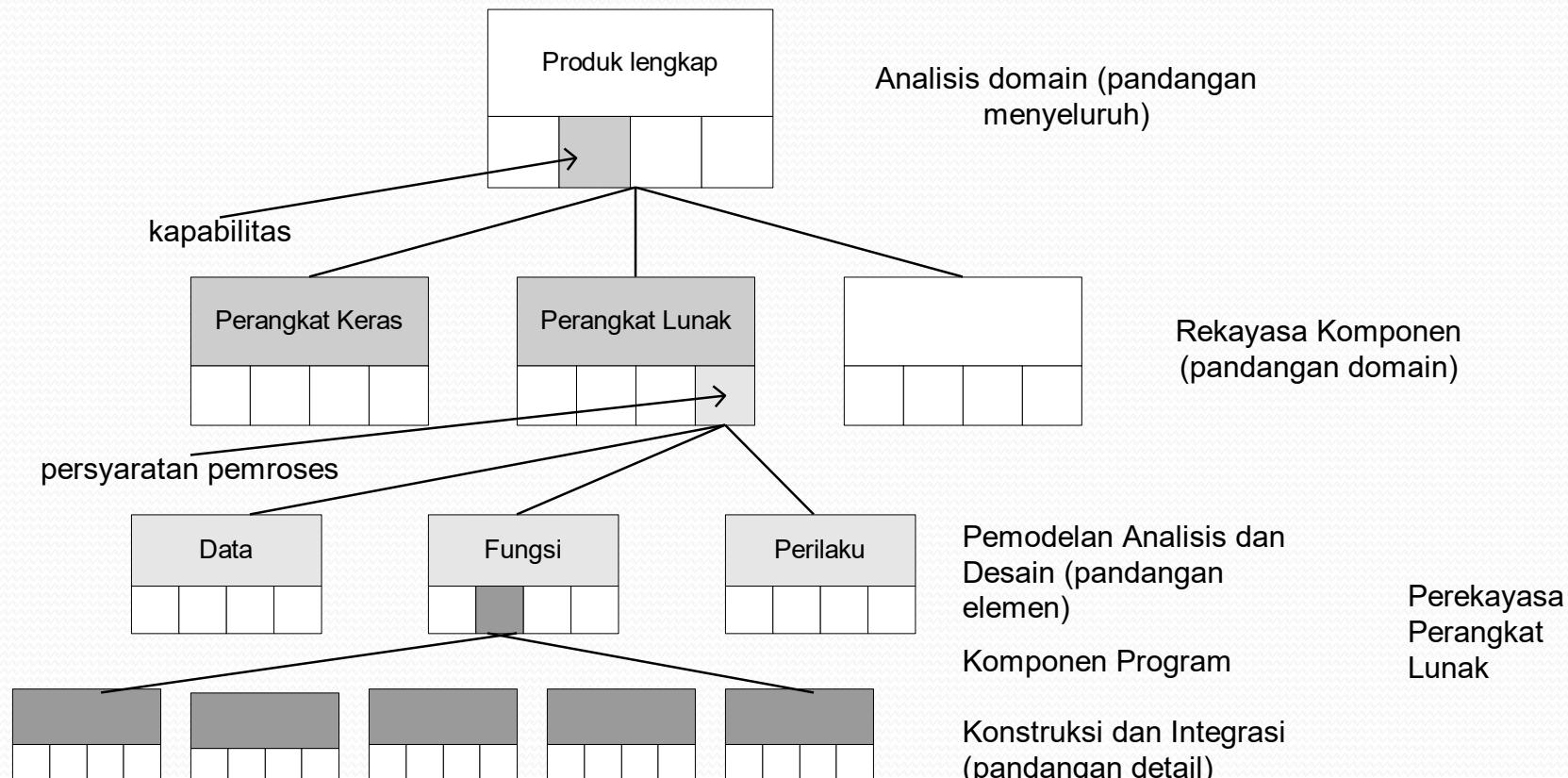
# ISP

- Setiap area bisnis banyak menggunakan teknologi informasi, maka ISP harus mengidentifikasi teknologi apa yang sekarang ada dan bagaimana teknologi itu digunakan untuk mencapai sasaran dan tujuan
- Analisis pengaruh teknologi akan menguji sasaran dan tujuan, dan memberikan indikasi mengenai teknologi-teknologi yang akan berpengaruh langsung dan tidak langsung terhadap upaya mencapai sasaran dan tujuan dengan sukses.

# Rekayasa Produk

- Rekayasa Produk merupakan suatu aktivitas pemecahan masalah. Data produk, fungsi dan tingkah laku yang dinginkan, ditemukan, dianalisis dan dialokasikan ke dalam komponen rekayasa individual
- Produk yang dihasilkan adalah perangkat lunak. Rekayasa Perangkat lunak merupakan suatu aktivitas pemecahan masalah. Data produk, fungsi dan tingkah laku yang dinginkan, ditemukan, dianalisis dan dialokasikan ke dalam komponen rekayasa individual

# Hirakhi rekayasa



# Lapisan Perangkat Lunak

- Lapisan fondasi berupa Proses.

Lapisan fondasi adalah lapisan proses pengembangan perangkat lunak yang merupakan perekat bagi lapisan-lapisan teknologi yang lain, serta yang memungkinkan pengembangan perangkat lunak yang rasional dan tepat.

Lapisan proses mendefinisikan kerangka kerja untuk sekumpulan proses pokok (KPA –Key Process Area)

- Lapisan Metode

Lapisan Metode memberikan cara teknis dalam membangun perangkat lunak pada kegiatan penetapan kebutuhan, analisis, perancangan, pembangunan program, pengujian.

- Lapisan Alat Bantu

Lapisan yang mendefinisikan alat Bantu yang digunakan untuk mendukung pelaksanaan proses dan metode.

# Milestone Activity

## Kerangka Kerja Proses Umum

Aktivitas Kerangka Kerja

Rangkaian Tugas

Tugas

Kejadian Penting

SQA

Aktivitas Pendukung

# Alat Bantu Perangkat Lunak

## 1. Berorientasi aliran data (proses)

Alat bantu yang digunakan :

- a) Konteks Diagram – Data Flow Diagram – Kamus Data
- b) Control Flow Diagram – State Transition Diagram

## 2. Berorientasi Struktur Data

Alat Bantu yang digunakan :

- a) Diagram Warnier Orr (DSSD - Data Structured System Development)
- b) JSD (Jackson System Development)
- c) Diagram ER – Normalisasi – Relasi Tabel

## 3. Berorientasi Objek

Alat Bantu yang digunakan :

- a) OOAD – Object Oriented Analysis and Design (Coad dan Yourdon)
- b) OMT – Object Modelling Technique (Rumbaugh)
- c) OOSE – Object Oriented Software Engineering (Ivar Jacobson)
- d) UML – Unified Modeling Language

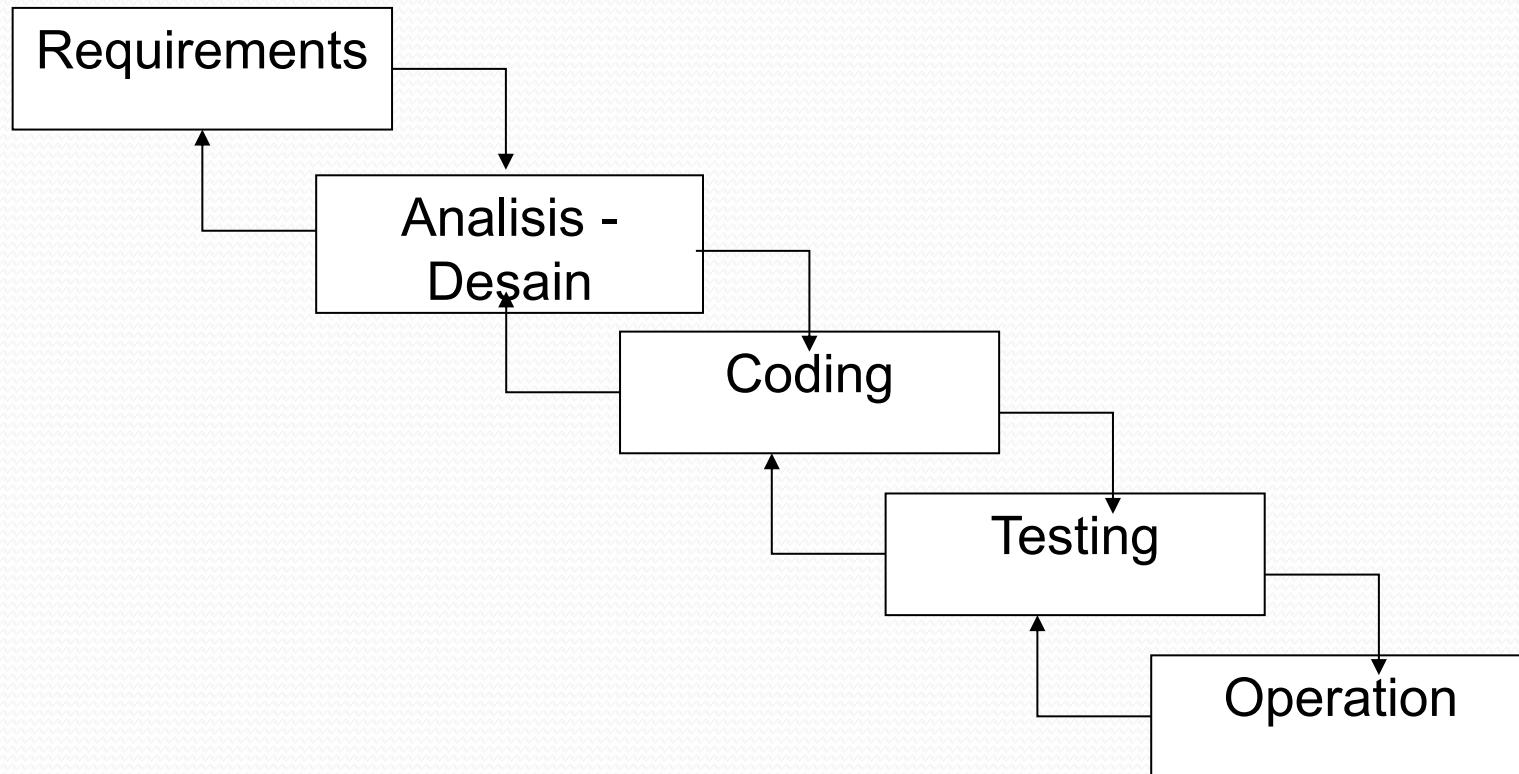
# Siklus Hidup Perangkat Lunak

Rekayasa Perangkat Lunak

# Definisi Perangkat Lunak

- Rekayasa perangkat lunak adalah penetapan dan penggunaan prinsip-prinsip rekayasa yang tangguh/teruji dalam upaya memperoleh perangkat lunak secara ekonomis, handal dan bekerja efisien di mesin nyata, dan berkaitan dengan metode dan kaidah yang diperlukan dalam mengembangkan perangkat lunak untuk computer. [Roger S. Pressman, “S/W Engineering”]
- Rekayasa Perangkat Lunak merupakan suatu aplikasi yang menerapkan prinsip-prinsip keilmuan untuk (1) Mengubah suatu permasalahan ke dalam solusi permasalahan kerja perangkat lunak, (2) Keberlangsungan perawatan perangkat lunak hingga akhir hidup perangkat lunak [Alan M. Davis, “S/W Requirement”]

# Model Umum Perangkat Lunak



# Requirement

- Definisi Requirement (Permintaan)
  - Rumusan bahasa : Sesuatu yang diinginkan atau diperlukan
  - IEEE : [1] Kondisi atau kemampuan yang diperlukan oleh seorang user untuk memecahkan suatu permasalahan atau mencapai suatu sasaran, [2] Suatu kondisi atau kemampuan yang harus dicapai atau dikerjakan oleh sistem, untuk memenuhi suatu kontrak, standar, spesifikasi atau dokumen lain secara formal.

# Kegiatan dalam tahapan requirement

- Menganalisis Masalah

Bertukar pikiran tentang permasalahan tersebut, mengidentifikasi semua kendala yang mungkin ada dalam pemecahan masalah, mencari informasi dan pengetahuan tentang masalah tersebut.

- Mendeskripsikan perangkat lunak

Menyiapkan dokumen yang menjelaskan perilaku dari perangkat lunak yang akan dibuat, mengelola ide, menyelesaikan konflik pendapat, dan mengurangi ketidakkonsistenan dan ambiguitas.

# Analisis dan Design

- Pada tahap awal desain, yang dilakukan adalah membagi sistem perangkat lunak ke dalam komponen tertentu, secara berulang hingga subkomponen tersebut tidak dapat dipecahkan lagi.
- Tahapan desain lebih lanjut adalah membuat dokumentasi dari algoritma untuk setiap modul yang akan diperlukan.

# Coding

- Tahapan Coding merupakan tahapan implementasi untuk membuat/membangun aplikasi dengan menggunakan suatu bahasa pemrograman tertentu.
- Bahasa pemrograman yang digunakan, baik untuk DDL (*Data Definition Language*) maupun DML (*Data Manipulation Language*)

# Testing

- Tahapan ini dilakukan setelah selesainya fase coding. Pengujian yang dilakukan dapat menggunakan metode white box dan metode blackbox.
- Pengujian dapat dilakukan terhadap data maupun kapasitas perangkat.

# Maintenance

- Fase maintenance merupakan fase untuk merawat sistem, dengan cara melakukan modifikasi/revisi terhadap perangkat lunak.
- Perubahan perangkat lunak dapat terjadi karena perubahan kebijaksanaan perusahaan, penggantian perangkat keras, kebutuhan perangkat lunak yang berubah.

# Software Process

Serangkaian kegiatan dan hasil-hasilnya yang diperlukan untuk menghasilkan aplikasi tertentu.

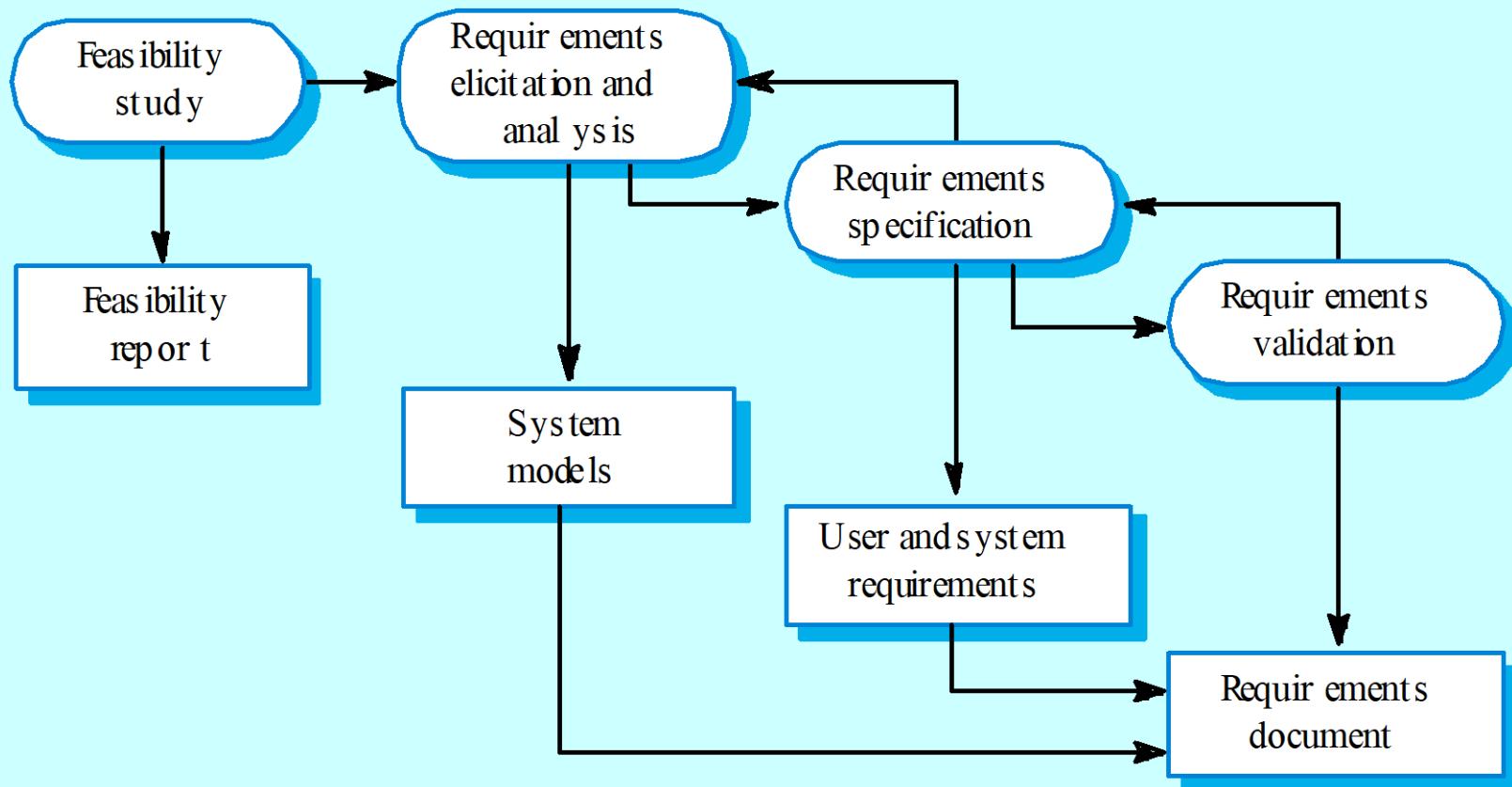
**Spesifikasi**

**Pengembangan**

**Validasi**

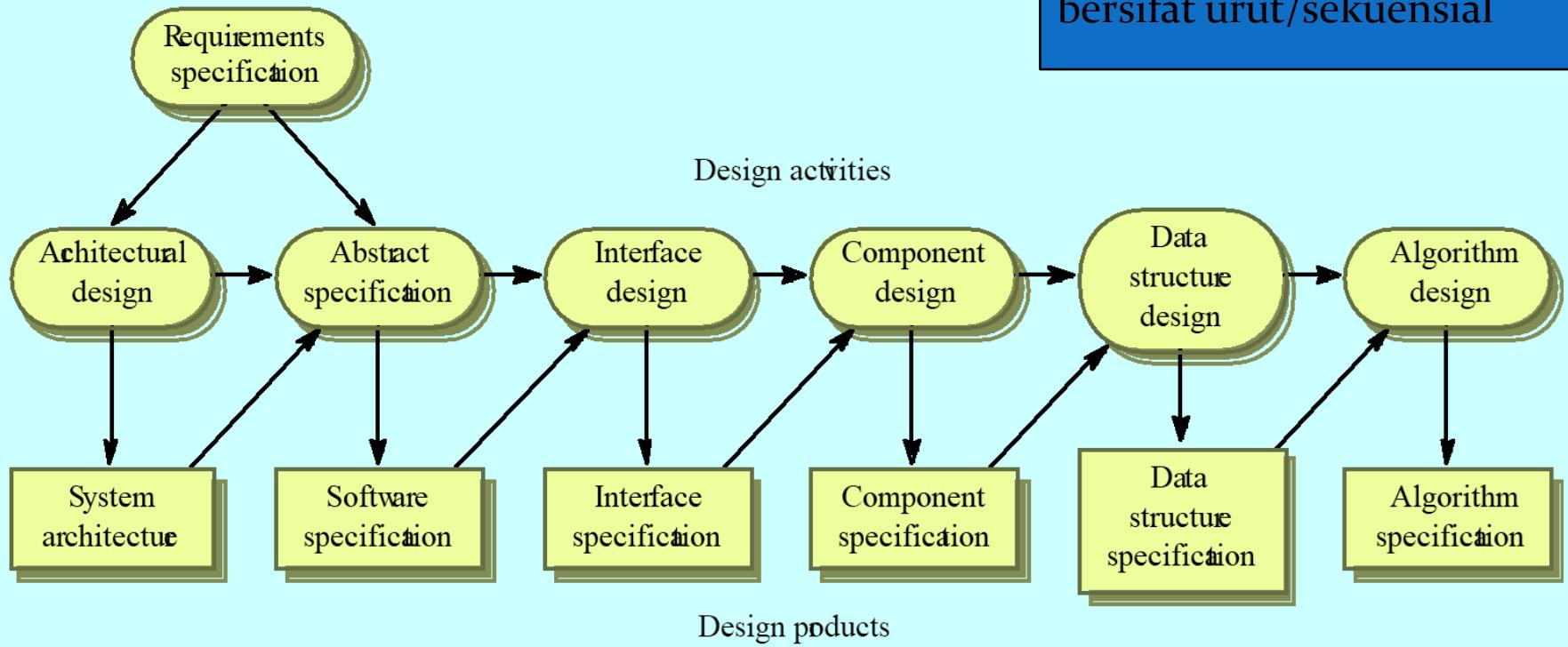
**Evolusi**

# Proses Rekayasa Persyaratan

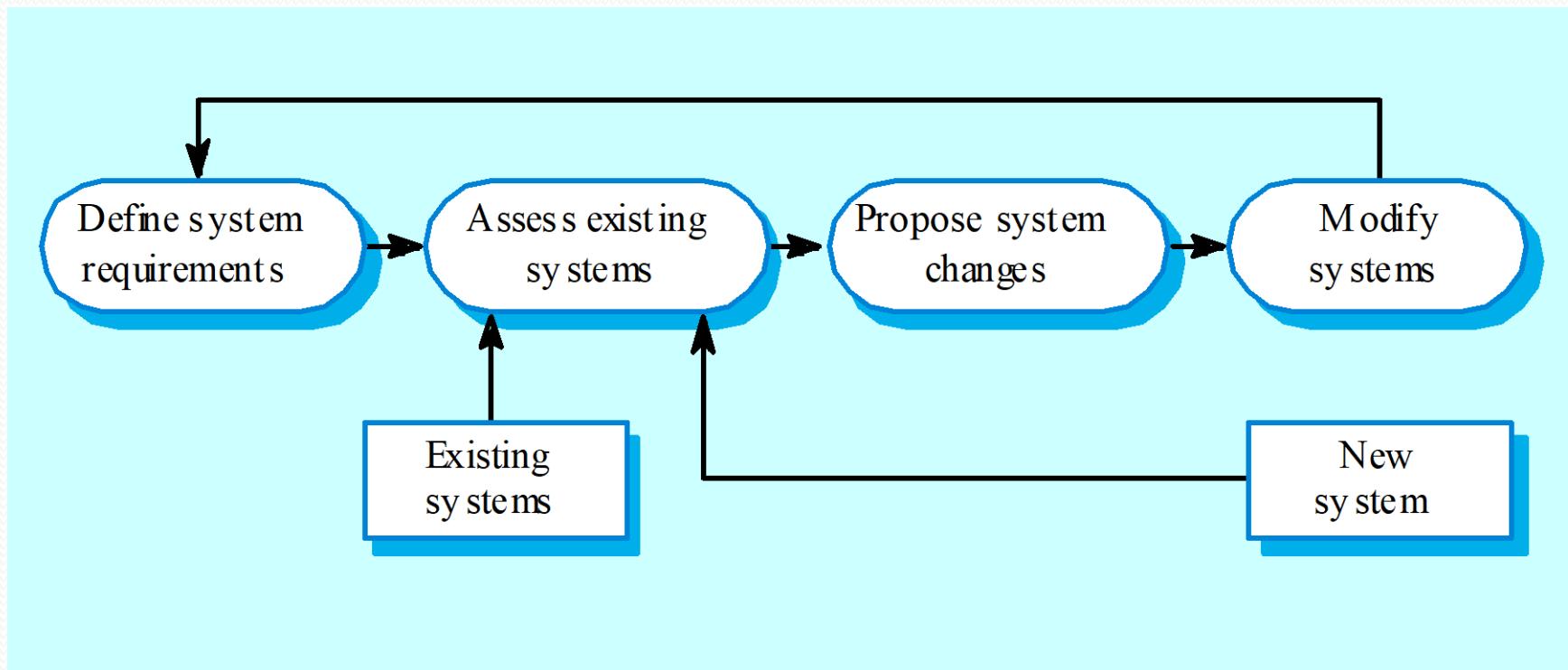


# Proses Perancangan Software

Tahapan perancangan bersifat urut/sekuensial



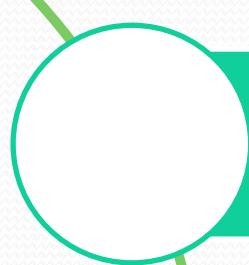
# System evolution



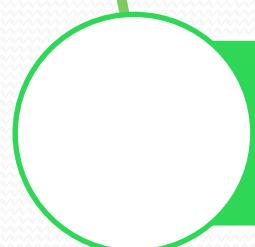
# Produk Software

- ▶ **Generik** (terbuka utk siapapun) ≈ DBMS, Word Processor, Sistem Operasi, paket untuk menggambar, alat bantu manajemen proyek
  - Spek hanya dikontrol oleh sendiri oleh Vendor Software
- ▶ **Pesanan** (disesuaikan dgn kebutuhan pelanggan tertentu saja)
  - Berdasarkan kontrak kerja
  - Spek dikontrol oleh pelanggan tertentu

# Model Proses Software



Waterfall (linear)



Evolutionere (iterasi)



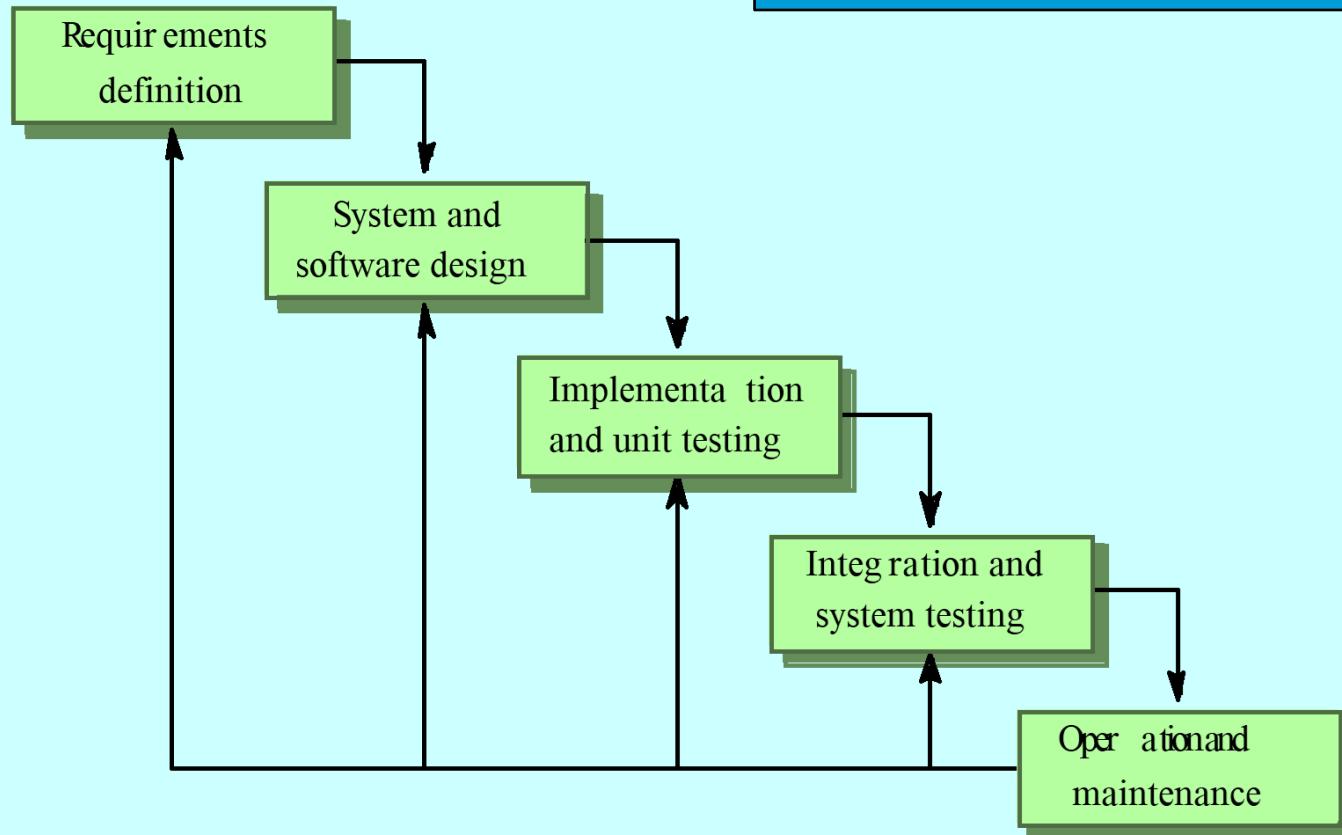
Component-based

# Model Proses Software

- ▶ **Waterfall** – pengembangan yang bersifat linear dari mulai spesifikasi s/d pemeliharaan.
- ▶ **Evolutionere** – pendekatan tumpang tindih kegiatan spesifikasi, pengembangan, dan validasi. Sistem sejak awal dikembangkan dgn cepat berdasarkan spesifikasi abstrak, lalu disempurnakan berdasarkan masukan dari pelanggan sampai sistem dapat memenuhi kebutuhan pelanggan tersebut.
- ▶ **Component-based** – pengembangan dengan cara menggunakan komponen yang dapat dipakai ulang.

# Model Waterfall

#Bukan model linear murni

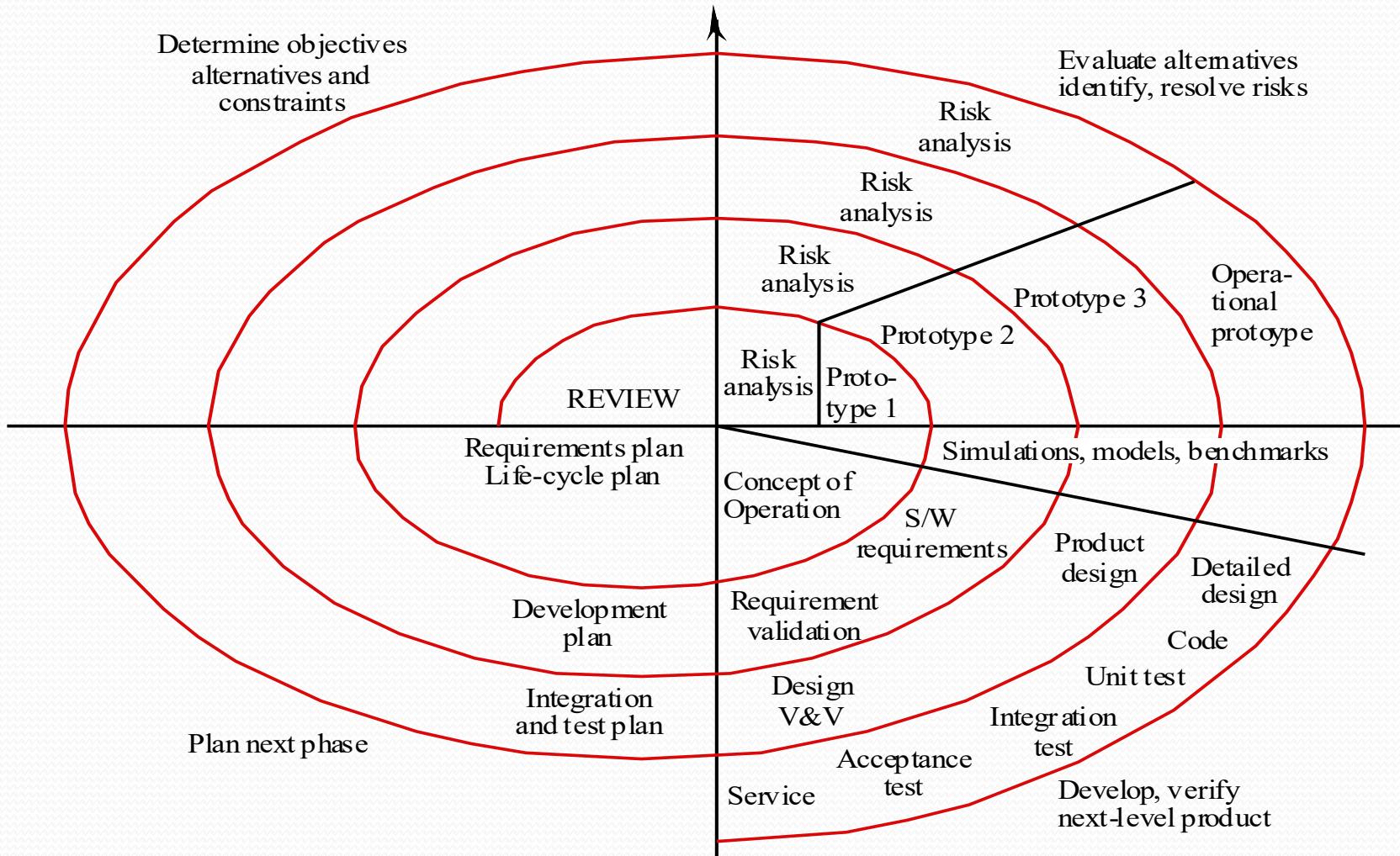


# Analisis Waterfall

## Features

- ▶ Sistematis, setiap tahapan prosesnya jelas
- ▶ Sudah terbukti handal
- ▶ Cocok untuk pengembangan software yang bersifat generik
  - Prosesnya sudah benar-benar jelas dan tidak berubah-ubah

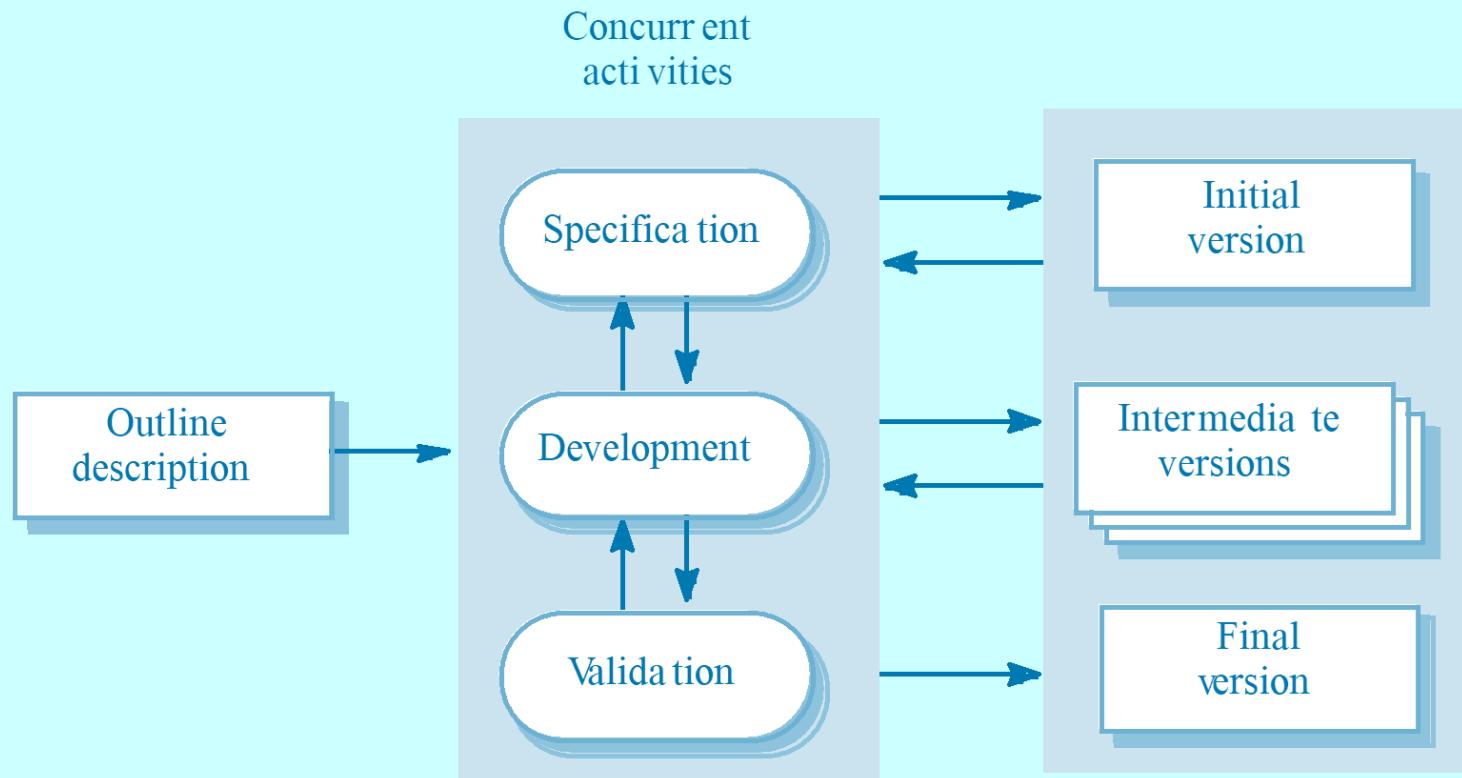
# Spiral model of the software process



# Evolutionary development

- **Exploratory development**
  - Objective is to work with customers and to evolve a final system from an initial outline specification.  
Should start with well-understood requirements and add new features as proposed by the customer.
- **Throw-away prototyping**
  - Objective is to understand the system requirements.  
Should start with poorly understood requirements to clarify what is really needed.

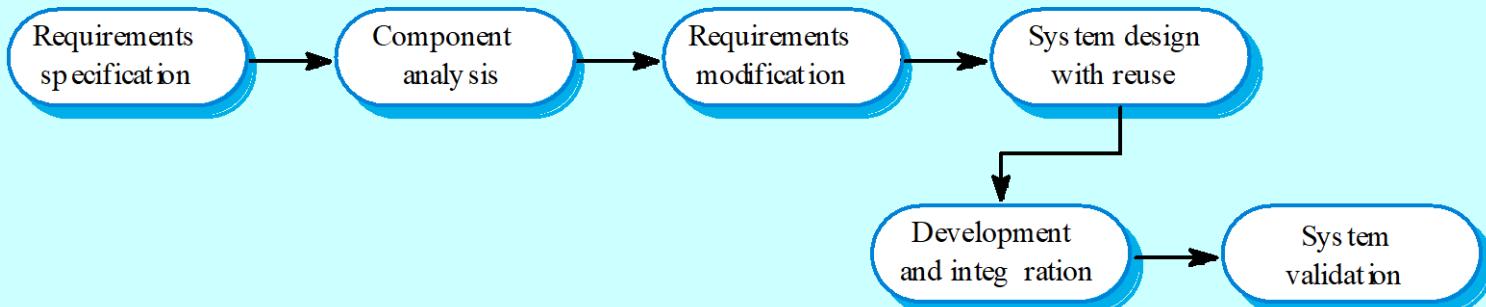
# Iterative



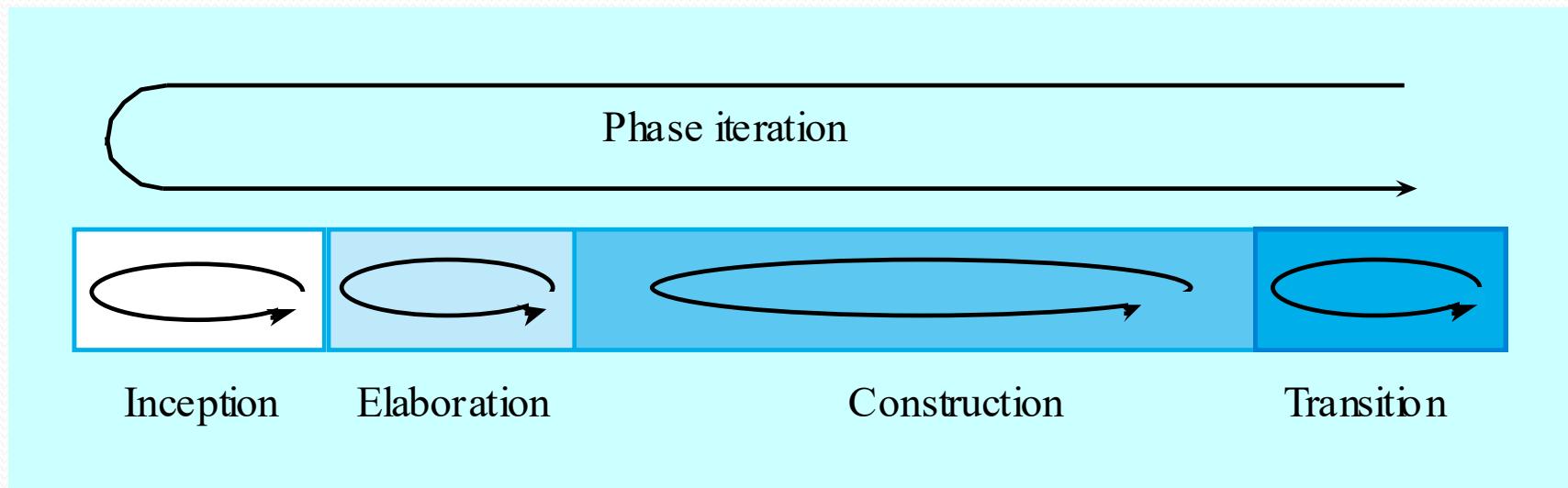
# Component-based software engineering

- Based on systematic reuse where systems are integrated from existing components or COTS (Commercial-off-the-shelf) systems.
- Process stages
  - Component analysis;
  - Requirements modification;
  - System design with reuse;
  - Development and integration.
- This approach is becoming increasingly used as component standards have emerged.

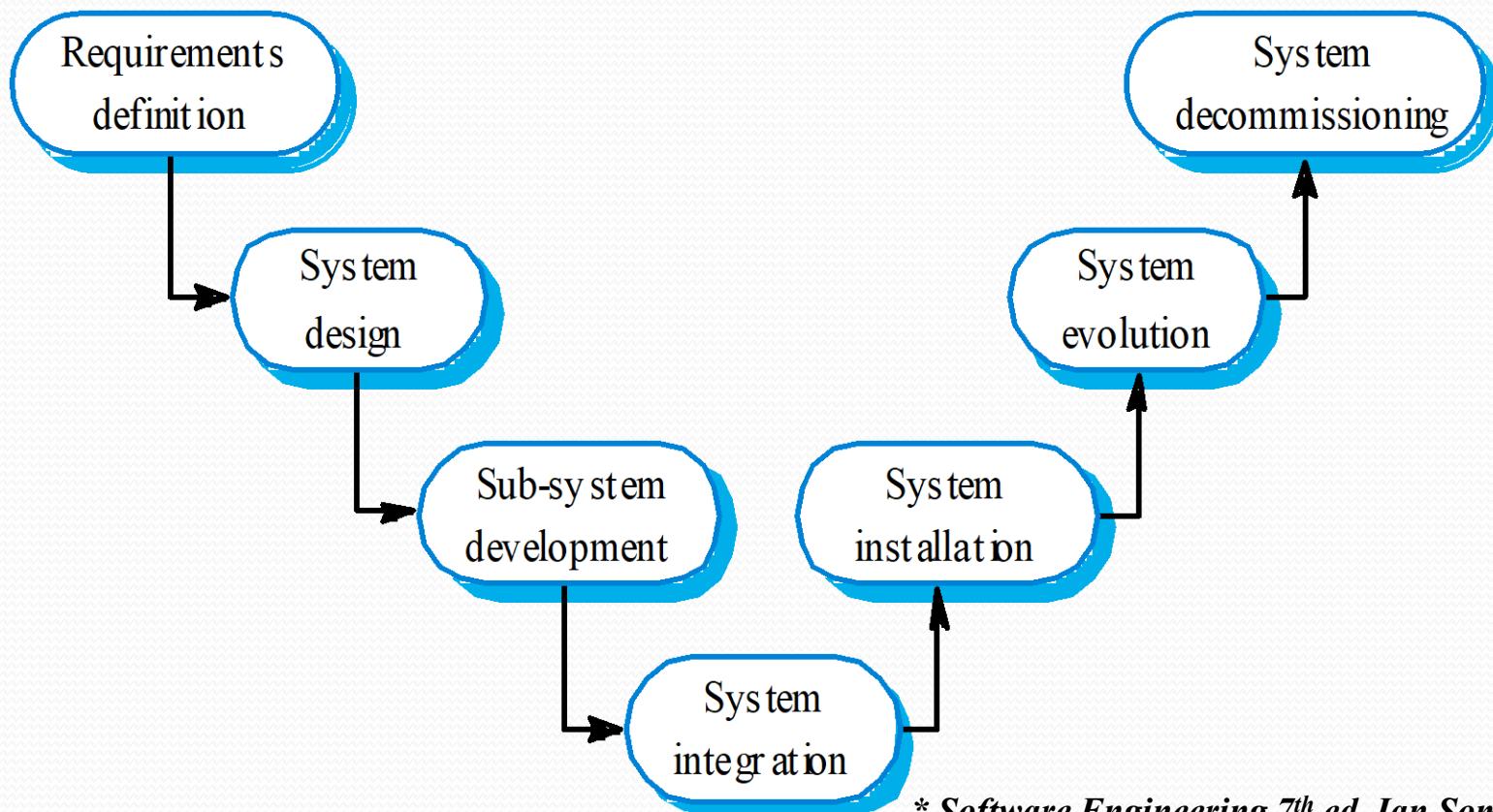
# Component Based



# RUP phase model



# The Systems Engineering Process



\* *Software Engineering 7<sup>th</sup> ed, Ian Sommerville*