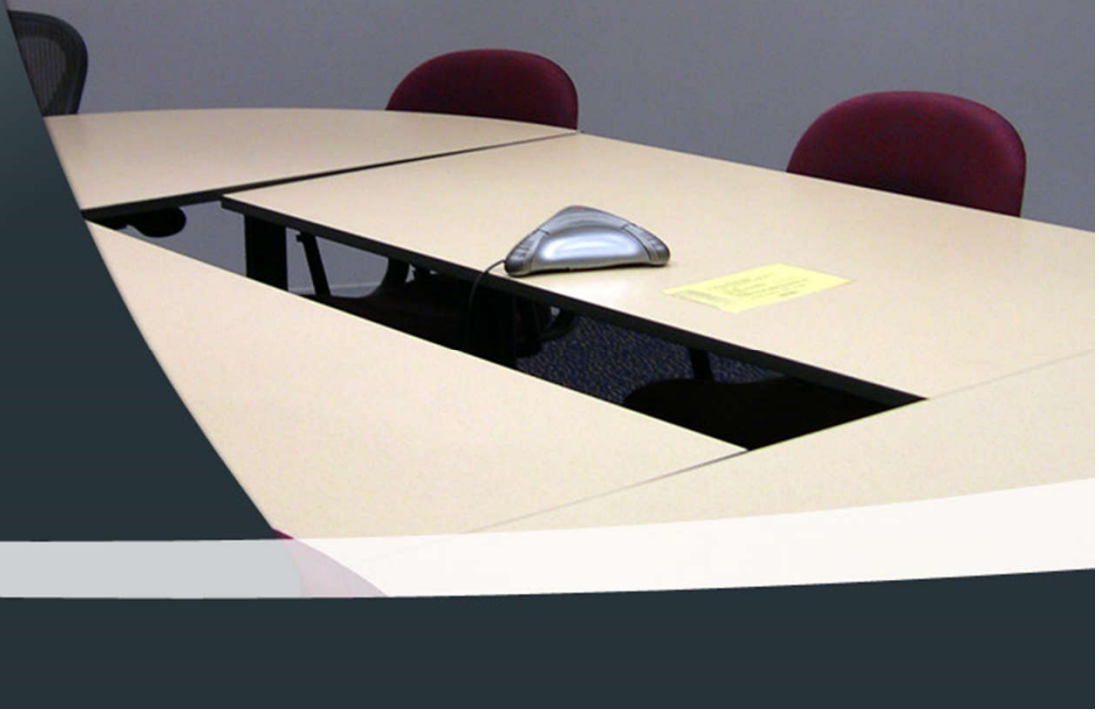
Rekayasa Perangkat Lunak



Modul 3

Understanding Requirement

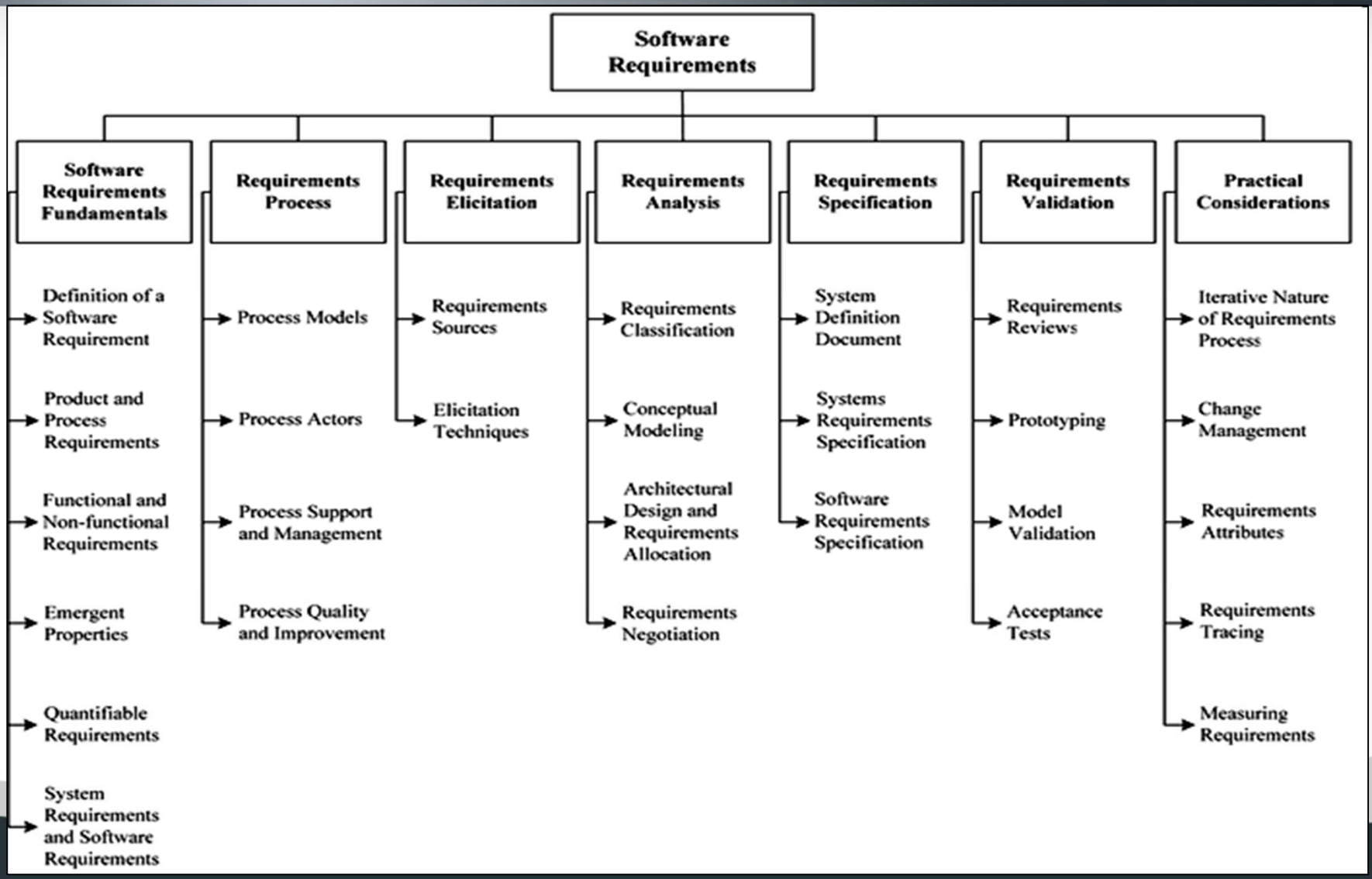
Disusun Oleh :
Irman Hariman, ST., MT.
Email : iirmanhariman@gmail.com

Materi Kuliah

Understanding Requirement :

- Requirement Engineering
- Developing Usecase
- Building analysis Model

Area Pengetahuan



Understanding Requirement

- Analisis kebutuhan perangkat lunak (software requirements analysis) merupakan aktivitas awal dari siklus hidup pengembangan perangkat lunak
- Tahap analisis adalah tahapan pengumpulan kebutuhan-kebutuhan dari semua elemen sistem perangkat lunak yang akan di bangun, Hal ini dibentuk :
  - Spesifikasi Kebutuhan Perangkat Lunak
  - Fungsi Perangkat Lunak Yang Dibutuhkan
  - Performansi (Unjuk Kerja) Sistem Perangkat Lunak
  - Penjadwalan Proyek
  - Identifikasi Sumber Daya (Manusia, Perangkat Keras Dan Perangkat Lunak Yang Dibutuhkan)
  - Taksiran Biaya Pengembangan Perangkat Lunak.

Fungsi dan Tujuan Analisis

- Kegunaannya adalah untuk memodelkan permasalahan dunia nyata agar dapat dimengerti.

- Tujuan aktivitas ini adalah untuk mengetahui ruang lingkup produk (product space) dan pemakai yang akan menggunakannya

Definisi Kebutuhan (Requirement)

- Menurut IEEE (The Institute of Electrical and Electronics Engineers) kebutuhan adalah :
  - Kondisi atau kemampuan yang diperlukan pemakai untuk menyelesaikan suatu persoalan, atau untuk mencapai sebuah objek.
  - Kondisi atau kemampuan yang harus dipenuhi oleh sistem, dalam arti memenuhi kontrak, standar, spesifikasi atau dokumen formal lain yang diinginkan.

Definisi Kebutuhan (Requirement) Lanjutan

- Tahap kebutuhan akan perangkat lunak dimulai dengan :
  - Kenali adanya sebuah permasalahan yang membutuhkan sebuah penyelesaian. Identifikasi sebuah permasalahan mungkin dapat dilakukan dengan berorientasi pada aplikasi, berorientasi pada bisnis, atau berorientasi pada kenaikan produktivitas (product improvement oriented).
  - Munculnya ide untuk membuat sebuah perangkat lunak baru (sebagai sebuah kemajuan).

Definisi Kebutuhan (Requirement) Lanjutan

7 hal yang perlu dipenuhi, yaitu :

- Inception : bagaimana memulai proyek pembangunan perangkat lunak. Secara umum banyak proyek dimulai ketika suatu bisnis butuh di identifikasi / pertumbuhan potensial pasar yang baru atau peningkatan layanan
- Elicitation : Tanyakan kepada pelanggan, pengguna tentang hal yang menjadi objektivitas suatu sistem atau produk. Fokusnya “lingkup permasalahan” (problem of scope). Kemudian temukan “pemahaman masalah” (problem of uderstanding), dan jika terjadi adanya perubahan kebutuhan dilain waktu maka perlu pemahaman problem of volatility. Pengumpulan informasi ini dilakukan dengan menggunakan metode wawancara, kuisioner, skenario dan prototyping.

Definisi Kebutuhan (Requirement) Lanjutan

3. Elaboration, informasi dari pelanggan (tahap inception dan elicitation) dilanjutkan Elaboration akan lebih mengarah dengan yang dijelaskan oleh pengguna.
4. Negotiation, perlu kejelasan mengenai sumberdaya bisnis guna mengetahui kejelasan dari kebutuhan sistem, termasuk biaya dan risiko bahkan konflik internal yang mungkin terjadi.
5. Specification, Hasil dari elicitation dianalisa dan direkam menggunakan teknik modeling. Spesifikasi perlu dicatat dalam sebuah dokumen secara model grafis, format matematis, atau dengan skenario, prototyping atau kombinasi semuanya agar jelas dan dapat ditelusuri ulang hal-hal apa sajakah yang harus dikembangkan

Definisi Kebutuhan (Requirement) Lanjutan

- Validation, Periksa kebutuhan yang ada dan sesuaikan dengan tujuan stakeholder terhadap sistem. Pastikan bahwa semua kebutuhan perangkat lunak tidak mengalami ambiguiti, ketidaksesuaian, kesalahan yang dapat dikenali dan diperbaiki yang suatu waktu akan mengganggu pelaksanaan proyek
- Management, Aktivitas ini dapat menolong tim proyek untuk melakukan identifikasi kebutuhan dan merasakan perubahan pada kebutuhan sistem yang dapat dikendalikan, serta terdapat tahapan pekerjaan.

Definisi Kebutuhan (Requirement) Lanjutan

- Langkah pertama dari aktivitas analisa sistem adalah analisa kebutuhan dengan mengidentifikasi kebutuhan dari pelanggan.
-

Contoh Pertanyaan Saat Wawancara

Beberapa pertanyaan yang dapat digunakan untuk membantu mengevaluasi informasi dari sistem :

- Adakah teknologi untuk membangun sistem?
- Batasan apa saja yang akan dialokasikan terhadap jadwal dan biaya?
- Pengembangan dan sumber daya apa saja yang dibutuhkan?

Jika sistem atau produk akan dijual ke pelanggan, ada beberpa pertanyaan yang bisa diajukan yaitu

- Bagaimana produk tersebut dapat bersaing dengan produk yang telah ada?
- Pasar apa saja yang potensial bagi produk yang akan dibangun?

Kebutuhan Perangkat Lunak

Tiga Jenis Kebutuhan Perangkat Lunak [IEE93] :

- Kebutuhan fungsional (functional requirement)

  Disebut juga kebutuhan operasional, yaitu kebutuhan yang berkaitan dengan fungsi atau proses transformasi yang harus mampu dikerjakan oleh perangkat lunak.

Sebagai contoh :

  - Perangkat lunak harus dapat menyimpan semua rincian data pesanan pelanggan.
  - Perangkat lunak harus dapat membuat laporan penjualan sesuai dengan periode waktu tertentu.
  - Perangkat lunak harus mampu menyajikan informasi jalur pengiriman barang terpendek.

Kebutuhan Perangkat Lunak Lanjutan

- Kebutuhan antarmuka (interface requirement)

  Kebutuhan antarmuka yang menghubungkan perangkat lunak dengan elemen perangkat keras, perangkat lunak, atau basis data.

Sebagai contoh:

- Perangkat untuk memasukkan data dapat berupa keyboard, mouse atau scanner.
- Akses ke basisdata menggunakan ODBC (Open Database Connectivity).

Kebutuhan Perangkat Lunak Lanjutan

- Kebutuhan unjuk kerja (performance requirement)

  Kebutuhan yang menetapkan karakteristik unjuk kerja yang harus dimiliki oleh perangkat lunak, misalnya: kecepatan, ketepatan, frekuensi.

Sebagai contoh :

- Perangkat lunak harus bisa mengolah data sampai 1 juta record untuk tiap transaksi.
- Perangkat lunak harus dapat digunakan oleh multiuser sesuai dengan otoritas yang diberikan pada user.
- Waktu tanggap penyajian informasi maksimal selama satu menit.

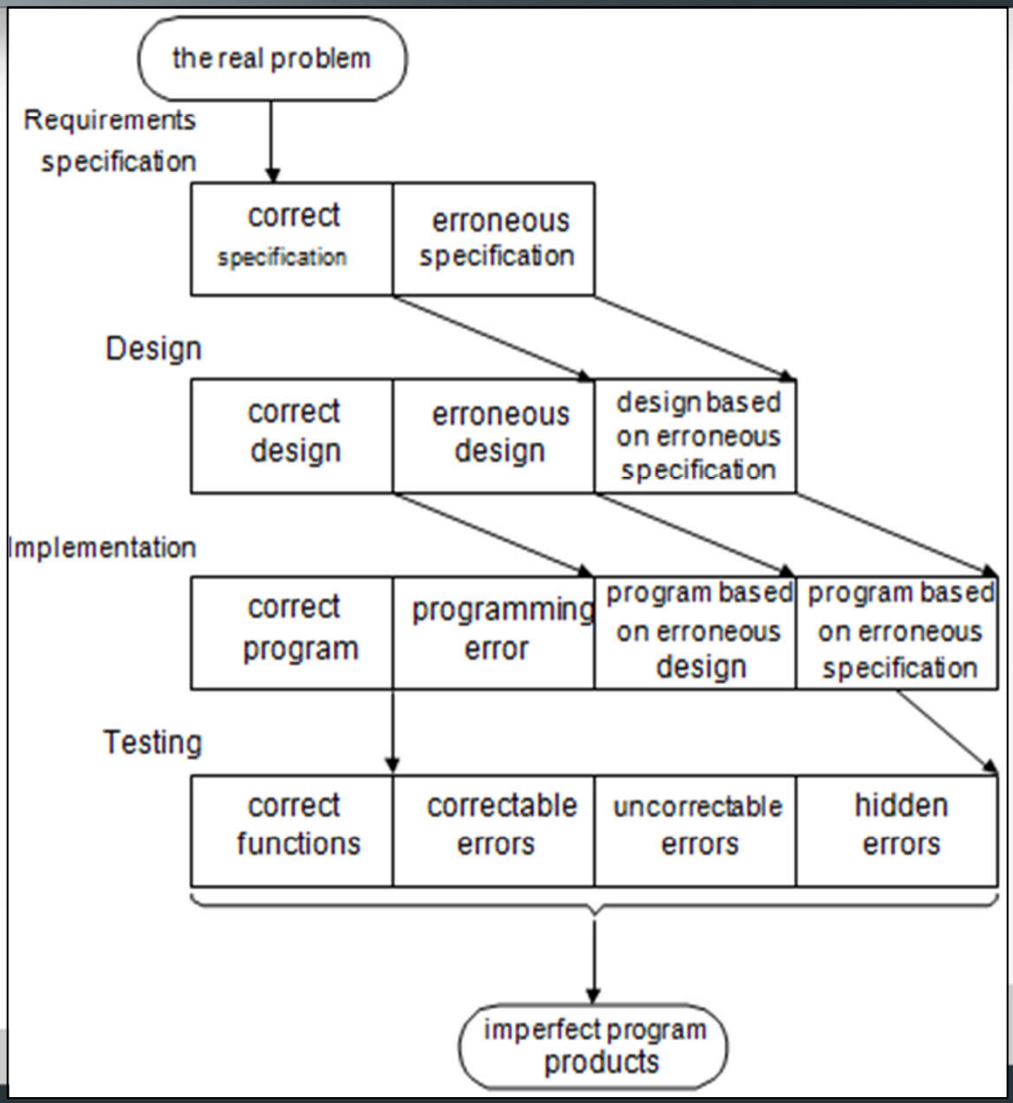
Pentingnya “Kebutuhan”

- Pentingnya pendefinisian kebutuhan karena sangat mempengaruhi sukses atau gagalnya pelaksanaan pengembangan perangkat lunak.

- Hasil survey DeMarco, 56% kegagalan proyek pengembangan perangkat lunak dikarenakan ketidaklengkapan pendefinisian kebutuhan dari perangkat lunak tersebut.

Pentingnya "Kebutuhan" Lanjutan

Dampak kesalahan kumulatif akibat kesalahan dalam pendefinisian kebutuhan



Pentingnya "Kebutuhan" Lanjutan

Kesalahan penentuan kebutuhan akan memberikan dampak [DAV93]:

- Perangkat lunak yang dihasilkan tidak akan memenuhi kebutuhan pemakai yang sebenarnya.
- Interpretasi kebutuhan yang berbeda-beda sehingga dapat menyebabkan ketidaksepakatan antara pelanggan dan pengembang, menyia-nyiakan waktu dan biaya, dan mungkin akan menghasilkan perkara hukum.
- Pengujian kesesuaian perangkat lunak dengan kebutuhan yang dimaksud tidak akan mungkin dilaksanakan dengan sesungguhnya.
- Waktu dan biaya akan terbuang percuma untuk membangun sistem yang salah.

Emergent Properties

- Muncul ketika sejumlah entitas sederhana beroperasi di sebuah lingkungan, membentuk perilaku yang lebih kompleks sebagai sebuah kolektifitas.
- Mungkin sesuatu yang sangat dapat diprediksi, atau tidak dapat diprediksi, bahkan belum pernah terjadi sebelumnya.
- Kebutuhan yang tak dapat diatas oleh komponen tunggal.

  Contoh Emergent Properties : kebutuhan throughput untuk call center akan bergantung pada bagaimana sistem telepon, sistem informasi, dan operator dimana semua berinteraksi dalam kondisi operasi aktual
  - Kebutuhan perangkat lunak harus dinyatakan sejelas mungkin, dan bila perlu secara kuantitatif.
  - Contoh kebutuhan terukur : software call center harus meningkatkan throughput pusat sebesar 20%

Requirement Engineering

- Analisis kebutuhan perangkat lunak (software requirements analysis) merupakan aktivitas awal dari siklus hidup pengembangan perangkat lunak

Analisis kebutuhan dapat diartikan sebagai berikut :

- Proses mempelajari kebutuhan pemakai untuk mendapatkan definisi kebutuhan sistem atau perangkat lunak [IEE93].
- Proses untuk menetapkan fungsi dan unjuk kerja perangkat lunak, menyatakan antarmuka perangkat lunak dengan elemen-elemen sistem lain, dan menentukan kendala yang harus dihadapi perangkat lunak [PRE01].

Tujuan pelaksanaan analisis kebutuhan adalah :

- Memahami masalah secara menyeluruh (komprehensif) yang ada pada perangkat lunak yang akan dikembang seperti ruang lingkup produk perangkat lunak (product space) dan pemakai yang akan menggunakannya.
- Mendefinisikan apa yang harus dikerjakan oleh perangkat lunak untuk memenuhi keinginan pelanggan.

Developing Usecase

- Use case membantu mengidentifikasi objek atau kelakuan sistem dan membantu mendesain antarmuka dan spesifikasi kode, juga berfungsi sebagai rencana pengujian sistem.
- Use case juga berfungsi sebagai garis pokok untuk mempersiapkan semua dokumentasi pengguna dan sistem, juga sebagai alat untuk latihan pengguna.
- Use case diawali atau dipicu oleh pengguna eksternal yang dinamakan actor/pelaku

Menggambarkan fungsionalitas yang diharapkan (diusulkan) dari sebuah sistem. Yang ditekankan adalah “apa” yang diperbuat sistem, dan bukan “bagaimana”.

- Menggambarkan kebutuhan sistem dari sudut pandang user.
- Memfokuskan pada proses komputerisasi (automated processes).
- Menggambarkan hubungan antara use case dan actor

Pemodelan Use Case

- Dua alat utama yang digunakan saat menyajikan pemodelan use case :
- Use case diagram adalah diagram yang menggambarkan interaksi antara sistem dengan sistem eksternal dan pengguna. Dengan kata lain, secara grafis menggambarkab siapa yang akan menggunakan sistem dan dengan cara apa pengguna mengharapkan untuk berinteraksi dengan sistem
- Use case Narative adalah deskripsi tekstual kegiatan bisnis dan bagaimana pengguna akan berinteraksi dengan sistem untuk menyelesaikan suatu tugas

Tipe Actor

- Primary business actor : stakholder yang terutama mendapatkan keuntungan dari pelaksanaan use case (contoh : karyawan dengan menerima gaji untuk periode tertentu)
- Primary System actor : stakholder yang secara langsung berhadapan dengan sistem untuk menginisiasi atau memicu kegiatan suatu sistem. (contoh operator telepon yang memberi bantuan kepada pelanggan, kasir bank yang memproses transaksi
- External server actor : stakholder yang melayani kebutuhan pengguna use case (contoh : biro kredit yang memiliki kuasa atas perubahan kartu kredit)

- External Receiving Actor : stakholder yang bukan pelaku utama , tapi menerima nilai yang terukur atau teramati (output) dari use case (contoh : gudang menerima paket permintaan untuk menyiapkan pengiriman sesudah seorang pelanggan memesannya)
- Temporal Event : kejadian sistem yang diicu dengan waktu (contoh : sistem billing untuk perusahaan kartu kredit secara otomatis mencetak tagihan pada hari ke lima dalam bulan itu (tanggal billing);Billing PLN; Billing PAM , bank Merekonsiliasi transaksi tiap hari pada jam 5 sore) yang menjadi actor/ pelaku disini adalah waktu

- Association (gabungan) adalah hubungan antara aktor/pelaku dengan usecase dimana terjadi interaksi diantara mereka
- Extend use case adalah usecase yang terdiri dari langkah yang diekstraksi dari usecase yang lebih kompleks untuk menyederhanakan masalah orisinil dan karena itu memperluas fungsinya
- Abstract Use case adalah usecase yang mengurangi redudandi antara dua atau lebih usecase lain dengan menggabungkan langkah-langkah yang biasa ditemukan pada usecase tersebut
- Depens On adalah hubungan ketergantungan antara usecase dimana sebuah use case tidak bisa dieksekusi sebelum mengeksekusi use case yang lain

- Inheritance pada saat dua atau lebih actor/pelaku berbagi kelakuan umum (mereka dapat menginisiasi use case yang sama.maka yang paling baik adalah mengeksploitasi kelakuan umum dan menetapkannya ke actor/pelaku abstrak baru untuk mengurangi komunikasi redudan dengan sistem

- Karakteristik pendekatan terstruktur adalah sebagai berikut:
- Penekanan pada sesuatu yang harus dikerjakan (algoritma pemacahan masalah), dimulai dari menerima input, melakukan proses, menghasilkan output.
- Program berukuran besar dipecah menjadi program-program yang lebih kecil.
- Kebanyakan fungsi/prosedur berbagi data global
- Data bergerak secara bebas dalam sistem, dari satu fungsi ke fungsi lain yang terkait
- Fungsi-fungsi mentransformasi data dari satu bentuk ke bentuk yang lain.
- Pendekatan yang digunakan yaitu pendekatan atas ke bawah (top down approach)

- Karakteristik pendekatan objek adalah sebagai berikut:
  - Pendekatan lebih pada data dan bukanya pada prosedur/fungsi.
  - Program besar dibagi pada sesuatu yang disebut objek-objek
  - Struktur data dirancang dan menjadi karakteristik dari objek-objek.
- Fungsi-fungsi yang mengoperasikan data tergabung dalam suatu objek yang sama.
- Data tersembunyi dan terlindung dari fungsi/prosedur yang ada diluar
- Objek-objek dapat saling berkomunikasi dengan saling mengirim *message* satu sama lain.
- Pendekatan yang digunakan yaitu pendekatan bawah ke atas (bottom up approach)