PERTEMUAN 8

ORIENTASI OBJEK (LANJUTAN)

A. TUJUAN PEMBELAJARAN

Pada pertemuan ini dijellaskan tentang pengertian dan proses analisa sistem. Dari pertemuan ini diharapkan mahasiswa mampu mendeskripsikan apa itu analisa sistem.

B. URAIAN MATERI

1. Identifikasi Objek

Dalam Identifikasi objek ada beberapa metedelogi berorientasi objek yang mempunyai 3 karakteristik utama:

a. Encapsulation (Pengkapsulan)

1) Enkapsulasi adalah dasar dari parameter lingkup program untuk data yang diolah.
2) Data dan proses atau fungsi disimpan bersama sebuah objek sehingga tidak ada prosedur atau fungsi lain yang diperkenalkan dari luar bisa mengunjunginya.
3) Lindungi data agar tidak terpengaruh oleh proses atau objek lain (kecuali proses) berada di dalam objek itu sendiri.

b. Inheritance (Pewarisan)

1) Pewarisan adalah teknik untuk menunjukkan anak suatu benda akan langsung mewarisi data / atribut dan proses dari induknya. Properti dan metode objek induk diteruskan ke Anak objek, dll.
2) Warisan berarti atribut dan operasi dimiliki antara kelas terkait Hirarki.
3) Suatu kelas dapat menentukan kategori terlebih dahulu, lalu menentukan dibagi secara khusus ke dalam sub-kategori. Setiap subclass memiliki hubungan atau mewarisi semua properti yang dimiliki oleh kelas induk, dan ditambah itu memiliki atribut unik. Objek kelas dapat mendefinisikan atribut dan layanan kelas objek lain.

4) Inheritance menjelaskan generalisasi suatu kelas

Contohnya:

a) Mobil dan sepeda motor adalah subkategori kendaraan bermotor.
b) Kedua subclass ini mewarisi atribut yang dimiliki oleh kendaraan bermotor, dengan mesin, dan bisa berjalan.
c) Kedua subkategori ini memiliki karakteristiknya masing-masing beda, seperti jumlah roda dan kemampuan berjalan.

c. Polymorphism (Polimorfisme)

1) Polimorfisme adalah sebuah konsep, yang menunjukkan bahwa ada sesuatu orang yang sama dapat memiliki bentuk dan perilaku yang berbeda.
2) Polimorfisme berarti operasi yang sama dimungkinkan ada perbedaan antara kelas yang berbeda.
3) Kemampuan untuk mengeksekusi metode untuk objek yang berbeda tanggapan yang tepat untuk pesan yang sama.
4) Pilih metode yang sesuai menurut kategori mana objek harus dibuat.

2. Model Desain

OOAD yang dikemas mencakup penggunaan metode objek untuk analisis dan desain sistem. Object-Oriented Analysis (OOA) adalah metode analisis yang meneliti persyaratan (persyaratan / persyaratan yang harus dipenuhi oleh sistem) dari perspektif kelas dan objek yang ditemui di dalam perusahaan. Pada saat yang sama, desain berorientasi objek (OOD) adalah metode memandu arsitektur perangkat lunak berdasarkan operasi objek sistem atau subsistem. Ada beberapa konsep dasar di OOAD, yaitu :

a. Objek (object)

Objek adalah objek fisik dan konseptual di sekitar kita. Beberapa contoh objek, seperti perangkat keras, perangkat lunak, dokumen, orang, konsep, dll. Misalnya, untuk tujuan pemodelan, eksekutif memperlakukan karyawan, gedung, departemen, arsip, dan keuntungan perusahaan sebagai objek. Pada saat yang sama, teknisi otomotif akan mempelajari ban, pintu, mesin, kecepatan dan nomor tertentu. Bahan bakar adalah obyeknya. Contoh lain adalah insinyur perangkat lunak memperlakukan tumpukan,

antrian instruksi, jendela, dan kotak centang sebagai objek.

State dari sebuah objek adalah kondisi dari objek atau sekumpulan status yang menggambarkan objek tersebut. Misalnya, status rekening tabungan dapat mencakup saldo saat ini, status jam tangan adalah rekor saat ini, dan status bola lampu adalah "hidup" atau "mati". Status diwakili oleh nilai properti objek.

Atribut adalah Nilai internal objek, yang mencerminkan karakteristik objek, kondisi sesaatnya, hubungannya dengan objek lain, dan identifikasinya. Perubahan status tercermin melalui perilaku objek.

Behaviour atau Perilaku suatu objek mendefinisikan perilakunya (perilaku) dan bagaimana reaksinya. Perilaku ditentukan oleh rangkaian semua atau lebih operasi yang dapat dilakukan oleh objek itu sendiri. Perilaku suatu objek direfleksikan oleh antarmuka, layanan, dan metodenya.

Interface adalah Akses ke pintu layanan.

Service adalah fungsi yang dapat dilakukan oleh objek.

Method adalah mekanisme internal dari objek yang mencerminkan perilaku dari objek tersebut.

b. Kelas (Class)

Class adalah definisi umum sekelompok objek serupa (pola, templat, atau cetak biru). Kelas menentukan perilaku (perilaku) dan properti objek. Kelas adalah abstraksi entitas di dunia nyata. Dan objek adalah contoh kelas.

Misalnya, atribut hewan adalah hewan berkaki empat dan memiliki ekor. Tingkah lakunya adalah makan dan tidur. Sedangkan hewan contoh adalah kucing, gajah dan kuda.

c. Kotak Hitam (Black Boxes)

Sebuah Objek adalah kotak hitam. Konsep ini menjadi dasar realisasi objek. Dalam operasi OO, hanya pengembang (pemrogram, desainer, analis) yang dapat memahami detail proses yang terdapat dalam kotak hitam, dan pengguna tidak perlu mengetahui operasi yang sedang dilakukan, tetapi yang terpenting adalah mereka dapat menggunakan objek untuk memproses mereka Permintaan.

Encapsulation, Proses menyembunyikan detail implementasi suatu objek. Satu-satunya cara untuk mengakses data objek adalah melalui antarmuka.

Antarmuka ini dapat melindungi keadaan internal objek dari "gangguan" eksternal. Oleh karena itu, objek direpresentasikan sebagai kotak hitam untuk menerima dan mengirim pesan. Dalam OOP, kotak hitam berisi kode (instruksi yang dapat dimengerti komputer) dan data (informasi yang memerintahkan operasi padanya). Dalam OOP, kode dan data ditempatkan di "objek", dan konten objek disembunyikan, yaitu objek. Pengguna subjek tidak perlu mengetahui isi kotak. Untuk berkomunikasi dengan objek tersebut, diperlukan sebuah pesan.

Message adalah Proses menyembunyikan detail implementasi suatu objek. Satu-satunya cara untuk mengakses data objek adalah melalui antarmuka. Antarmuka ini dapat melindungi keadaan internal objek dari "gangguan" eksternal. Oleh karena itu, objek direpresentasikan sebagai kotak hitam untuk menerima dan mengirim pesan. Dalam OOP, kotak hitam berisi kode (instruksi yang dapat dimengerti komputer) dan data (informasi yang memerintahkan operasi padanya). Dalam OOP, kode dan data ditempatkan di "objek", dan konten objek disembunyikan, yaitu objek. Pengguna subjek tidak perlu mengetahui isi kotak. Untuk berkomunikasi dengan objek tersebut, diperlukan sebuah pesan.

d. Asosisasi dan Agregasi

Asosiasi adalah hubungan yang berarti antara banyak objek. Asosiasi direpresentasikan dengan menghubungkan garis antar objek. Contoh: asosiasi antara objek mobil dan seseorang. Sebuah mobil dapat dimiliki oleh satu atau beberapa orang, dan satu orang dapat memiliki nol, satu atau lebih mobil

Asosiasi antara Karyawan tempat kerja. Seorang karyawan bekerja di satu unit kerja. Sebuah unit kerja dapat memiliki beberapa karyawan.

Agregasi adalah Bentuk asosiasi khusus yang menjelaskan bahwa semua bagian suatu objek adalah bagian dari objek lain. Contoh: Kopling dan piston adalah bagian dari mesin. Mesin, roda, dan bodi adalah bagian dari mobil. Tanggal, bulan dan tahun adalah bagian dari tanggal lahir. Padahal, tanggal lahir, nama, alamat, dan jenis kelamin menjadi bagian dari identitas seseorang.

3. Spesifikasi Interface Objek

Desain antarmuka adalah proses untuk menentukan bagaimana sistem berinteraksi dengan entitas eksternal (seperti pelanggan, pemasok, dan sistem lainnya). Tentukan bagaimana pengguna berinteraksi dengan sistem (bagaimana memberikan masukan dan mendapatkan keluaran, dan bagaimana sistem menerima dan menjalankan perintah).

a. User Interface Problems

Dalam pembuatan interface pada system, biasanya ada masalah-masalah yang dialami oleh user. Berikut ini beberapa masalah yang sering tditemui:

1) Penggunaan istilah komputer yang ekstensif.
2) Desain tidak jelas
3) Tidak dapat membedakan tindakan Pilih ("Apa yang harus saya lakukan lanjut").
4) Bukan solusi untuk masalah tersebut konsisten.
5) Desain tidak konsisten.

b. Prinsip Desain Interface

Untuk membuat desain interface yang nyaman saat digunakan, terdapat beberapa prinsip yang bisa dijadikan acuan Dari buku "General Principles Of UI Design" ditulis oleh Deborah J. Mayhew, ada 17 prinsip desain antarmuka secara umum.

1) User Compatibility (Kompatibilitas Pengguna).
   Artinya, karena pengguna yang berbeda mungkin memiliki persyaratan tampilan yang berbeda, tampilan disesuaikan dengan tipikal pengguna. Misalnya, jika aplikasi cocok untuk anak-anak, jangan gunakan bahasa atau tampilan dewasa.
2) Product Compatibility (Kompatibilitas Produk)
   Produk aplikasi akhir juga harus sesuai untuk orang awam dan ahli serta memiliki tampilan yang sama / mirip.
3) Task Compatibility (Kompabilitas Tugas)
   Fungsi tugas yang ada harus sesuai dengan tampilannya. Misalnya untuk opsi laporan, user akan langsung menjelaskan laporan yang akan ditampilkan.
4) Work Flow Compatibility (Kompabilitas Alur Kerja)

Aplikasi dapat dijalankan untuk berbagai pekerjaan dalam satu tampilan, dengan pertimbangan tidak terlalu kelebihan beban.

5) Consistency (Konsisten)
   Desainnya harus konsisten. Misal: jika menggunakan kata "Save" artinya "Simpan" kemudian terus digunakan kata itu dan jangan sampai berubah..
6) Familiarity (Keakraban)
   Faktor kebiasaan masih mudah diidentifikasi. Seperti Ikon floppy disk akan lebih familiar digunakan sebagai icon Simpan perintah.
7) Simplicity (Kesederhanaan)
   Aplikasi harus mengumpulkan semua yang ada di dalamnya lebih sederhana dan lebih bermakna.
8) Direct Manipulation (Manipulasi Langsung)
   Aktivitas tersebut disajikan langsung kepada pengguna (user) sehingga komputer melakukan aktivitas tersebut saat pengguna memberikan instruksi langsung pada layar komputer. Contoh: Untuk membuat tulisan menjadi tebal, tekan saja ctrl + B.
9) Control (Kontrol)
   Kontrol penuh pengguna, pengguna biasa biasanya tidak membutuhkan terlalu banyak aturan
10) WYSIWYG (What You See Is What You Get)
    Ada korespondensi satu-ke-satu antara informasi di layar dan informasi pada hasil cetak atau file. Buat agar terlihat seperti pengguna asli dan pastikan fungsinya berfungsi seperti yang diharapkan.
    Contoh: Ketik huruf A, huruf A output dilayar, dan jika ada maka hasilnya juga huruf A.
11) Flexibility (Fleksibelitas)
    Sistem memiliki kemampuan untuk mencapai tujuan dengan berbagai cara. Sistem dapat beradaptasi dengan keinginan pengguna (bukan pengguna yang harus beradaptasi dengan kerangka desain sistem)
12) Responsiveness (Responsivitas)
    Konten yang ditampilkan harus responsif. Misalnya, harap tunggu 68% untuk tampilan layar ... atau bilah pemuatan, dll.
13) Invisible Technology (Teknologi Tak Terlihat)
    Tidak masalah algoritma mana yang pengguna tahu untuk digunakan.

Misalnya, untuk mengurutkan pengguna, tidak perlu mengetahui algoritme yang digunakan oleh programmer (pengurutan maks, pengurutan gelembung, pengurutan cepat, dll.).

14) Robustness (Kekokohan)
Dapat mengakomodasi kesalahan pengguna tanpa menampilkan kesalahan, apalagi mogok. Misalnya, jika pengguna memasukkan format email yang salah, masalah dapat diselesaikan dan umpan balik diberikan.
15) Protection (Perlindungan)
Lindungi pengguna dari kesalahan umum. Misalnya dengan memberikan fungsi return atau undo.
16) Ease of Learning (Mudah Dipelajari)
Mudah untuk mempelajari aplikasi pemula (awam). Ini akan memotivasi pengguna untuk menggunakannya.
17) Ease of Use (Mudah Digunakan)
membuat sistem yang mudah digunakan untuk semua kategori pengguna (awam atau ahli).

4. Proses Desain Interface

Untuk membuat Interface ada beberapa langkah yang bisa digunakan agar lebih efektif. Berikut ini adalah beberapa langkahnya:

a. Menggunakan User Sekenario

Berikut ini adalah contoh sekenario untuk “Melihat Data Diri Mahasiswa”.

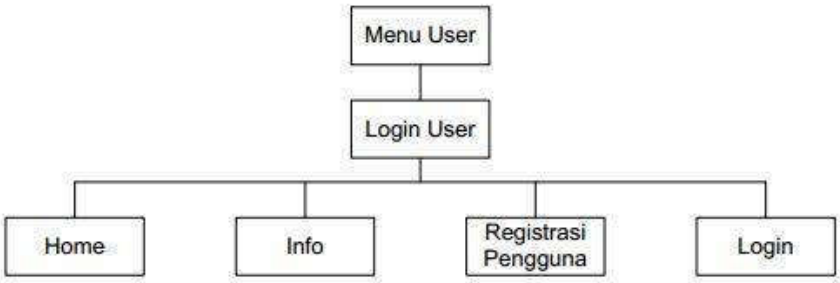
Use case Melihat Data Diri Mahasiswa	
Tujuan	Mengijinkan administrator untuk melihat data diri mahasiswa
Aktor	Administrator
Kondisi awal	Login tervalidasi dan valid
Skenario utama	1. Administrator memilih menu data diri mahasiswa 2. Sistem menampilkan daftar mahasiswa, untuk dilihat data dirinya
Skenario alternatif	1. Jika data diri mahasiswa yang dipilih masih kosong, maka sistem akan menunjukkan pesan “data diri masih kosong”. 2. Jika data diri mahasiswa yang dipilih masih belum lengkap, maka sistem akan menampilkan pesan “data diri masih belum lengkap”.
Kondisi akhir	Sistem menampilkan data diri mahasiswa sesuai dengan identitas mahasiswa yang dipilih.

Gambar 26. Contoh User Sekenario

b. Membuat Struktur Desain Interface

Desain struktur antarmuka mendefinisikan komponen dasar antarmuka dan bagaimana mereka bekerja sama untuk menyediakan fungsi bagi pengguna.

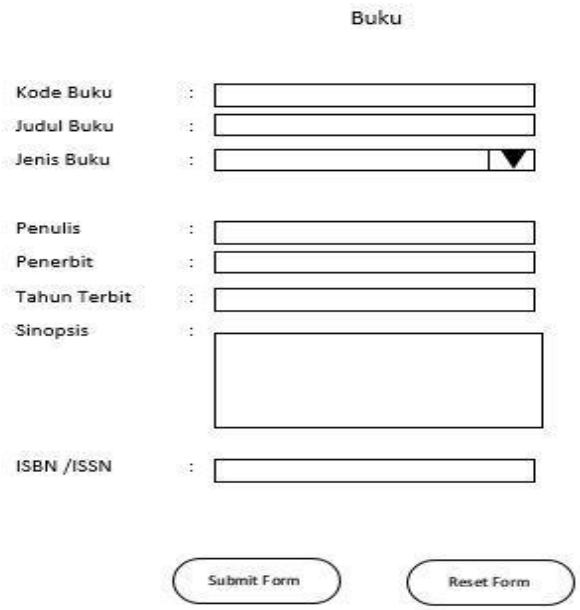
Diagram Struktur Antarmuka (ISD) digunakan untuk menunjukkan bagaimana sistem terkait menggunakan semua layar, formulir dan laporan, dan bagaimana pengguna berpindah dari satu ke yang lain-lain.



Gambar 27. Contoh Struktur Desain Interface

c. Membuat Prototype

Berikut adalah contoh prototype interface Insert Buku sebuah program.



Gambar 28. Prototype Interface Insert Buku

d. Evaluasi
   1) Evaluasi heuristik: perbandingan dengan prinsip dasar UI
   2) Evaluasi latihan: bertemu dengan pengguna
   3) Pengujian kegunaan

C. SOAL LATIHAN/TUGAS

1. Sebutkan dan jelaskan tahapan untuk membuat desain interface!
2. Sebutkan dan jelaskan metedologi berorientasi objek!
3. Buatlah desain prototype desain interface selain yang ada di modul!
4. Apa saja masalah yang sering dialamin user interface?
5. Apa itu Black Box? Jelaskan dengan gaya Bahasa Anda!

D. REFERENSI

Grady Booch, 1991, Object-Oriented Analysis and Design with Application,Benjamin/Cummings.

Dennis, Alan., Barbara Halley Wixom and Roberta M. Roth. 2012. System Analysis and Design 5 th Edition. John Willey and Sons, Inc. New Jersey

Satzinger, John., Robert Jackson and Stephen Burd. 2010. System Analysis and Design in Changing World 5 th Edition. Cengange Learning. Boston.