**Modul Perkuliahan Sesi 13**

**Interaksi Manusia dan Komputer**

**Perancangan User Interface**

Merancang sebuah interface untuk pengguna adalah hal sangat penting dalam mengembangkan sebuah sistem informasi. Untuk merancang interface harus memperhatikan faktor psikologis serta keindahan sehingga hasilnya mampu menarik perhatian pengguna. Ada 2 jenis metafora yang secara luas digunakan dalam merepresentasikan sebuah user interface. Pertama adalah ide bahwa pengguna sedang melakukan dialog dengan sistem. Kedua adalah ide bahwa pengguna secara langsung melakukan manipulasi terhadap objek yang muncul pada layar.

User Interface?

Interaksi pengguna dengan sistem dapat dilakukan dengan berbagai cara, antara lain :

* Membaca dan menafsirkan inforasi mengenai bagaimana menggunakan system memberikan perintah untuk sistem untuk melakukan tugas tertentu
* Memasukkan kata atau angka ke dalam sistem sebagai suatu data
* Membaca dan menafsirkan hasil yang dihasilkan oleh sistem dalam tampilan layar atau laporan tercetak
* Menanggapi dan mengoreksi kesalahan

Metafora

Adalah istilah yang digunakan secara kiasan untuk menjelaskan sesuatu namun tidak dapat diartikan secara harfiah. Dalam interaksi manusia dengan komputer ada 2 jenis pendekatan metafora, yaitu :

1. Metafora dialog
2. Metafora manipulasi langsung

Metafora Dialog

Komunikasi yang terjadi antara manusia dengan komputer dianggap semacam dialog. Tidak ada pembicaraan secara nyata namun manusia memberikan pesan ke komputer kemudian komputer merespon pesan tersebut. Komputer merespon dengan berbagai cara untuk kemudian mendorong manusia merespon kembali. Demikian seterusnya.

Bentuk skema dialog antara manusia dengan komputer digambarkan sebagai berikut :



Gambar 1. Skema bentuk dialog manusia dengan computer



Tabel 1. Tipe Message pada Dialog



Perancangan User Interface

1. Interface Design
2. User Interface Design (Three Golden Rules)
3. User Interface Analysis and Design
4. Data Design
5. Component Level Design

Elemen-elemen perancangan interface untuk perangkat lunak menjelaskan Bagaimana arus informasi masuk dan keluar dari sistem, dan bagaimana arus informasi tersebut berkomunikasi diantara komponen-komponen yang didefinisikan sebagai bagian dariarsitektur.

Fokus Interfaces Design > 3 Area

1. Inter-modular interface design

2. External interface design

3. Human-Computer Interface (HCI) design

1. Internal : merupakan desain interface antarmodul dalam PL yang dikendalikan oleh data yang harus mengalir di antara modul-modul. Aliran transformasi dalam DFD merupakan pijakan utama dalam desain ini selain kemampuan bahasa pemrograman.

2. Eksternal: merupakan interface untuk entitas eksternal (tidak termasuk manusia), misalnya sensor pada PL Safehome.

3. Manusia – Mesin: merupakan interface antara manusia dengan PL (Human – Computer Interface). Interface ini memiliki tantangan besar karena berkaitan dengan pengguna dengan berbagai karakter yang lebih sulit untuk dipelajari. Terdapat tiga kategori pedoman desain HCI sbb.

Desain HCI

1. Interaksi Umum

* Format konsisten
* Perlindungan thd kegagalan
* Berikan petunjuk singkat (tools tips) pada setiap button / ikon / nama
* Berikan umpan balik
* Konfirmasi untuk aksi destruktif (misal Hapus)
* Ijinkan pembatalan (misal Undo)
* Kurangi jumlah informasi yang harus diingat
* Efisiensi dalam dialog, gerakan (tangan), pemikiran
* Kategorikan aktivitas sejenis dan posisinya di layar
* Sediakan Help yang sensitif konteks
* Gunakan perintah dan nama2 yang pendek

2. Input

* Minimalkan jumlah aksi input (combo box, list, dsb.)
* Konsisten
* Berikan kemungkinan kustomisasi input (utk advance user)
* Mode input harus fleksibel (mouse / keyboard)
* Non-aktifkan button/ ikon yang tidak relevan dengan aksi
* Berikan kesempatan untuk mengontrol aliran interaksi (mengubah, membetulkan, mengulang)
* Sediakan Help
* Jangan meminta aktivitas manual (perhitungan, tanggal, waktu, dsb) bila dapat dilakukan oleh PL

3. Output

* Tampilkan informasi yang relevan dg konteks
* Jangan membanjiri pemakai dg informasi
* Gunakan label, singkatan, warna yg standar dan konsisten
* Peliharalah konteks visual saat pengguna melakukan zoom-in / zoom-out
* Pesan kesalahan harus memiliki arti yang jelas
* Gunakan variasi huruf, indentasi, pengelompokan untuk memudahkan pemahaman
* Gunakan jendela untuk tipe-tipe informasi yang berbeda
* Gunakan tampilan alami (bukan angka / grafik) bila memungkinkan
* Geografi layar dioptimalkan shg tidak ada jendela yang „hilang‟ / sulit ditemukan
* Berikan kemungkinan kustomisasi output (utk advance user)
* Jangan ada informasi / data yang tidak lengkap / hilang sebagian

Three golden rules – Theo Mandel

1. Menempatkan user di dalam kontrol

* “ Apa yang saya inginkan adalah sebuah sistem yang membaca pikiran saya. Dia tahu apa yang ingin saya lakukan sebelum saya butuhkan dan membuat mudah saya untuk melakukannya”

2. Mengurangi muatan memori user,

* Semakin banyak user harus mengingat, semakin banyak interaksi kesalahan dengan sistem
* Sistem seharusnya mengingat

3. Membuat Interface yang Konsisten

Tempatkan pengguna dalam Control

* Mendefinisikan interaksi dalam sedemikian rupa sehingga pengguna tidak dipaksa melakukan tindakan yang tidak perlu atau tidak diinginkan
* Menyediakan untuk interaksi yang fleksibel (pengguna memiliki berbagai preferensi)
* Memungkinkan interaksi pengguna menjadi interruptible dan reversibel
* Merampingkan interaksi sebagai tingkat keterampilan meningkat dan memungkinkan kustomisasi interaksi
* Menyembunyikan internal teknis dari pengguna biasa
* Desain untuk interaksi langsung dengan objek yang muncul di layar

Mengurangi Muatan Memori User

* Mengurangi tuntutan pada memori jangka pendek pengguna (exp. Menyediakan isyarat visual)
* Menetapkan default bermakna ("reset" pilihan harus tersedia)
* Mendefinisikan intuitif potongan pendek (mudah diingat)
* Visual tata letak antarmuka pengguna harus didasarkan pada familiar kiasan dunia nyata
* Mengungkapkan informasi secara progresif

Make Interface Consistent

* Memungkinkan pengguna untuk menempatkan tugas saat dalam konteks yang bermakna
* Menjaga konsistensi di sebuah famili aplikasi
* Jika model interaksi masa lalu telah menciptakan harapan pengguna, tidak membuat perubahan kecuali ada alasan yang baik untuk melakukannya

User Interface Analysis and Design

* The models
	+ Para Engineer menetapkan model pengguna
	+ Software engineer menghasilkan model desain
	+ Pengguna akhir mengembangkan persepsi sistem
	+ Pelaksana menciptakan model implementasi
* The process
	+ Pengguna, tugas, dan analisis lingkungan dan pemodelan
	+ Desain antarmuka
	+ Konstruksi antarmuka
	+ Antarmuka validasi

The Models

* User model
	+ Profil Pengguna Akhir:
		- Awam
		- Knowledgeable, Pengguna intermittent,
		- Knowledgeable, Pengguna Sering
* Design model
	+ Menggabungkan data, Arsitektur, Interface, dan Representasi prosedural dari perangkat lunak.
* User's model or system perception
	+ user's mental image of system
* Implementation model
	+ tampilan dan nuansa antarmuka dan media pendukung

The Process – User Analysis

* Memahami yang akhir-pengguna
* Apa yang mungkin memotivasi dan memuaskan mereka
* Bagaimana mereka dapat dikelompokkan menjadi pengguna class atau profil yang berbeda
* Apa model tampilan mereka dari sistem itu
* Bagaimana user interface harus ditandai untuk memenuhi kebutuhan mereka

The Process – Task Analysis and Modeling

* Studi software engineer tugas para pengguna manusia harus menyelesaikan untuk mencapai tujuan mereka di dunia nyata tanpa komputer dan memetakannya tersebut ke dalam satu set sama tugas-tugas yang harus dilaksanakan dalam konteks user interface
* Studi engineer perangkat lunak spesifikasi untuk solusi berbasis komputer yang ada dan berasal satu set tugas yang akan mengakomodasi model pengguna, desain model, dan persepsi sistem.
* Software engineer dapat merancang pendekatan berorientasi objek dengan mengamati benda-benda dan tindakan pengguna yang menggunakan di dunia nyata dan model objek antarmuka setelah rekan-rekan dunia nyata mereka
* Tahu pengguna, mengetahui tugas

The Process – Task Analysis of Display Content

* Type of content
	+ Character-based reports
	+ Graphical displays
	+ Specialized information
* Source of content
	+ Generated by components
	+ Acquired from data store
	+ Transmitted from systems external

The Process – Interface Design Activities

* Menetapkan tujuan dan maksud dari masing-masing tugas
* Memetakan setiap tujuan / niat untuk urutan tindakan tertentu
* Menentukan urutan aksi tugas dan subtugas (user skenario)
* Menunjukkan keadaan sistem pada saat scenario pengguna dilakukan
* Mendefinisikan mekanisme kontrol objek dan tindakan
* Menunjukkan bagaimana mekanisme control mempengaruhi keadaan dari sistem
* Menunjukkan bagaimana pengguna menafsirkan keadaan sistem dari informasi yang diberikan melalui antarmuka

Masalah Desain Antarmuka

* Waktu respon sistem
* Waktu antara titik di mana pengguna memulai beberapa tindakan kontrol dan waktu ketika system merespon
* Fasilitas Bantuan pengguna
* Terpadu, konteks bantuan sensitif terhadap add-on bantuan
* Kesalahan informasi penanganan
* Pesan harus tidak menghakimi, menjelaskan masalah tepatnya, dan menyarankan solusi valid
* Pelabelan perintah
* Berdasarkan pengguna kosakata, tata bahasa sederhana, dan memiliki aturan yang konsisten untuk singkatan

User Interface Evaluation Cycle

1. Desain awal
2. Membangun prototipe antarmuka pertama
3. Pengguna mengevaluasi antarmuka
4. Evaluasi dipelajari oleh desainer
5. Modifikasi desain dibuat
6. Membangun prototipe berikutnya
7. Jika antarmuka tidak lengkap kemudian pergi ke langkah 3

User Interface Design Evaluation Criteria

* Panjang dan kompleksitas dari spesifikasi antarmuka ditulis memberikan indikasi jumlah pembelajaran yang dibutuhkan oleh pengguna sistem
* Jumlah tugas pengguna dan jumlah rata-rata tindakan per tugas memberikan indikasi waktu interaksi dan efisiensi sistem secara keseluruhan
* Sejumlah tugas, tindakan, dan keadaan sistem dalam model desain memberikan indikasi beban memori yang dibutuhkan pengguna sistem
* Antarmuka gaya, fasilitas bantuan, dan penanganan kesalahan protokol memberikan indikasi umum
* kompleksitas sistem dan tingkat penerimaan oleh pengguna

Data Design

* Data Design Principles
	+ Prinsip-prinsip analisis sistematis diterapkan untuk fungsi dan perilaku juga harus diterapkan pada data.
	+ Semua struktur data dan operasi yang akan dilakukan pada masing-masing harus diidentifikasi.
	+ Kamus data harus ditetapkan dan digunakan untuk mendefinisikan data dan desain program.
	+ Proses desain tingkat rendah harus ditunda sampai akhir dalam proses desain.
	+ Representasi dari struktur data harus diketahui hanya kepada mereka modul yang harus menggunakan langsung dari data yang terdapat dalam dalam struktur data.
	+ Sebuah library struktur data yang berguna dan operasi harus dikembangkan.
	+ Sebuah desain perangkat lunak dan bahasa pelaksanaannya harus mendukung spesifikasi dan realisasi dari tipe data abstrak.

Componen Level Design

* Tujuan dari desain tingkat komponen untuk menerjemahkan model desain ke dalam perangkat lunak operasional.
* desain tingkat komponen terjadi setelah data, arsitektur, dan desain antarmuka ditetapkan.
* desain komponen-tingkat mewakili software dengan cara yang memungkinkan desainer untuk meninjau untuk kebenaran dan konsistensi, sebelum dibangun.
* Produk kerja yang dihasilkan adalah desain prosedural untuk setiap komponen software, diwakili menggunakan tabel, atau notasi grafis, berbasis teks

Procedurel Design Notation

* Flowcharts
	+ arrows for flow of control, diamonds for decisions, rectangles for processes
* Box diagrams
	+ also known as Nassi-Scheidnerman charts
* Decision table
	+ subsets of system conditions and actions are associated with each other to define the rules for processing inputs and events
* Program Design Language
	+ PDL - structured English or pseudocode used to describe processing details

Design Notation Assessment Criteria

* Modularity
	+ notation supports development of modular software
* Overall simplicity
	+ easy to learn, easy to use, easy to write
* Ease of editing
	+ easy to modify design representation when changes are necessary
* Machine readability
	+ notation can be input directly into a computer-based development system
* Maintainability
	+ maintenance of the configuration
* Structure enforcement
	+ enforces the use of structured programming constructs
* Automatic processing
	+ allows the designer to verify the correctness and quality of the design
* Data representation
	+ ability to represent local and global data directly
* Logic verification
	+ automatic logic verification improves testing adequacy
* Easily converted to program source code
	+ makes code generation quicker

Metoda dalam Perancangan User Interface

Ada beberapa metoda perancangan antara lain yaitu :

1 User Centered Design

 Salah satu pendekatan dalam merancang antarmuka pengguna adalah User Centered Design (UCD) yang melibatkan pengguna pada proses desain. Pengguna terlibat sejak tahap analisis awal kebutuhan pengguna untuk pengujian dan evaluasi [1].

Konsep UCD yaitu “ In user - centered design, the users are the center focus.”

Maksud dari pernyataan di atas adalah tujuan/ sifat-sifat, konteks, penguna, sifat konteks dan lingkungan produk semua didasarkan dari pengalaman penguna. Selanjutnya ditetapkan model pekerjaan pengguna yang akan didukung sistem.

Tahapan dalam UCD mencakup [1] :

 1. Memahami kebutuhan user.

2. Mendeskripsikan kebutuhan user.

3. Merancang prototype sebagai alternatif

4. Mengevaluasi perancangan.



Gambar 1. Proses User Centered Design [3]

 IBM UCD development community, sebuah komunitas penelitian terkemuka yang berfokus pada UCD, menjelaskan bahwa prinsip UCD terdiri dari enam [3]. (1) set the business goals; (2) understand the user; (3) Assessing competitiveness; (4) designing the total user experience; (5) Evaluating the design (6) Maintenance and support.

Prinsip pertama ‘Menetapkan tujuan bisnis' berarti mencakup semua aspek bisnis perangkat lunak. Itu harus mempertimbangkan pasar di mana sistem akan digunakan, pengguna yang akan mengoperasikan sistem atau yang disebut intended user, dan kompetisi pada sistem sejenis lainnya.

Prinsip yang kedua 'memahami pengguna'. Hal ini berfokus pada mendengarkan kebutuhan pengguna dan permintaan pasar. Untuk melakukan hal ini, pengembang software harus menyertakan pengguna dalam proses desain. Dengan kata lain pengguna harus didengar dan dicermati kebutuhannya karena akan menjadi bahan utama untuk merancang seluruh sistem. Prinsip ini hampir tidak dipertimbangkan dalam pendekatan lainnya, karena membutuhkan usaha, uang, waktu, dan pengetahuan yang paling besar.

Prinsip 'menilai daya saing' berfokus pada pengujian sistem oleh pengguna, dengan melakukan perbandingan pengguna melaksana-kan tugas pada sistem yang sedang dirancang dengan milik pesaing. Perbandingan atau pengujian ini dilakukan dalam rangka untuk menghasilkan sistem yang lebih baik.

Yang keempat adalah prinsip 'merancang total pengalaman pengguna' atau membuat pengguna senang dengan produk dan bangga memilikinya. Ini berfokus pada aspek internal dan eksternal seperti dokumentasi, peningkatan, dukungan, instalasi, iklan dan lainnya. Ini dimaksudkan untuk dipertimbangkan oleh tim desain dan penting untuk memberikan yang baik kesan produk ke pelanggan atau pengguna, sehingga kepuasannya.

Selanjutnya 'Mengevaluasi desain'. Pertama, desain lengkap diberikan kepada pengguna untuk mengetes. Ini disebut prototipe perangkat lunak. Prototipe dan rilis selalu diberikan pada iterasi dan umpan balik yang diterima dari pengguna setelah iterasi digunakan untuk memperbaiki desain dan pengembangan iterasi berikutnya atau rilis.

Yang terakhir ialah prinsip 'Pemeliharaan dan dukungan'. Ini berfokus pada aspek yang perlu dipertimbangkan setelah produk telah disampaikan. Kita semua tahu bahwa tidak ada hal seperti itu sebagai sempurna perangkat lunak; ini adalah mengapa perawatan dan dukungan harus ada bersama dengan dokumentasi dalam rangka memecahkan masalah tak terduga. Ini juga harus mencakup mendengarkan pengguna dan menonton mereka dan menggunakan umpan balik mereka untuk membuat sistem yang lebih baik.

2 Metode Prototyping

 Prototyping merupakan suatu metode pengembangan sistem yang menggunakan pendekatan untuk membuat sesuatu program dengan cepat dan bertahap sehingga segera dapat dievaluasi oleh pemakai [2]. Prototipe mewakili model produk yang akan dibangun atau mensimulasikan struktur, fungsionalitas dan operasi sistem.

Dimensi Prototyping [2] terdiri dari (1) Penyajian, yaitu bagaimana desain dilukiskan atau diwakili? Dapat berupa uraian tekstual atau dapat visual dan diagram. (2) Lingkup, yaitu apakah hanya interface atau mencakup komponen komputasi?; (3) Executability (Dapat dijalankan). Jika dikodekan, apakah akan ada periode saat prototype tidak dapat dijalankan? (4) Maturation yaitu tahapantahapan pengembangan produk. Ada dua tahap yakni secara (a) revolusioner: mengganti yang lama. Jika dirasa sistem yang lama sudah tidak dapat mengikuti perkembangan teknologi. (b) Evolusioner : terus melakukan perubahan pada perancangan yang sebelumnya.

Referensi :

Program Studi Teknik Informatika STEI ITB - IF-ITB/YW/Revisi: Oktober 2008