

BAB VI

DAYA GUNA

Menurut ISO 1998 :

Daya guna adalah tingkat produk dapat digunakan yang ditetapkan oleh user untuk mencapai tujuan secara efektif dan tingkat kepuasan dalam menggunakannya.

Daya guna merupakan salah satu faktor yang digunakan untuk mengukur sejauh mana penerimaan pengguna terhadap sistem.

Atribut dari daya guna tersebut adalah :

- Efektivitas
Ketelitian dan *kelengkapan* dimana user mencapai tujuan
- Learnabilitas
Mudah dipelajari oleh user baru
- Efisiensi
Sumber daya pembelajaran dalam hubungannya dengan ketelitian dan kelengkapan untuk user. (tidak membutuhkan alat-alat bantu)

- Memorabilitas
Mudah didalam menggunakan sistem dan perintah-perintahnya mudah diingat
- Kesalahan
Tingkat kesalahan yang kecil
- Kepuasan subjektif
Bebas dari ketidaknyamanan dan sikap positif dalam menggunakan produk

Untuk **mengukur daya guna** suatu produk dapat dilakukan hal berikut ini :

- Pembelajaran (*learnability*)
- Keefisienan (*efficiency*)
- Kemampuan mengingat (*memorability*)
- Kadar kesalahan (*errors*)
- Kepuasan (*satisfaction*)
- Presentasi (*presentation*)
- Susunan layar (*screen layout*)
- Istilah yang digunakan dan perintah yang disediakan oleh sistem
- Kemampuan sistem (*system capabilities*)

No	Kriteria untuk Metode Pengukuran Rekamaya Usabilitas
1	Waktu untuk menyelesaikan tugas
2	Berapa persen tugas bisa diselesaikan
3	Berapa persen tugas diselesaikan per unit waktu
4	Rasio keberhasilan dan kegagalan
5	Berapa waktu terjadi kesalahan
6	Berapa persen jumlah kesalahan
7	Berapa jumlah kompetitor dari produk yang sama
8	Jumlah perintah yang digunakan
9	Frekuensi help dan dokumentasi digunakan
10	Jumlah komentar dari user, yang baik maupun tidak
11	Jumlah perulangan perintah-perintah yang error
12	Jumlah run yang berhasil dan error
13	Berapa jumlah interface yang menyatkan user
14	Jumlah fitur yang baik dan yang jelek yang digunakan
15	Jumlah perintah yang tidak pernah digunakan
16	Jumlah kelakuan sistem yang tidak diperlukan
17	Jumlah user yang menyukai sistem yang dibuat
18	Jumlah waktu yang digunakan untuk menyelesaikan suatu permasalahan
19	Jumlah waktu yang tidak efektif dalam menyelesaikan masalah
20	Berapa banyak user yang kehilangan kontrol terhadap sistem
21	Berapa jumlah user yang puas dan tidak dalam menggunakan sistem

DAYA GUNA HEURISTIK

Daya guna heuristik merupakan prinsip atau panduan untuk merencanakan bentuk user interface, diantaranya adalah :

- Dialog yang sederhana dan alami (*simple and natural dialogue*)
- Berbicara dengan bahasa user (*speak the user language*)
- Mengurangi beban ingatan user (*minimize user memory load*)
- Konsisten (*consistency*)
- Sistem timbal balik (*system feedback*)

- Jalan keluar yang jelas (*clearly mark exit*)
- Jalan pintas (*shortcut*)
- Pesan-pesan kesalahan yang baik (*good error message*)
- Mencegah kesalahan (*prevent errors*)
- Bantuan dan dokumentasi (*help and documentation*)

Adapun penjelasan dari daya guna heuristik di atas yaitu :

■ Dialog yang Sederhana dan Alami

User interface harus sesingkat mungkin dan bersifat natural. Setiap dialog seharusnya menghindari perintah-perintah yang tidak perlu dan tidak ada hubungannya dengan interface, karena untuk setiap ciri atau elemen baru yang ditambahkan berarti satu masalah baru yang harus dipelajari oleh pengguna.

o Pendekatan yang harus digunakan adalah :

- Hanya menampilkan perintah yang diperlukan
- Bentuk elemen grafik dalam user interface modern
- Penggunaan warna yang baik dan tidak berlebihan
- Desain layar dalam bentuk yang lebih ringkas
- Dialog yang natural

■ Berbicara dengan Bahasa Pengguna

Dialog seharusnya menggunakan bahasa yang dipahami oleh user. Perintah-perintah yang berorientasi mesin mestinya tidak digunakan sama sekali. Selain itu frasa-frasa yang digunakan harus mudah dipahami kebanyakan user, bukan hanya segelintir saja. Penggunaan singkatan dan bahasa yang tidak jelas juga harus dihindari karena dapat disalah tafsirkan sehingga membuat user keliru.

- Penggunaan *metafora* merupakan salah satu pendekatan yang boleh digunakan. Objek yang tampil di layar, jenis perintah, jenis interaksi pengguna, cara sistem memberikan feedback dan sebagainya adalah berdasarkan frasa yang biasa digunakan, misalnya *desktop*, *icon*, *menu*, *cut*, *copy and paste*.

- Mengurangi Beban Ingatan Pengguna

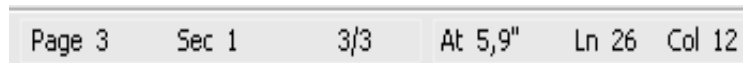
User seharusnya tidak dibebani untuk mengingat atau menghafal pada saat berinteraksi dengan sistem. Sebagai contoh penggunaan menu dapat mengurangi beban user dibandingkan penggunaan baris perintah. Aplikasi yang menggunakan menu lebih memuaskan dan fleksibel. Dalam kasus-kasus tertentu format perintah perlu disampaikan dengan jelas.
misalnya perintah DOS untuk menghapus dengan *del* dan membuat duplikasi dengan perintah *copy*

- Konsisten

Ciri-ciri konsisten adalah dapat menghindarkan user dari rasa was-was atau ragu-ragu di saat menggunakan suatu perintah atau fungsi. Disamping itu juga dapat mempercepat interaksi, misalnya perintah cetak dari windows dengan *File > Print*.

- Sistem Timbal Balik

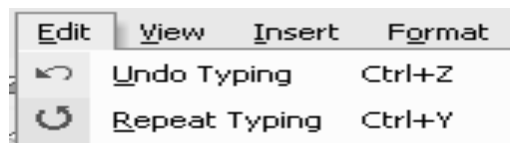
Sistem seharusnya memberitahu pengguna segala aktifitas yang sedang berlaku atau status dari sistem. Status sistem menunggu input dari pengguna, memproses input, menampilkan output, dan sebagainya. Proses ini juga akan memberitahu status suatu sistem jika terjadi suatu kerusakan, misalnya *status bar* pada Microsoft word.



Page 3 Sec 1 3/3 At 5,9" Ln 26 Col 12

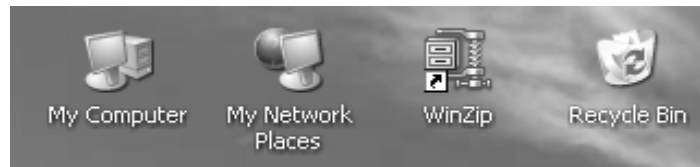
- Jalan Keluar yang Jelas

Sistem seharusnya dapat memberikan penjelasan tentang kondisi dan solusi untuk menghindari user terperangkap dalam tampilan-tampilan yang tidak diinginkan, aktivitas atau situasi dalam berinteraksi dengan sistem. Apabila user melakukan kesalahan dalam memilih perintah maka ia dapat keluar dari kesalahan tanpa ada masalah, misalnya perintah *Undo*.



- Jalan Pintas

Demi kemudahan dan kecepatan interaksi di dalam menggunakan suatu sistem maka sudah seharusnya bila tersedia *shortcut* yang berguna untuk membantu user agar dapat menggunakan berbagai fungsi dengan mudah.



- Pesan Kesalahan yang Baik

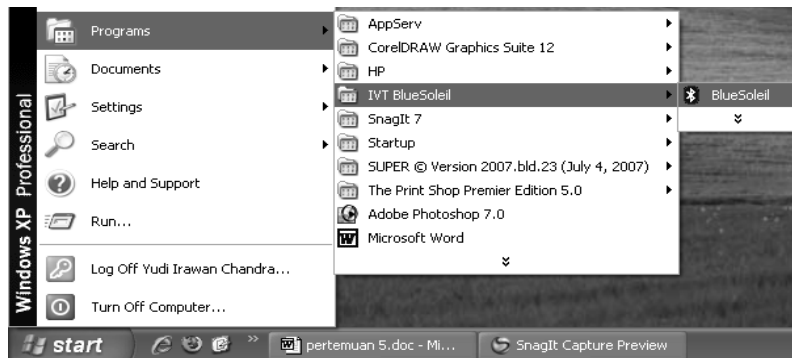
Menyediakan mekanisme pemberitahuan kesalahan dan menunjukkan situasi bahwa user berada dalam kondisi bermasalah serta membantu user untuk lebih memahami sistem.

- Terdapat empat peraturan yang harus diikuti dalam penggunaan pesankesalahan, yaitu :
 - Pesan kesalahan yang digunakan harus jelas dan mudah dipahami, disampaikan dalam bentuk teks, frasa atau konsep yang mudah dipahami
 - Pesan yang disampaikan bersifat khusus
 - Pesan kesalahan yang disampaikan sebaiknya menyediakan cadangan penyelesaian atas kesalahan
 - Penyampaian kesalahan dilakukan secara sopan.
- Contoh :



■ Mencegah Kesalahan

Rekayasa interface yang baik seharusnya mampu membuat user menghindari kesalahan, misalnya interaksi dengan menggunakan menu



■ **TEKNIK DAYA GUNA SIKLUS HIDUP**

Siklus hidup suatu daya guna memiliki elemen, antara lain :

- Kenali Pengguna (Know the user)
- Daya guna Benchmarking
- Desain Interaksi Berorientasi Tujuan (Goal-oriented interaction design)
- Iterative Design
- Studi Lanjutan (Follow up studies)

■ KENALI PENGGUNA

Mengenal siapa user bertujuan untuk :

- Mempelajari, mengenali dan memahami pengguna yang akan menggunakan sistem
- Merangkum keperluan user
- Kepuasan
- Kemahiran komputer

Masalah yang sering dihadapi adalah kesulitan untuk mendapatkan sasaran. Oleh karena itu perlu dilakukan hal-hal sebagai berikut :

- Riset kualitatif seperti pengamatan dan wawancara
- Mengklasifikasikan user berdasarkan perilaku dan variabel *demografis* (lingkungan)
- Identifikasi tujuan user dan *attitude*
- Menganalisa aliran kerja dan konteks kerja
- Menyusun tipikal skenario user

User dapat diklasifikasikan menjadi sebagai berikut :

- Pengalaman
- Tingkat pendidikan
- Umur
- Statistik pengguna sistem yang sudah ditraining

- Jika seorang pengguna ahli tidak memperbarui ilmunya secara terus-menerus maka dia akan turun ke posisi menengah.
- Hal ini juga dipertimbangkan dalam desain sistem, misalnya penggunaan sistem operasi Windows 98 sampai Windows Vista, tidak banyak mengalami perubahan sehingga memudahkan user dalam menerima sistem baru yang ditawarkan perusahaan.

■ **DAYA GUNA BENCHMARKING**

Produk-produk kompetitif atau produk yang telah ada perlu dipelajari untuk memperbaiki sistem yang sedang dibangun. Produk tersebut bisa dijadikan prototipe terbaik untuk membangun suatu produk.

Menganalisa produk kompetitif dilakukan dengan jalan :

- Menentukan kondisi dan memutuskan sejauh mana akan mengembangkan produk
- Meneliti perbedaan produk
- *Inteligency Borrowing*, ide dari sistem pesaing

Untuk menetapkan sasaran daya guna dan menentukan metrik daya guna serta *tingkat ukur daya guna* dengan cara :

- Sistem mempunyai kesalahan 4,5 % setiap satu jam ketika digunakan oleh user ahli. Untuk versi berikutnya mempunyai tingkat kesalahan 3% setiap satu jam
- Pada web kompetitif terdapat user setiap 8 menit dan 21 detik, target untuk web site yang baru adalah 6 menit.

■ DESAIN INTERAKSI BERORIENTASI TUJUAN

- Desain yang dibangun selalu memiliki tujuan untuk berinteraksi. Sewaktu mempelajari daya guna suatu sistem, parameter daya guna seharusnya bisa diukur. Sebelum merekayasa bentuk user interface yang baru, metrik daya guna seharusnya didiskusikan terlebih dahulu.
- Cara kerja komputer tidak sama dengan manusia. Bagian perangkat lunak harus jelas, yang dituliskan pada instruksi pemrograman dan bentuk interface harus bisa menyesuaikan dengan permintaan manusia.

■ ITERATIVE DESIGN

Bertujuan untuk desain, tes dan re-desain, kemudian membangun prototipe interface dengan cara :

- Menemukan masalah daya guna
- Menetapkan masalah untuk interface baru
 - Mengikuti dasar pemikiran desain, mengapa perubahan dibuat
 - Mengevaluasi interface

■ PROSES DESAIN INTERAKSI

Agar proses desain interaksi dapat mencapai tujuan maka harus dilakukan hal-hal berikut :

- Wawancara user
- Membuat persona
- Menjelaskan tujuan
- Membuat skenario yang jelas
- Solusi desain

■ MEMBUAT PERSONA

- Persona adalah suatu karakteristik yang diamati oleh orang lain atau disebut juga dengan *prototypical user*, seperti :
 - Imajinasi khusus, contohnya adalah user dengan tipe tertentu
 - Tidak real tetapi hipotesis
 - Digunakan sebagai *rule play* melalui desain interface

- Contoh persona adalah perusahaan mobil yang mendesain produknya.
- Kriteria pemrogram yang memiliki persona yang baik adalah :
Membuat program untuk rata-rata user, tidak hanya end user dengan tujuan agar user baru selalu mempelajarinya.

- Sifat user selalu elastis yang didefinisikan sebagai penampung ide-ide si pemrogram
- Pemrogram juga harus memperhatikan semua latar belakang user yang akan menggunakan program yang akan dibuat karena setiap individu memiliki persona yang berbeda.

■ Contoh lain adalah *inflight console* pada perusahaan pesawat terbang yang didesain sesuai untuk banyak persona sehingga diharapkan dapat memuaskan setiap penumpang. Mereka dapat mengisi waktu dengan menonton film, bermain game, belanja online, melihat berita dan mendengarkan musik.

- Persona membantu para perancang untuk :
- Menentukan apakah suatu produk diperlukan dan bagaimana cara kerjanya
 - Menyediakan bahasa suatu umum untuk mendiskusikan keputusan desain dan membantu proses desain
 - Mengurangi kebutuhan akan model diagramatik yang rumit
 - Efektifitas desain dapat diuji
 - Dapat melihat target yang diinginkan karena telah diuji coba terlebih dahulu

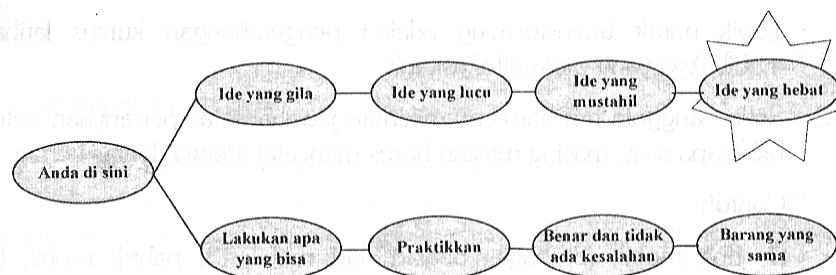
- Masalah yang dapat timbul selama pengembangan suatu produk :
 - User bersifat elastis, meski hari ini user telah puas dengan produk yang digunakan belum tentu esok hari juga merasa puas. Oleh sebab itu masih ada tahap selanjutnya yaitu pengembangan produk yang telah jadi
 - Percaya diri, karena jika ragu-ragu untuk meluncurkan produknya maka produk tersebut tidak akan pernah ada di pasaran

- Solusi desain yang baik untuk rekayasa interface :
 - Parallel Desain
Rekayasa bentuk yang dilakukan secara paralel merupakan pendekatan yang sering digunakan dalam rekayas sistem karena melibatkan banyak rekayasa untuk melihat dan sekaligus memberikan peluang untuk memilih rekayas bentuk awal dari berbagai alternatif pengembangan.

■ Brainstorming

Proses desain dengan brainstorming dapat dilakukan dengan :

- Brainstorming dengan suatu tim, misal ahli mesin, desainer grafik, penulis dan sebagainya
- Menggunakan kertas hasil desain yang banyak dan menempelkannya di dinding
- Menggambar, coret-coret dengan pulpen berwarna
- Bersifat masa bodoh
- Berkhayal untuk membangun suatu yang sulit dan berpikir jauh ke depan
- Semua ide yang berhasil dikumpulkan kemudian diorganisasikan dan dipilih salah satu yang terbaik dan diimplementasikan



■ **Aturan waktu melakukan brainstorming :**

- Semua ide dikumpulkan dari semua orang dalam tim dan tidak boleh dikritik oleh orang lain
- Semua ide yang masuk, baik yang masuk akal maupun tidak harus diterima. Semakin banyak ide yang masuk semakin baik
- Tidak boleh ada diskusi selama brainstorming berjalan karena diskusi akan dilakukan setelah brainstorming selesai
- Jangan mengkritik, menghakimi atau mentertawakan ide yang dikemukakan peserta
- Tulis semua ide pada papan tulis sehingga tim bisa melihat
- Atur waktu untuk aktivitas brainstorming misalnya 30 menit atau lebih

■ **Urutan dalam brainstorming :**

- Salah satu tim harus me-review topik yang digunakan dengan pertanyaan Why, How atau What
- Setiap anggota tim harus memikirkan jawaban atas pertanyaan untuk beberapa saat dan mencatatnya di kertas
- Setiap orang membacakan idenya atau semua ide ditulis di papan tulis
- Membuat pilihan akhir :
- Bila semua ide telah dicatat dan dikombinasikan dengan ide-ide yang mungkin, kategori awal harus tetap disepakati
- Jumlah ide yang ada
- Voting anggota digunakan untuk membuat sejumlah ide yang akan didiskusikan. Isi daftar tidak boleh lebih dari sepertiga jumlah ide