UNIVERSITAS PGRI WIRANEGARA PASURUAN

PANDUAN & PRINSIP DESAIN ANTARMUKA

DESAIN INTERAKSI VISUALISASI DAN GAME

PRINCIPLES OF UNIVERSAL DESIGN

Ron Mace, dari Pusat Desain Universal di North Carolina State University, mendefinisikan desain universal sebagai:

. . . desain produk dan lingkungan yang dapat digunakan oleh semua orang, semaksimal mungkin, tanpa perlu adaptasi atau desain khusus.

PRINSIP-PRINSIP DESAIN UNIVERSAL

PRINSIP	DESKRIPSI
Equitable use	Berguna dan dapat dipasarkan untuk orang-orang dengan beragam kemampuan.
Flexibility in use	Mengakomodasi berbagai preferensi dan kemampuan individu.
Simple and intuitive use	Penggunaan desain mudah dimengerti, terlepas dari pengalaman pengguna, pengetahuan, keterampilan berbahasa, atau tingkat konsentrasi saat ini.
Perceptible information	Mengkomunikasikan informasi yang diperlukan secara efektif kepada pengguna, terlepas dari kondisi ambien atau kemampuan sensorik pengguna.
Tolerance for error	Meminimalkan bahaya dan konsekuensi merugikan dari tindakan yang tidak disengaja atau tidak disengaja.
Low physical effort	Dapat digunakan secara efisien dan nyaman dengan tingkat kelelahan yang minimal.
Size and space for approach and use	Ukuran dan ruang yang sesuai disediakan untuk pendekatan, jangkauan, manipulasi, dan penggunaan terlepas dari ukuran tubuh, postur, atau mobilitas pengguna.

"

MENJEMBATANI KESENJANGAN ANTARA MENGUMPULKAN PERSYARATAN DAN MENCIPTAKAN DESAIN FISIK ANTARMUKA PENGGUNA (UI)

"

RE-ENGINEERING PEKERJAAN

- Untuk memaksimalkan manfaat mengembangkan UI baru, perlu dipertimbangkan untuk merekayasa ulang pekerjaan.
- Dengan UI baru, mungkin user perlu untuk bekerja secara berbeda agar mereka dapat bekerja secara efektif.
- Membutuhkan penanganan yang sensitif, dan merupakan alasan untuk melibatkan pengguna di seluruh proses pengembangan: jika saran yang Anda buat cenderung tidak umum atau benar-benar tidak masuk akal, pengguna akan segera memberi tahu Anda.

TUJUAN REKAYASA ULANG PEKERJAAN

Mayhew (1999, p. 172)

- Meningkatkan kekuatan dan efisiensi yang dimungkinkan oleh otomasi.
- Meningkatkan efektifitas dalam mendukung tujuan bisnis.
- Meminimalkan penugasan dengan memiliki line produk baru sebanyak mungkin ke dalam pengetahuan tugas pengguna yang ada, dan memaksimalkan efisiensi dan efektivitas dengan mengakomodasi kendala dan kemampuan kognitif manusia dalam konteks tugas mereka yang sebenarnya.

USER COMPATIBILITY

- Antarmuka merupakan pintu gerbang masuk ke sistem dengan diwujudkan ke dalam sebuah aplikasi software.
- Software seolah-olah mengenal usernya, mengenal karakteristik usernya, dari sifat sampai kebiasaan manusia secara umum.
- Desainer harus mencari dan mengumpulkan berbagai karakteristik serta sifat dari user karena antarmuka harus disesuaikan dengan user yang jumlahnya banyak dan mempunyai karakter yang berbeda.
- Hal tersebut harus terpikirkan oleh desainer dan tidak dianjurkan merancang antarmuka dengan didasarkan pada dirinya sendiri
- Survey adalah hal yang paling tepat

PRODUCT COMPATIBILITY

- Antarmuka harus sesuai dengan sistem aslinya.
- Seringkali sebuah aplikasi menghasilkan hasil yang berbeda dengan sistem manual atau sistem yang ada.
- Hal tersebut sangat tidak diharapkan dari perusahaan karena dengan adanya aplikasi software diharapkan dapat menjaga produk yang dihasilkan dan dihasilkan produk yang jauh lebih baik.
- Contoh : aplikasi sistem melalui antarmuka diharapkan menghasilkan report/laporan serta informasi yang detail dan akurat dibandingkan dengan sistem manual.

TASK COMPATIBILITY

- Sebuah aplikasi harus mampu membantu para user dalam menyelesaikan tugasnya. Semua pekerjaan serta tugas-tugas user harus diadopsi di dalam aplikasi tersebut melalui antarmuka.
- Sebisa mungkin user tidak dihadapkan dengan kondisi memilih dan berpikir, tapi user dihadapkan dengan pilihan yang mudah dan proses berpikir dari tugas-tugas user dipindahkan dalam aplikasi melalui antarmuka.
- Contoh : User hanya klik setup, tekan tombol next, next, next, finish, ok untuk menginstal suatu software.

WORK FLOW COMPATIBILITY

- Sebuah aplikasi sistem sudah pasti mengapdopsi sistem manualnya dan didalamnya tentunya terdapat urutan kerja dalam menyelesaikan pekerjaan
- Dalam sebuah aplikasi, software engineer harus memikirkan berbagai urutan pekerjaan yang ada pada sebuah sistem.
- Jangan sampai user mengalami kesulitan dalam menyelesaikan pekerjaannya karena user mengalami kebingungan ketika urutan pekerjaan yang ada pada sistem manual tidak ditemukan pada software yang dihadapinya.
- Selain itu user jangan dibingungkan dengan pilihan-pilihan menu yang terlalu banyak dan semestinya menu-menu merupakan urutan dari runutan pekerjaan.
- Sehingga dengan workflow compatibility dapat membantu seorang user dalam mempercepat pekerjaannya.

FAMILIARITY

- Sifat manusia mudah mengingat dengan hal-hal yang sudah sering dilihatnya atau didapatkannya. Secara singkat disebut dengan familiar.
- Antarmuka sebisa mungkin didesain sesuai dengan antarmuka pada umumnya, dari segi tata letak, model, dsb.
- Hal ini dapat membantu user cepat berinteraksi dengan sisem melalui antarmuka yang familiar bagi user.

DIRECT MANIPULATION

- User berharap aplikasi yang dihadapinya mempunyai media atau tools yang dapat digunakan untuk melakukan perubahan pada antarmuka tersebut.
- User menginginkan aplikasi yang dihadapannya bias disesuaikan dengan kebutuhan, sifat dan karakteristik user tersebut.
- Selain itu, sifat dari user yang suka merubah atau mempunyai rasa bosan.
- Contoh : tampilan warna sesuai keinginan (misal pink) pada window bisa dirubah melalui desktop properties, tampilan skin winamp bisa dirubah, dll.

CONTROL

- Prinsip control ini berkenaan dengan sifat user yang mempunyai tingkat konsentrasi yang berubah-ubah. Hal itu akan sangat mengganggu proses berjalannya sistem.
- Kejadian salah ketik atau salah entry merupakan hal yang biasa bagi seorang user. Akan tetapi hal itu akan dapat mengganggu sistem dan akan berakibat sangat fatal karena salah memasukkan data 1 digit/1 karakter saja informasi yang dihasilkan sangat dimungkinkan salah.
- Oleh karena itu software engineer haruslan merancang suatu kondisi yang mampu mengatasi dan menanggulangi hal-hal seperti itu.
- Contoh : “illegal command”, “can’t recognize input” sebagai portal jika terjadi kesalahan.

WYSIWYG = WHAT YOU SEE IS WHAT YOU GET

- Apa yang didapat adalah apa yang dilihatnya.
- Contoh : apa yang tercetak di printer merupakan informasi yang terkumpul dari data-data yang terlihat di layar monitor pada saat mencari data.
- Hal ini juga perlu menjadi perhatian software engineer pada saat membangun antarmuka.
- Informasi yang dicari/diinginkan harus sesuai dengan usaha dari user pada saat mencari data dan juga harus sesuai dengan data yang ada pada aplikasi sistem (software).
- Jika sistem mempunyai informasi yang lebih dari yang diinginkan user, hendaknya dibuat pilihan (optional) sesuai dengan keinginan user. Bisa jadi yang berlebihan itu justru tidak diinginkan user.
- Yang mendasar disini adalah harus sesuai dengan kemauan dan pilihan dari user.

FLEXIBILITY

Fleksibel merupakan bentuk dari dari solusi pada saat menyelesaikan masalah.

Software engineer dapat membuat berbagai solusi penyelesaian untuk satu masalah.

Sebagai contoh adanya menu, hotkey, atau model dialog yang lainnya.

RESPONSIVENESS

- Setelah memberikan inputan atau memasukkan data ke aplikasi system melalui antarmuka, sebaiknya sistem langsung memberi tanggapan/respon dari hasil data yang diinputkan
- Selain teknologi komputer semakin maju sesuai dengan tuntutan kebutuhan manusia, software yang dibangun pun harus mempunyai reaksi tanggap yang cepat. Hal ini didasari pada sifat manusia yang semakin dinamis/tidak mau menunggu.

INVISIBLE TECHNOLOGY

- Secara umum, user mempunyai keingintahuan sebuah kecanggihan dari aplikasi yang digunakannya. Untuk itu aplikasi yang dibuat hendaknya mempunyai kelebihan yang tersembunyi. Bisa saja kelebihan itu berhubungan dengan system yang melingkupinya atau bisa saja kecanggihan atau kelebihan itu tidak ada hubungannya.
- Contoh : sebuah aplikasi mempunyai voice recognize sebagai media inputan, pengolah kata yang dilengkapi dengan language translator.

ROBUSTNESS

- Pembangunan antarmuka yang baik dapat berupa frase-frase menu atau error handling yang sopan. Kata yang digunakan harus dalam kondisi bersahabat sehingga nuansa user friendly akan dapat dirasakan oleh user selama menggunakan sistem

- Contoh yang kurang baik : YOU FALSE !!!, BAD FILES !!!, FLOPPY ERROR, dsb. Akan lebih baik jika BAD COMMAND OR FILES NAMES, DISK DRIVE NOT READY,dll.

PROTECTION

- Suasana nyaman perlu diciptakan oleh software engineer di antarmuka yang dibangunnya.
- Nyaman disini adalah suasana dimana user akan betah dan tidak menemui suasana kacau ketika user salah memasukkan data atau salah eksekusi.
- Seorang user akan tetap merasa nyaman ketika dia elakukan kesalahan, misal ketika user menghapus file tanpa disengaja tidak mengakibatkan kesalahan yang fatal karena misal ada *recovery tools* seperti *undo*, recycle bin, dll atau “are you sure….”
- Proteksi disini lebih menjaga kenyamanan user ketika menggunakan aplikasi sistem, khususnya data-data berupa file.

EASE OF LEARNING AND EASE OF USE

- Kemudahan dalam mengoperasikan softwarehanya dengan memandangi atau belajar beberapa saat saja.
- Kemudahan dalam memahami icon, menu-menu, alur data software, dsb.
- Sesudah mempelajari, user dengan mudah dan cepat menggunakan software tersebut. Jika sudah memahami tentunya akan membantu proses menjalankan sistem dengan cepat dan baik.

PERANTI BANTU PENGEMBANGAN SISTEM

- Komputer tidak dapat berinteraksi dengan manusia bila tidak ada jembatan yang menghubungkan keduanya.
- Komputer dan manusia bisa berhubungan dengan menggunakan software atau tool

Thank you!