PRINSIP, PARADIGMA INTERAKSI & DESAIN PROSES

Sistem Interaktif memungkinkan user mencapai suatu tujuan tertentu dalam suatu domain aplikasi. Sistem interaktif harus dapat didayagunakan (usability) untuk meningkatkan keberhasilan suatu sistem aplikasi.

Pradigma yaitu sistem interaksi yang berhasil pada umumnya diyakini akan meningkatkan daya guna dari system tersebut.

Model interaksi membantu kita untuk mengerti apa yang terjadi di antara pengguna dan sistem, menerjemahkan tujuan, antara apa yang didinginkan user dan dan apa yang harus dikerjakan oleh sistem.

Dengan memahami bagaimana, manusia berprilaku dan sistem kerja komputer maka akan terjadi interaksi yang baik antara manusia dan komputer sehingga akan didapatkan hasil yang paling baik.

A. Paradigma Interaksi

Paradigma yaitu sistem interaksi yang berhasil pada umumnya diyakini akan meningkatkan daya guna dari sistem tersebut.

Prinsip yaitu: Interaksi efektif dari bebagai aspek pengetahuan seperti psikologi, komputasi dan sosiologi yang mengarahkan pada peningkatan desain, dan evolusi produk yang pada akhirnya akan meningkatkan daya guna sistem.

Sistem Interaktif memungkinkan user mencapai suatu tujuan tertentu dalam suatu domain aplikasi. Sistem interaktif harus dapat didayagunakan (usability) untuk meningkatkan keberhasilan suatu sistem aplikasi.

Dua pendekatan agar suatu sistem interatif dibangun dan cara mengukur atau mendemontra-sikan dayaguna (usability) suatu sistem interaktif

1. Paradigma adalah sistem interaktif yang berhasil /sukses pada umumnya diyakini akan mening-katkan dayaguna (usability) dari sistem tersebut.

2. Prinsip adalah interaksi efektif dari berbagai aspek pengetahuan psikologi, komputasi dan sosiologi mengarahkan peningkatan desain dan evolusi suatu produk, yang pada akhirnya akan meningkatkan daya-guna sistem tersebut.

B. Jenis-jenis Paradigma

1. Time Sharing

* Satu komputer yang mampu mendukung (dapat digunakan oleh)  
  multiple user
* Meningkatkan keluaran (throughput) dari sistem

2. Video Display Units (VDU)

* Dapat memvisualisasikan dan memanipulasi informasi yang sama  
  dalam representasi yang berbeda
* Mampu memvisualisasikan abstraksi data

3. Programming Toolkits (Alat Bantu Pemrograman)

* Alat Bantu Pemrograman memungkinkan programmer  
  meningkatkan produktivitasnya

4. Komputer Pribadi (Personal Computing)

* Mesin berukuran kecil yang powerful, yang dirancang untuk user  
  tunggal.

5. Sistem Window dan interface WIMP (Windows, Icons, Menus and

Pointers)

* Sistem window memungkinkan user untuk berdialog / berinteraksi  
  dengan komputer dalam beberapa aktivitas/topik yang berbeda

6. Metapora (Metaphor)

* Metapora telah cukup sukses digunakan untuk mengajari konsep  
  baru, dimana konsep tersebut telah dipahami sebelumnya.
* Contoh metapora (dalam domain PC):
* Spreadsheeet adalah metapora dari Accounting dan Financial Modelling. Keyboard adalah metapora dari Mesin Tik

7. Manipulasi Langsung (Direct Manipulation)

* Manipulasi Langsung memungkinkan user untuk mengubah  
  keadaan internal sistem dengan cepat.
* Contoh Direct Manipulation adalah konsep WYSIWYG (what you see  
  is what you get)

8. Bahasa vs. Aksi (Language versus Action)

* Bahasa digunakan oleh user untuk berkomu-nikasi dengan  
  interface
* Aksi dilakukan interface untuk melaksanakan perintah user

9. Hypertext

* Penyimpanan informasi dalam format linear tidak banyak  
  mendukung pengaksesan infor-masi secara random dan browsing  
  asosiatif.
* Hypertext merupakan metode penyimpanan informasi dalam format  
  non-linear yang memungkinkan akses atau browsing secara nonlinear  
  atau random.

10. Multi-Modality

* Sistem multi-modal interaktif adalah sistem yang tergantung pada  
  penggunaan beberapa (multiple) saluan (channel) komunikasi pada  
  manusia
* Contoh channel komunikasi pada manusia : visual (mata), haptic  
  atau peraba (kulit), audio (telinga).

11. Computer-Supported Cooperative Work (CSCW)

* Perkembangan jaringan komputer memung-kinkan komunikasi  
  antara beberapa mesin (personal komputer) yang terpisah dalam  
  satu kesatuan grup.
* Sistem CSCW dirancang untuk memungkin-kan interaksi antar  
  manusia melalui komputer dan direpresentasikan dalam satu  
  produk.
* Contoh CSCW: e-mail (electronic)

C. Prinsip-prinsip Pendukung interaksi

1. Learnability : Kemudahan yang memungkinkan user baru berinteraksi secara efektif dan dapat mencapai performance yang maksimal.

Prinsip yang Mempengaruhi Learnability:

|  |  |  |
| --- | --- | --- |
| **PRINSIP** | **DEFINISI** | **PRINSIP YANG TERKAIT** |
| Predictablity | Mendukung user untuk menentukan efek dari *future action* berdasarkan catatan atau sejarah interaksi sebelumnya | Operation Visibility |
| Synthesizability | Mendukung user untuk memperkirakan efek dan operasi sebelumnya pada keadaan saat ini | Immediate |
| Familiarity | Pengetahuan dan pengalaman user dalam domain berbasis komputer atau dunia nyata lainnya dapat diterapkan ketika berinteraksi dengan sistem yang baru | Guessability Affordance |
| Generalizability | Mendukung user untuk menambah pengetahuan dari interaksi spesifik di dalam dan di luar aplikasi ke situasi yang lebih mirip |  |
| Consistency | Kemiripan dalam perilaku input atau output yang muncul dari situasi atau tugas obyektif yang sama |  |

2. Flexibility : Menyediakan banyak cara bagi user dan sistem untuk bertukar informasi.

Prinsip yang Mempengaruhi Flexibility:rinsip yang mempengaruhi *flexibilty* :

|  |  |  |
| --- | --- | --- |
| **PRINSIP** | **DEFINISI** | **PRINSIP YANG TERKAIT** |
| Dialogue Initiative | Memungkinkan user terbebas dari kendala-kendala buatan (*artificial*) pada dialog input yang dipaksakan oleh sistem | Sistem atau *user primitiveness* |
| Multi Treading | Kemampuan sistem untuk mendukung interaksi user yang berhubungan dengan lebih dari satu *task* pada suatu saat | Concurrent versus interleaving multimodality |
| Task Migratability | Kemampuan untuk melewatkan/memberikan kontrol dari eksekusi *task*yang diberikan sehingga menjadi *task internal* *user* atau sistem atau berbagi antara keduanya |  |
| Substitutivity | Memungkinkan nilai-nilai (*values*) ekuivalen antara input dan output yang masing-masing secara bebas dapat disubstitusi | Representasi perkalian, kesamaan kesempatan (*opportunity*) |
| Customizability | Kemampuan *user interface* untuk dimodifikasi oleh user atau sistem | Adaptivitas, Adaptabilitas |

yang mempengaruhi *robustness* :

3. Robustness : Tingkat dukungan yang diberikan agar user dapat menentukan keberhasilan ataupun tujuan (goal) yang diinginkan.

Prinsip yang Mempengaruhi Robustness:

|  |  |  |
| --- | --- | --- |
| **PRINSIP** | **DEFINISI** | **PRINSIP YANG TERKAIT** |
| Observability | Kemampuan untuk mengevaluasi keadaan sistem internal dari representasi yang dapat dimengerti atau dirasakan | Browsability, static atau dinamic defaults, reachability, persistence, operation visibility |
| Recoverability | Kemampuan user untuk melakukan koreksi bila sebuah kesalahan telah dikenali | Reachability, forward atau backward recovery, commensurate effort |
| Responsiveness | Bagaimana user mengetahui atau menyadari laju komunikasi dengan sistem | Stabilitas |
| Task Conformance | Tingkatan dimana sistem pelayanan mendukung semua task yang user ingin lakukan dan dengan cara yang user ketahui | Task completeness, task adequacy |

D. Siklus Perkembangan Softwere

Siklus hidup Software (Software Development lifecycle) adalah sebuah usaha untuk mengidentifikasi aktifitas yang terjasi selama pengembangan sebuah perangkat lunak. Aktifitas ini kemudian diurutkan sesuai dengan waktu pelaksanaannyaproyek pengembangan manapun dan diaplikasikan tehnik yang tepat pada setiap aktifitasnya.

1. Model Life Cycle Software

Dalam pengembangan software ada beberapa tahapan untuk mencapai kualitas pembuatan/ siklus hidup software. Dapat kami jabarkan siklus hidup software atau tahap penegmbangan software sebagai berikut :

Tahap Pengembangan Software ( Siklus Hidup Software )

a. Requirements Analysis ( Analisa Kebutuhan )

Tahap ini menganalisa masalah dan kebutuhan yang harus diselesaikan dengan sistem komputer yang akan dibuat. Tahap ini berakhir dengan pembuatan laporan kelayakan yang mengidentifikasi kebutuhan sistem yang baru dan merekomendasikan apakah kebutuhan atau masalah tersebut dapat diselesaikan dengsn sistem komputer yang ada.

b. System and Software Design ( Prencanaan Sistem dan Software )

Tahap ini melakukan rancangan design sistem. Tahap ini memberikan rincian kinerja program dan interaksi antara user dengan program tersebut.

c. Implementation ( Implementasi )

Tahap ini adalah spesifikasi design yang telah dibuat untuk diterjemahkan de dalam program / instruksi yang ditulis dalam bahasa pemrograman.

d. System Testing ( Pengujian Sistem )

Tahap ini semua program digabungkan dan diuji sebagai satu sistem yang lengkap untuk menjamin sumua berkerja dan memenuhi kebutuhan penanganan masalah yang dihadapi.

e. Operation and Maintenance ( Pengoperasian dan Pemeliharaan )

Tahap ini merupakan pengaplikasian program yang telah dibuat untuk digunakan secara utuh dan masalah baru yang muncul sebagai bahan masukan untuk memperbaiki sistem program yang baru.

2. Waterfall Model

Nama model ini sebenarnya adalah “Linear Sequential Model”. Model ini sering disebut dengan “classic life cycle” atau model waterfall. Model ini adalah model yang muncul pertama kali yaitu sekitar tahun 1970 sehingga sering dianggap kuno, tetapi merupakan model yang paling banyak dipakai didalam Software Engineering (SE).

Model ini melakukan pendekatan secara sistematis dan urut mulai dari level kebutuhan sistem lalu menuju ketahap analisis, desain, coding, testing dan maintenance.Model ini merupakan model yang paling banyak dipakai oleh para pengembang software.Adalima tahap dalam model waterfall, yaitu: Requirement Analysis, System Design, Implementation, Integration & Testing, Operations & Maintenance.

Sesuai dengan namanya waterfall (air terjun) maka tahapan dalam model ini disusun bertingkat, setiap tahap dalam model ini dilakukan berurutan, satu sebelum yang lainnya (lihat tanda anak panah). Selain itu dari satu tahap kita dapat kembali ketahap sebelumnya. Model ini biasanya digunakan untuk membuat sebuah software dalam skala besar dan yang akan dipakai dalam waktu yang lama.

Kelebihan Waterfall Model

Ketika semua kebutuhan sistem dapat didefinisikan secara utuh, eksplisit, dan benar diawal proyek, maka software engineering dapat berjalan dengan baik dan tanpa masalah.

Kekurangan Waterfall Model

1. Ketika problem muncul, maka proses berhenti, karena tidak dapat menuju ketahapan selanjutnya.
2. Karena pendekatannya secara sequential, maka setiap tahap harus menunggu hasil dari tahap sebelumnya.
3. Pada setiap tahap proses tentunya dipekerjakan sesuai spesialisasinya masing-masing.

3. RAD ( Rapid Application Development ) Model

Rapid Application Development (RAD) atau Rapid Prototyping adalah model proses pembangunan perangkat lunak yang tergolong dalam teknik incremental (bertingkat). RAD menekankan pada siklus pembangunan pendek, singkat, dan cepat. Waktu yang singkat adalah batasan yang penting untuk model ini. Rapid application development menggunakan metode interatif (berulang) dalam mengembangkan sistem dimana working model (model bekerja) sistem dikonstruksikan di awal tahap pengembangan dengan tujuan menetapkan kebutuhan (requirement) user dan selanjutnya disingkirkan. Working model digunakan kadang-kadang saja sebagai basis desain dan implementasi sistem final.

Kelebihan RAD Model :

RAD memang lebih cepat dari  Waterfall. Jika kebutuhan dan batasan proyek sudah diketahui dengan baik. Juga jika proyek memungkinkan untuk dimodularisasi.

Kekurangan RAD Model :

1. Tidak semua proyek bisa dipecah (dimodularisasi), sehingga belum tentu RAD dipakai pada semua proyek. Karena proyek dipecah menjadi beberapa bagian, maka dibutuhkan banyak orang untuk membentuk suatu tim yang mengerjakan tiap bagian tersebut.
2. Membutuhkan komitmen antara pengemang dengan pelanggan.
3. Model RAD memerlukan sumber daya yang cukup besar, terutama untuk proyek dengan skala besar.
4. Resiko teknis yang tinggi kurang cocok untuk model ini.
5. Sistem yang tidak bisa dimodularisasi tidak cocok untuk model ini.
6. Karena dibuat dengan reuse komponen-komponen yang sudah ada, fasilitas-fasilitas pada tiap komponen belum tentu digunakan seluruhnya oleh program yang me-reuse-nya sehingga kualitas program.

E. Aturan Desain

1. Standard

a. Standar suatu desain selalu ditentukan oleh organisasi nasional atau internasional, ISO atau BSI, untuk memastikan kepastian pemenuhan syarat-syarat tertentu oleh komunitas besar para desainer.

b. Standar memerlukan teori mendasar dan secara pelan mengubah teknologi

c. Standar memerlukan teori mendasar pada faktor fisiologi atau ergonomic

d. Standar perangkat lunak berdasarkan pada faktor psikologi dan teori kognotif

e. Standar perangkat keras lebih umum digunakan dibanding standar perangkat lunak

f. Perangkat keras memiliki harga yang mahal dan sulit untuk diubah. Perubahan sering terjadi

g. Perangkat keras memiliki harga yang murah dan untuk untuk diubah. Perubahan tidak sering terjadi

h. ISO 9241 mendefinisikan tingkat kegunaan sebagai efektivitas, efisiensi dan kepuasan dengan mana pengguna menyelesaikan suatu tugas

i. Daya guna: efektivitas, efisiensi dan kepuasan bagi user untuk mencapai tujuan.

2. Guidelines (Garis Pedoman)

a. Lebih bersifat saran dan umum

b. Banyak biku teks dan laporan yang berisi garis pedoman

c. Abstrak dari garis pedoman (prinsip) dapat digunakan selama aktivitas awal siklus hidup

d. Detail garis pedoman (petunjuk gaya – style guides) dapat digunakan selama aktifitas awal siklus hidup

e. Pemahaman pembenaran untuk garis pedoman ini akan membantu penyelesaian konflik yang teradi.

F. Prototyping Model

Prototype adalah bagian dari produk yang mengekspresikan logika maupun fisik antarmuka ekternal yang ditampilkan.

Komponen potensial menggunakan prototype dan menyediakan masukan tim pengembangan sebelum  pengembangan skala besar dimulai. Melihat dan mempercayai menjadi hal yang diharapkan untuk dicapai dalam prototype. Dengan menggunakan pendekatan ini, konsumen dan tim pengembangan dapat mengklarifikasi kebutuhan pengembangan software dan intrepetasi mereka.

Tahap-tahap rekayasa software dalam Prototype Model, yaitu:

1. Pengumpulan kebutuhan

2. Perancangan Cepat

3. Bangun Prototype

4. Evaluasi prototype

5. Perbaikan Prototype

G. Desain Rasionalitas

Kelebihan Prototype Model adalah :

1. Penentuan kebutuhan lebih mudah diwujudkan.
2. Mempersingkat waktu pengembangan software.

Kekurangan Prototype Model adalah :

1. Proses analisis dan perancangan terlalu singkat.
2. Mengesampingkan alternatif pemecahan masalah.
3. Bisanya kurang fleksibel dalam menghadapi perubahan.
4. Prototype yang dihasilkan tidak selamanya mudah dirubah.
5. Prototype terlalu cepat selesai.

1. Desain Rasiionalitas

Rasionalitas design (design rasionality) adalah informasi yang menjelaskan alasan mengapa suatu keputusan dalam suatu tahap perancangan / desain sistem komputer dibuat atau diambil, termasuk deskripsi struktural atau arsitektural dan deskripsi fungsi atau perilakunya.

Beberapa keuntungan rasionalitas desain:

Dalam bentuk yang eksplisit rasionaitas desain menyediakan makanisme komunikasi diantara anggota tim desain sehingga pada tahapan desain atau pemeliharaan( maintenance), anggota tim memahami keputusan kritis/penting mana yang telah dibuat,alternatif  apa saja yang telah diteliti, dan alasan apa yang menyebabkan suatu alternatif  dipilih diantara alternatif  lain.

Akumulasi pengetahuan dalam bentuk rasionalitas desain untuk suatu set produk dapat digunakan kembali untuk mentransfer hal yang berhasil dalam suatu situasi ke siatuasi yang lainnya yang mirip. Usaha yang diperlukan untuk menghasilkan sebuah rasionalitas desain  memaksa desainer untuk bersikap hati-hati dalam mengambil suatu keputusan desain.