BAB 2
LANDASAN TEORI

2.1 Teori Umum

Teori – teori dasar atau umum yang digunakan untuk mendukung dan membantu pembuatan Skripsi ini.

2.1.1 Pengertian Sistem

Istilah sistem merupakan istilah dari bahasa Yunani "*system*" yang artinya adalah himpunan bagian atau unsur yang saling berhubungan secara teratur untuk mencapai tujuan. Berikut ini pengertian sistem dari beberapa ahli :

Sistem menurut pendapat Satzinger, Jackson, & Burd (2010, p6) adalah kumpulan komponen-komponen yang saling berkaitan yang berfungsi bersama untuk mencapai beberapa hasil.

Sedangkan sistem menurut pendapat O'Brien, & Marakas (2009, p24) adalah sekelompok komponen yang saling berkaitan dan bekerja sama kearah tujuan bersama dengan menerima masukan-masukan dan menghasilkan keluaran dalam proses pengelolaan transformasi atau perubahan.

Menurut Romney dan Steinbart (2006, p4), sistem adalah kumpulan dari dua atau lebih komponen yang berinteraksi untuk mencapai tujuan. Sistem terdiri dari subsistem yang lebih kecil, masing-masing melaksanakan fungsi penting dan mendukung sistem yang lebih besar.

Berdasarkan pendapat-pendapat di atas, maka dapat disimpulkan bahwa definisi dari sistem adalah suatu koleksi dari komponen – komponen yang memiliki suatu kesatuan dan berhubungan dengan satu dengan yang lain dan memiliki fungsi yang sama.

2.1.2 Pengertian Informasi

Menurut Rainer & Cegielski (2010, p10), informasi mengacu pada data yang telah diorganisasikan sehingga mempunyai arti dan nilai bagi penerimanya.

Menurut O'Brien & Marakas (2009, p38), informasi merupakan data yang telah diubah menjadi konteks yang berarti dan berguna untuk penerima akhir tertentu.

Berdasarkan definisi diatas dapat disimpulkan bahwa informasi adalah data yang telah diolah untuk menghasilkan suatu hasil (*output*) dimana hasil dari proses tersebut bermanfaat bagi penerimanya.

2.1.3 Pengertian Sistem Informasi

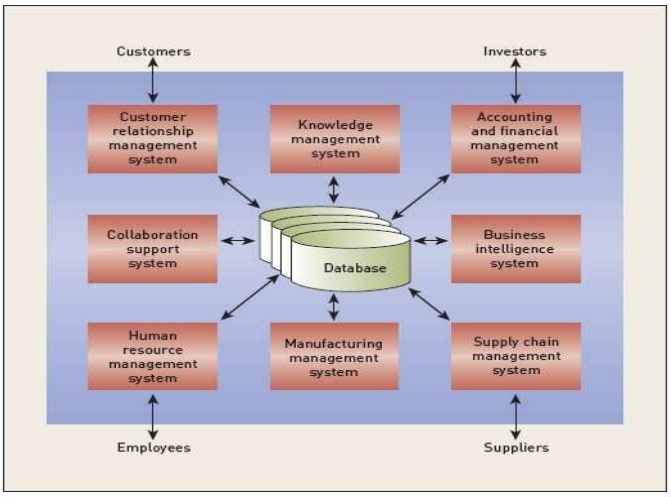
Menurut O'Brien & Marakas (2009, p4), Sistem Informasi adalah suatu kombinasi yang teratur baik dari orang – orang, hardware, software, jaringan komunikasi, dan sumber daya data dimana mencakup kegiatan mengumpulkan, mengubah, dan menyebarkan informasi dalam sebuah organisasi.

Menurut Satzinger, Jackson, & Burd (2010, p6), Sistem Informasi adalah kumpulan komponen yang saling berhubungan dimana mengumpulkan, memproses, menyimpan, dan menyajikan *output* dari informasi yang dibutuhkan untuk menyelesaikan tugas bisnis.

Setiap sistem mempunyai sebuah batas antara lingkungan sistem dan sistem itu sendiri. Setiap *input* maupun *output* harus melewati *system boundary*. *System boundary* adalah batas antara sebuah sistem dan lingkungannya dimana harus dilewati oleh *input* dan *output*. Dalam sebuah sistem informasi, manusia juga merupakan sebuah kunci komponen dan manusia tersebut melakukan sebuah kerja untuk diselesaikan oleh sebuah sitem. Jadi selain terdapat *system boundary*, juga terdapat batas lainnya yang dinamakan *automation boundary* yang merupakan hal penting untuk seorang sistem analis. *Automation boundary* adalah batas antara bagian sistem yang terotomatisasi dengan bagian sistem yang bersifat manual.

Jenis – jenis Sistem Informasi menurut Satzinger, Jackson, & Burd (2010, p9) sebagai berikut

A. *Customer Relationship Management System*
B. *Supply Chain Management System*
C. *Accounting and Financial Management System*
D. *Human Resource Management System*
E. *Manufacturing Management System*
F. *Knowledge Management System*
G. *Collaboration Support System*
H. *Business Intelligence System*



Gambar 2.1 Jenis – jenis Sistem Informasi

Sumber : (Satzinger, Jackson, Burd, 2010, p9)

2.1.4 Komponen – Komponen Sistem Informasi

Sistem informasi terdiri dari sekelompok komponen yang saling berhubungan, bekerja sama untuk mencapai tujuan bersama dengan menerima input serta menghasilkan output dalam proses transformasi yang teratur (Ladjamudin, 2005, p14).

Kerangka konsep berbagai komponen dan aktivitas sistem informasi dapat dilihat pada gambar di bawah ini :

Gambar 2.2 Komponen-Komponen Sistem Informasi

Sumber : (Ladjamudin, 2005, p15)

Berdasarkan gambar di atas, komponen-komponen sistem informasi terdiri dari :

A. Manusia / *User*

Manusia diperlukan dalam operasi sistem informasi. Sumber daya manusia ini meliputi pemakai akhir dan pakar sistem. Pemakai akhir adalah orang yang menggunakan informasi yang dihasilkan sistem informasi, misalnya pelanggan, pemasok, teknisi, mahasiswa, dosen dan orang-orang yang berkepentingan. Sedangkan pakar sistem informasi adalah orang yang mengembangkan dan mengoperasikan sistem informasi, misalnya *system analyst*, developer, operator sistem dan staf administrasi lainnya (Mulyanto, 2009, p31).

B. Perangkat Keras / *Hardware*

Sumber daya hardware adalah semua peralatan yang digunakan dalam memproses informasi, misalnya komputer dan periferalnya, lembar kertas, *disk magnetic* atau optik dan *flash disk* (Mulyanto, 2009, p31).

C. Perangkat Lunak / *Software*

Software merupakan sekumpulan perintah/fungsi yang ditulis dengan aturan tertentu untuk memerintahkan komputer agar melaksanakan sesuatu (Ladjamudin, 2005, p15).

D. Data

Data merupakan dasar sumber daya organisasi yang diperlukan untuk memproses informasi. Data dapat berbentuk teks, gambar, audio maupun video. Sumber daya informasi umumnya diatur, disimpan dan diakses oleh berbagai pengelolaan sumber daya data ke dalam database dan dasar pengetahuan (Ladjamudin, 2005, p15).

E. Jaringan / *Network*

Sumber daya jaringan merupakan media komunikasi yang menghubungkan komputer, pemroses komunikasi dan peralatan lainnya dengan kendali software komunikasi. Jaringan dapat berupa kabel, satelit, seluler dan pendukung jaringan seperti modem, software pengendali serta prosesor antar jaringan (Ladjamudin, 2005, p15).

Keseluruhan komponen sistem informasi tersebut saling terkait satu sama lain dalam sistem informasi. Sistem informasi dibangun menggunakan teknologi komunikasi dan informasi yaitu hardware, software dan jaringan. Ketiga komponen tersebut dipakai untuk mengolah data yang diperoleh untuk menghasilkan informasi yang lebih bermanfaat. Keseluruhan proses pengolahan informasi tidak lepas dari komponen manusia. Manusia adalah komponen penting sistem informasi karena sistem informasi adalah benda yang tidak bermanfaat bila tidak digunakan oleh manusia.

2.1.5 Pengertian Database

Menurut O' Brien dan Marakas(2009, p196) database adalah kumpulan terintegrasi dari elemen data yang secara logika saling berhubungan. Berbeda dengan Connolly & Begg (2005, p15) yang mengatakan bahwa database adalah kumpulan relasi-relasi logikal dari data, dan deskripsi dari data yang dapat digunakan bersama dan dibuat untuk memperoleh informasi yang dibutuhkan oleh perusahaan.

2.1.6 Pengertian Aplikasi

Aplikasi merupakan suatu subekelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas yang diinginkan pengguna. Biasanya dibandingkan dengan perangkat lunak sistem yang mengintegrasikan berbagai kemampuan komputer.

Terdapat beberapa teori yang mendefinisikan Aplikasi yang dikemukakan oleh beberapa para ahli, diantaranya adalah :

a) Menurut Hengky W. Pramana (2012, p17) Aplikasi adalah satu unit perangkat lunak yang dibuat untuk melayani kebutuhan akan beberapa aktivitas seperti sistem perniagaan, game, pelayanan masyarakat, periklanan, atau semua proses yang hampir dilakukan manusia.

b) Menurut Shelly, Cashman, Verman (2009, p57) Aplikasi adalah seperangkat instruksi khusus dalam komputer yang dirancang agar kita menyelesaikan tugas-tugas tertentu.

c) Menurut Dhanta (2009, p32), aplikasi adalah *software* yang dibuat oleh suatu perusahaan komputer untuk mengerjakan tugas – tugas tertentu.

Jadi Aplikasi merupakan sebuah program yang dibuat dalam sebuah perangkat lunak dengan komputer untuk memudahkan pekerjaan atau tugas-tugas tertentu seperti penerapan, penggunaan, dan penambahan data yang dibutuhkan.

Menurut O'Brien (2010, p124) *software* komputer terbagi menjadi 2 yaitu :

1. *Application Software*

   Mengarahkan kinerja penggunaan tertentu atau aplikasi komputer untuk memenuhi kebutuhan pengolahan informasi dari pengguna.

   Contoh : *word processing*.

2. *System software*

   Mengendalikan dan mendukung operasi dari sistem komputer karena melakukan berbagai tugas pengolahan informasi.

   Contoh : *operating system, network management, database management.*

2.1.7 Pengertian Aplikasi berbasis Web

Menurut O'Brien (2010, p157), *Web service* merupakan komponen *software* yang berbasis *framework web* dan standar *object-oriented* dan teknologi untuk penggunaan web yang secara elektronik menghubungkan aplikasi user yang berbeda dan *platform* yang berbeda. *Web service* dapat menghubungkan fungsi bisnis untuk pertukaran data secara *real time* dalam aplikasi berbasis web.

Banyak dari perusahaan - perusahaan berkembang yang menggunakan Aplikasi berbasis Web dalam merencanakan sumber daya mereka dan untuk mengelola perusahaan mereka. Aplikasi berbasis Web ini menggunakan protokol *HTTP*, aplikasi di sisi server berkomunikasi dengan *client* melalui *Web server.* Aplikasi di sisi *client* umumnya berupa *Web browser* jadi. Aplikasi berbasis *Web(client / server-side script)* berjalan di atas aplikasi berbasis internet.

Menurut Simarmata (2010, p185), aplikasi berbasis web adalah sistem perangkat lunak yang berdasarkan pada teknologi dan standar *World Wide Web Consortium (W3C)*. Mereka menyediakan sumber daya web spesifik seperti konten dan layanan melalui sebuah antarmuka pengguna dan *browser web.*

2.1.7.1 Tujuan Aplikasi Berbasis Web

Tujuan Aplikasi berbasis web, yaitu :

1. Aplikasi berbasis web dapat digunakan untuk membantu operasional perusahaan seperti membuat *invoice,* sistem informasi persediaan.
2. Memudahkan dalam penyimpanan data di *database*
3. Aplikasi berbasis web juga dapat bekerja memonitoring sistem dalam hal tampilan, dapat didesain dan disesuaikan untuk berbagai jenis industri

2.1.7.2 Kelebihan dan Kelemahan Aplikasi berbasis Web

Kelebihan kompetitif dari Aplikasi berbasis Web :

1. Aplikasi tersebut ringan dan dapat diakses selama ada koneksi internet atau intranet ke server.
2. Dapat diakses dengan menggunakan *browser* tanpa harus menginstall aplikasi tersebut.

Kekurangan menggunakan Aplikasi berbasis web :

1. Antarmuka yang dapat dibuat terbatas sesuai spesifikasi standar untuk membuat dokumen *Web* dan keterbatasan kemampuan *Web browser* untuk menampilkannya.
2. Terbatasnya kecepatan internet mungkin membuat respon aplikasi menjadi lambat.
3. Tingkat keamanan yang lebih rentan untuk diakses oleh orang lain atau pihak yang tidak berhak.

   Oleh karena itu dapat disimpulkan, *user* dapat mengakses data atau informasi perusahaan mereka melalui laptop, *smartphone,*

atau bahkan komputer PC dengan mudah tanpa harus menginstal terlebih dahulu aplikasi tersebut.

2.1.8 Pengertian PHP (HyperText Preprocessor)

Menurut Doyle (2009, p3), PHP adalah bahasa pemrograman untuk membuat situs web dinamis dan interaktif. PHP berjalan di server web dan melayani pengunjung dengan halaman web sesuai permintaan. Artinya adalah sintaks dan perintah-perintah yang kita berikan akan sepenuhnya dijalankan di server tetapi disertakan HTML biasa.

PHP dikenal sebgai bahasa scripting yang menyatu dengan tag HTML, dieksekusi di server dan digunakan untuk membuat halaman web yang dinamis seperti ASP (Active Server Pages) dan JSP (Java Server Pages).

Tujuan dari bahasa scripting ini adalah untuk membuat aplikasi-aplikasi yang dijalankan diatas teknologi web. Dalam hal ini, aplikasi pada umumnya akan memberikan hasil pada web browser, tetapi prosesnya secara keseluruhan dijalankan web server.

Kelebihan PHP Ketika e-commerce semakin berkembang, situs-situs yang statispun semakin ditinggalkan, karena dianggap sudah tidak memenuhi keinginan pasar, padahal situs tersebut harus tetap dinamis. Pada saat ini bahasa PERL dan CGI sudah jauh ketinggalan jaman sehingga sebagian besar designer web banyak beralih ke bahasa server-side scripting yang lebih dinamis seperti PHP. Seluruh aplikasi berbasis web dapat dibuat dengan PHP. Namun kekuatan yang paling utama PHP adalah pada konektivitasnya dengan system database didalam web.

2.1.9 Rich Picture

Berdasarkan pendapat Tan dan Lambe (2008, p116), *Rich Picture is a way to representing knowledge domains where you want to communicate different perspectives and concerns in the same space.*

Jadi, *rich picture* merupakan suatu gambaran umum mengenai proses bisnis yang ada di dalam perusahaan termasuk pelaku yang melakukan

kegiatan tersebut secara berkaitan untuk memudahkan memahami proses bisnis perusahaan tersebut.

Dalam hal ini adalah gambar keseluruahn orang, objek, dan proses bisnis yang ada di perusahaan. Tujuan *Rich Picture* adalah untuk memudahkan memahami proses bisnis perusahaan tersebut.

2.2 Teori Khusus

Teori – teori khusus yang digunakan untuk mendukung hasil analisis yang telah dilakukan. Teori khusus ini harus berhubungan dengan topik yang dibahas untuk menjelaskan atau mendefinisikan teori – teori yang dipakai dalam pembuatan skripsi ini.

2.2.1 Pengertian Surat

Surat adalah alat komunikasi tertulis yang berasal dari satu pihak dan ditujukan kepada pihak lain untuk menyampaikan warta (Barthos, 2003, p36). Sedangkan menurut Gie (2000, p15), surat adalah setiap bentuk catatan tertulis atau bergambar yang memuat keterangan mengenai sesuatu hal atau peristiwa yang dibuat orang untuk membantu ingatannya. Surat bersifat praktis yang artinya dapat menyimpan rahasia, efektif artinya sesuai dengan keadaan yang sebenarnya dan ekonomis artinya biaya pembuatan, peralatan dan pengirimannya murah.

Menurut Silmi (2002, p12) surat adalah sehelai kertas atau lebih yang digunakan untuk mengadakan komunikasi secara tertulis. Surat merupakan jembatan pengertian dan alat komunikasi bagi seseorang dan orang lain. Karena sifatnya yang demikian, maka surat-surat harus disusun secara singkat dan padat tetapi jelas dan tegas. Bahasa yang dipakai haruslah mudah dimengerti, sederhana dan teratur.

Surat masuk adalah surat yang masuk ke dalam suatu instansi atau bagian dalam suatu instansi, baik yang berasal dari instansi lain atau dari bagian lain pada instansi yang sama. Dengan demikian surat masuk dapat berasal dari pihak eksternal maupun pihak internal instansi tersebut.

Macam-macam surat antara lain (Barthos, 2003, p37) :

1. Macam-macam surat dibedakan menurut wujudnya antara lain : Kartu pos, Warkat pos, Surat bersampul, Memorandum dan Nota, Telegram, Surat pengantar.
2. Macam-macam surat dibedakan berdasarkan tujuannya antara lain : Surat pemberitahuan, Surat perintah, Surat permintaan, Surat peringatan, Surat panggilan, Surat susulan, Surat keputusan, Surat laporan, Surat perjanjian, Surat penawaran, pesanan dan lain-lain.
3. Macam-macam surat menurut sifat isi dan asalnya dibedakan sebagai berikut: Surat dinas, Surat niaga, Surat pribadi, Surat yang isinya masalah sosial.
4. Macam-macam surat menurut jumlah penerimanya dibedakan sebagai berikut : Surat biasa, surat ini untuk satu orang (pejabat/ organisasi), Surat edaran, untuk beberapa orang/ pejabat/ organisasi, Surat pengumuman, untuk sekelompok masyarakat.
5. Macam-macam surat menurut keamanan isinya dibedakan sebagai berikut: Surat sangat rahasia, Surat rahasia, Surat biasa.
6. Macam-macam surat menurut urgensi penyelesaiannya dibedakan sebagai berikut : Surat sangat rahasia, Surat segera, Surat biasa.
7. Surat menurut prosedur pengurusannya dibedakan menjadi beberapa macam yaitu : Surat masuk, Surat keluar.
8. Surat menurut jangkauannya dibedakan menjadi beberapa macam antara lain : Surat Intern, Surat Ekstern.

2.2.2 Pengertian Disposisi

Disposisi merupakan petunjuk singkat tentang tindak lanjut atau penyelesaian terhadap suatu urusan atau surat masuk (Vironica dan Sukadi, 2014, p35). Disposisi adalah petunjuk tertulis mengenai tindak lanjut

pengolahan surat bersama lembar disposisi diantarkan oleh kurir ke dinas. Apabila diperlukan, pejabat yang bersangkutan dapat melakukan disposisi lanjutan kepada bawahannya untuk ditindak lanjuti lagi dan bila sudah terlaksana maka pelaksana akan memberikan laporan kepada pemimpin yang telah memberikan disposisi.

Lembar Disposisi adalah suatu formulir yang disertakan pada surat masuk sebagai sarana untuk mencantumkan disposisi/pengarahan/catatan dari pejabat yang berwenang atau yang memperoleh delegasi wewenang untuk menyelesaikan atau menangani substansi surat serta sebagai sarana pengendalian yang berisi informasi perkembangan penanganan surat atau berkas (Setneg, p118).

2.2.3 Pengertian *Dashboard*

Menurut Turban, *et al.* (2011, p137) *Dashboard* adalah komponen yang umumnya memiliki *Performance Management Systems, Performance Measurement Systems, BPM Suites,* dan *BI Platforms*. *Dashboard* menyediakan tampilan visual dari informasi penting yang disatukan dan diatur dalam sebuah layar tunggal sehingga informasi dapat dipahami cukup dengan sekali lihat, serta mudah untuk dieksplorasi.

Turban, *et al*. (2011, p138) mengutip dari Eckerson (2006), yang adalah pakar yang terkenal dalam bidang BI dan *dashboard* secara umum, fitur yang paling dapat dibedakan dari sebuah *dashboard* adalah tiga lapisan informasinya, yaitu:

- *Monitoring*

  Data abstrak dan grafis untuk memonitor *key performance metrics.*

- *Analysis*

  Dimensi data yang dirangkum untuk dianalisis akar permasalahannya.

- *Management*

  Data operasional yang terperinci yang mengidentifikasi tentang tindakan yang harus dilakukan untuk menyelesaikan masalah.

Menurut Turban, *et al.* (2011, p139), semua *dashboard* yang dirancang dengan baik akan memiliki karakteristik sebagai berikut:

- *Dashboard* menggunakan komponen visual untuk menggarisbawahi secara sekilas, data dan pengecualian yang membutuhkan tindakan.
- *Dashboard* bersifat transparan terhadap pengguna, yang artinya pengguna cukup membutuhkan sedikit pelatihan dan mudah untuk menggunakan *dashboard* tersebut.
- *Dashboard* menggabungkan data dari berbagai macam sistem menjadi sebuah tampilan bisnis yang tunggal, ringkas, tergabung menjadi satu.
- *Dashboard* memungkinkan *drill-down* atau *drill-through* terhadap sumber data atau laporan yang ada dan menyediakan konteks yang dapat dibandingkan dan dievaluasi secara lebih terperinci.
- *Dashboard* menyediakan sebuah tampilan dinamis dan nyata dari data yang diperbaharui secara berkala.
- *Dashboard* memungkinkan pengguna untuk tetap mendapatkan informasi baru tentang setiap perubahan di dalam bisnis.
- *Dashboard* membutuhkan sedikit perubahan kode program untuk dikirim, diimplementasikan, dan dirawat.

Adapun pengertian *Dashboard* menurut Johan, *et al.* (2004, p93), *Dashboard* adalah sebuah aplikasi bagi *knowledge workers* yang berguna untuk menampilkan informasi yang sesuai dengan kebutuhan dalam satu layar. Informasi yang ditampilkan telah dipilah dan digabungkan dari berbagai sumber informasi yang ada. *Dashboard* merupakan bagian dari *Business Intelligence* yang secara langsung akan menampilkan berbagai informasi yang dibutuhkan oleh suatu organisasi atau perusahaan dengan beragam bentuk seperti grafik dan indikator warna yang dapat memudahkan pengguna dalam mengambil keputusan secara cepat dan tepat. *Dashboard*

yang dibuat mampu menyediakan informasi - informasi yang dibutuhkan secara akurat bagi pihak software maintenance. *Dashboard* membuat suatu bagian di dalam suatu perusahaan agara lebih merespon setiap perubahan dan perkembangan yang ada. Semua informasi tersebut ditampilkan dengan tujuan agar para penggunanya mampu mengendalikan proses kerjanya dengan baik dan mampu membuat keputusan yang tepat pada saat dibutuhkan. Menurut Johan, *et al.* (2004, p94), terdapat beberapa keuntungan dari *dashboard* antara lain :

1. Membuat pengguna dalam menentukan informasi yang utama dan penting. *Dashboard* membantu memilah informasi yang tidak diperlukan sehingga hanya menampilkan informasi yang penting.

2. Mendapatkan informasi yang telah terintegrasi. *Dashboard* dapat memberikan kemudahan dengan menampilkan informasi yang penting dalam satu layar.

3. Tetap mendapatkan informasi terkini meskipun tidak terhubung ke jaringan. Informasi tetap dapat diakses dari tempat yang berbeda baik secara offline maupun online.

4. Membantu meningkatkan kualitas dari keputusan yang diambil. Informasi yang ditampilkan secara cepat dan tepat akan membantu menghasilkan keputusan yang berkualitas.

.

2.3 User Interfaces

Mengacu pada pendapat Satzinger, Jackson, & Burd (2010, p530) User Interfaces merupakan bagian dari suatu sistem informasi yang memerlukan interaksi dari pengguna untuk membuat input dan output. User Interfaces digunakan untuk menghubungkan manusia dengan komputer sehingga dapat melakukan suatu pekerjaan yang berhubungan langsung dengan sistem. Ben Shneiderman mengemukakan delapan aturan yang dapat digunakan sebagai petunjuk dasar merancang suatu user interfaces. Delapan aturan ini disebut dengan “8 Golden Rules”, yaitu:

1. *Strive For Consistency*

   Konsistensi dilakukan pada urutan tindakan, perintah, dan istilah yang digunakan pada prompt, menu, serta layar bantuan.

2. *Enable Frequent Users To Use Shortcuts*

   Ada kebutuhan dari user yang sudah ahli untuk meningkatkan kecepatan dalam berinteraksi, sehingga diperlukan singkatan, tombol fungsi, perintah tersembunyi, dan fasilitas makro.

3. *Offer Informative Feedback*

   Untuk setiap action yang dilakukan oleh user, sebaiknya disertakan suatu feedback dari komputer agar user mengetahui action yang sedang dilakukan.

4. *Design Dialogs To Yield Closure*

   Urutan action yang akan dilakukan oleh user sebaiknya diorganisir dengan jelas, baik di awal, tengah, maupun di akhir sehingga user dapat mengetahui posisi saat mengerjakan tugasnya.

5. *Offer Simple Error Handling*

   Sedapat mungkin sistem dirancang untuk menghindari dan mengantisipasi kesalahan yang dilakukan oleh user. Jika kesalahan terjadi, sistem dapat mendeteksinya dengan cepat dan memberikan mekanisme yang sederhana dan mudah dipahami oleh user untuk penanganan kesalahan.

6. *Permit Easy Reversal of Actions*

   Hal ini dapat mengurangi kekhawatiran user, karena user dapat melakukan pembatalan action yang telah dilakukannya, sehingga user tidak akan takut untuk mengekplorasi pilihan-pilihan lain yang belum biasa digunakan.

7. *Support Internal Locus Of Control*

   Pada umumnya, user ingin menjadi pengontrol sistem dan sistem akan merespon tindakan yang dilakukan oleh user. Sebaiknya sistem dirancang sedemikan rupa sehingga user akan menjadi inisiator bukan responden. Selain itu, user tidak akan merasa di kontrol oleh sistem.

8. *Reduce Short-Term Memory Load*

   Manusia memiliki ingatan yang terbatas. Oleh karena itu, rancangan user interfaces yang dibuat harus mudah diingat serta diberikan petunjuk-petunjuk dasar untuk membantu user dalam mengerjakan tugas.

2.4 Analisa dan Perancangan Sistem Informasi

Dalam menganalisa dan merancang system informasi , metode yang digunakan adalah metode Object Oriented Analisis dan design (OOAD) menurut Satzinger.

2.4.1 Pengertian Object-Oriented Analysis and Design (OOAD)

Berpedoman dari Satzinger (2010, p60) pengertian Object Oriented Analysis and Design (OOAD) merupakan pengembangan sistem yang berpandangan bahwa sistem informasi adalah kumpulan objek yang saling berinteraksi dan bekerja sama untuk mencapai tujuan.

2.4.2 Pengertian Unified Modelling Language (UML)

Unified Modelling Language (UML) adalah bahasa yang digunakan untuk memspesifikasikan, memvisualisasikan, membangun, dan mendokumentasikan sebuah sistem informasi dan proses bisnis suatu organisasi. Unified Modeling Language (UML) merupakan satu sekumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem software yang terkait dengan objek.

Hal ini diperkuat oleh pernyataan Satzinger (2010, p240) bahwa Unified Modelling Language adalah satu set standar dari model dan notasi yang dikembangkan khusus untuk pengembangan berorientasi objek.

2.4.2.1 Activity Diagram

Activity Diagram merupakan tipe dari diagram workflow yang menjelaskan aktifitas-aktifitas pengguna dan arus tahapan aktvitasnya. (Satzinger, 2010, p141)

Dalam menggambarkan activity diagram terdapat beberapa simbol yang digunakan, yaitu:

1. *Synchronization bar*

   Merupakan notasi yang digunakan untuk mengontrol pemisahan atau penyatuan dari jalur yang berurutan

2. *Swimlane*

   Merupakan suatu daerah persegi dalam activity diagram yang mewakili aktivitas-aktivitas yang diselesaikan agen tunggal

3. *Starting activity (pseudo)*

   Merupakan notasi yang menandakan dimulainya sebuah aktivitas

4. *Transition arrow*

   Merupakan garis penunjuk panah yang menggambarkan transisi dari suatu aktivitas dan arah dari suatu aktivitas.

5. *Activity*

   Merupakan notasi yang menggambarkan suatu aktivitas

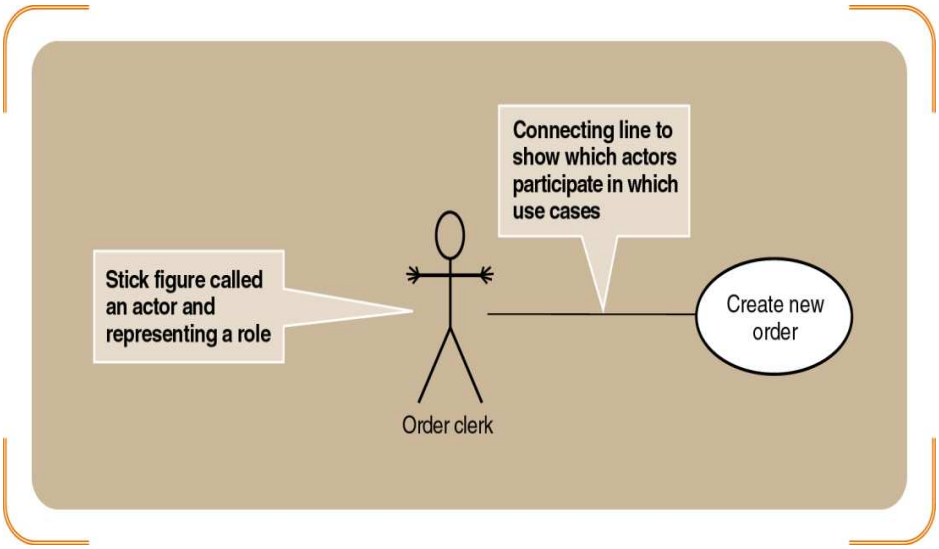
6. *Ending activity (pseudo)*

   Merupakan notasi yang menandakan berakhirnya suatu aktivitas

2.4.2.2 *Usecase* Diagram

Mengacu pada pendapat Satzinger, Jackson, & Burd (2010, p242) pengertian *Use Case* Diagram dikemukakan sebagai urutan langkah-langkah atau aktivitas yang terjadi ketika *actor* berinteraksi dengan sistem untuk suatu tujuan dari tugas tertentu, aktifitas yang dilaksanakan oleh sistem, biasanya dalam menanggapi permintaan oleh pengguna sistem. *Use Case* adalah pola untuk interaksi antara *actor* dengan sistem dalam application domain.

Jadi dapat disimpulkan bahwa *Use Case* adalah suatu pola urutan langkah-langkah yang menggambarkan interaksi antara sistem dan *actor* yang berhubungan dalam application domain.



Gambar 2.3 *Use case* sederhana dengan *actor*

Sumber*: (Satzinger et al.*, 2010, p243)

Gambar 2.3 menunjukkan bagaimana *use case* didokumentasi dalam sebuah *use case* diagram. *Actor* digambarkan dengan *simple stick* yang diberi nama yang mengkarakteristikkan peran yang dimainkan oleh *actor*. *Use case* digambarkan dengan bentuk oval dengan nama dari aktivitas yang dilakukan di dalamnya. Sedangkan garis penghubung (*connecting line*) menghubungkan *actor* dengan *use case* yang dilakukannya. Tangan dari *actor* mengindikasikan akses sistem langsung, *actor* harus dapat langsung mengakses ke sistem secara otomatis.

Di dalam *use case*, terdapat dua relasi yang menjelaskan. Hubungan antara *use case*, yaitu: *includes* dan *extends*.

- *Includes*

  Dalam pengembangan *use case* sangat memungkinkan lebih dari satu *use case* menggunakan *use case* yang sudah menjadi rutinitasnya. Sebagai contoh adalah Membuat *Order*. Dalam membuat *order* pasti diperlukan pengecekan ketersediaan barang. Oleh karena itu, Mengecek Ketersediaan Barang secara otomatis dijalankan bila *use case* Membuat *Order* dilakukan. Hubungan antara kedua *use case* tersebut dapat didenotasikan dengan garis penghubung (*connecting line*) dan panah. Arah dari panah

mengindikasikan *use case* mana yang menjadi *include* sebagai bagian dari *use case* utama. Relasinya dibaca menjadi Membuat *Order* <<*include*>> Mengecek Ketersediaan Barang.

- Extends

  Relasi <<*extends*>> memungkinkan *user* untuk memodifikasi *use case* dasar. Misalkan, *user* ingin menjual produk yang telah dibuat untuk memesan dan memerlukan tingkat spesifikasi pelanggan. Untuk produk ini, *user* perlu mencatat persyaratan tambahan pelanggan, seperti ukuran dan warna. Dalam hal ini, user menambahkan sesuatu yang ekstra untuk aliran *use case* dasar.

  Relasi «extends» menyatakan bahwa *user* menjalankan *use case* utama tetapi ketika *user* sampai di titik tertentu dalam aliran, jika kondisi yang tepat terpenuhi, *user* melakukan beberapa langkah yang berbeda. Jelas ini sangat mirip dengan Arus Alternatif. Keuntungannya adalah bahwa Arus Alternatif dan setiap subnya telah pindah ke *use case* yang terpisah.

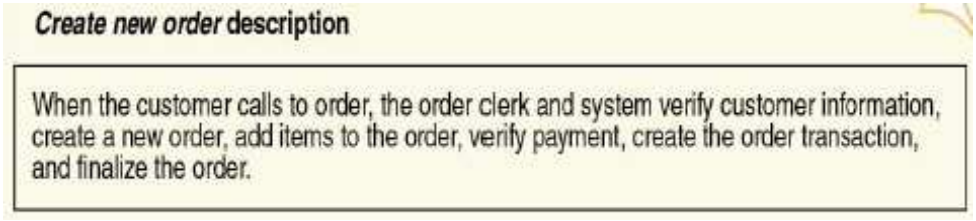
2.4.2.3 *Use Case* Descriptions

Mengacu pada pendapat Satzinger (2010) dapat dikemukakan bahwa membuat *use case*diagram merupakan bagian kecil dari analisis *use case*. *Use case* diagram membantu dalam identifikasi bermacam proses yang dilakukan oleh user dan yang harus didukung oleh sistem yang baru. Namun pengembangan sistem yang baik membutuhkan deskripsi atau diagram yang lebih rinci. Untuk membuatnya, harus mengerti sepenuhnya mengenai langkah-langkah itu secara terperinci.

Use Case Description merupakan perincian dari *use case* yang telah dibuat. *Use Case* Description dibagi menjadi tiga level berdasarkan tingkat kerinciannya, yaitu :

- **Brief Description**

Brief Description dapat digunakan untuk *use case* yang sangat sederhana, khususnya saat sistem yang dikembangkan sangat sederhana, aplikasi yang sangat mudah dipahami. Pada umumnya *use case* sederhana mempunyai satu skenario dan beberapa skenario, bila beberapa kemungkinan (exceptions condition) akan terjadi. Brief Description digunakan dalam konjungsi dengan activity diagram yang menggambarkan *use case* sederhana.



Gambar 2.4 *Brief Description* dari *use case*

Create New Order

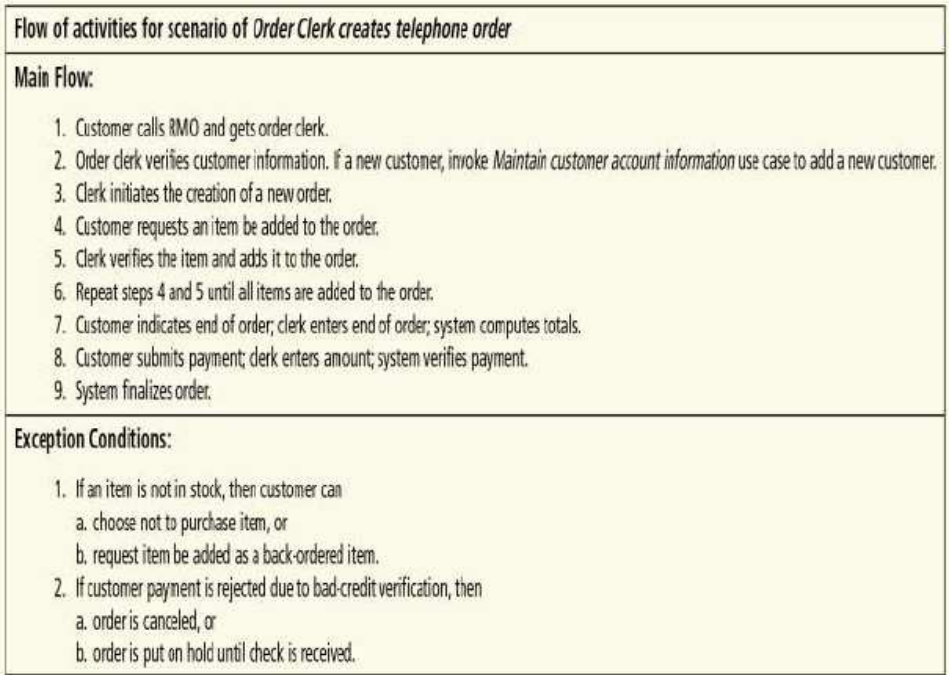
Sumber: (Satzinger *et al*., 2010, p246)

Gambar 2.4 menunjukkan brief description dari *use case* Create New *Order*. Pada umunya, *use case* seperti Create New *Order* termasuk cukup kompleks untuk dikembangkan dengan Intermediate atau Fully Developed Description.

- **Intermediate Description**

Intermediate Description memperluas brief description untuk mencakupi alur internal dari aktivitas *use case* tersebut. Bila terdapat beberapa skenario, maka setiap alur aktivitas dideskripsikan masing-masing. Exception conditions dapat didokumentasi bila dibutuhkan.

Gambar 2.5 menunjukkan intermediate description yang mendokumentasikan *Order Clerk* membuat pesanan telepon. Setiap deskripsi menjelaskan apa yang user dan sistem perlukan untuk memproses skenario tersebut. Exception conditions juga dijelaskan pada deskripsi. Setiap langkahnya ditandai dengan nomor untuk memudahkan dalam membacanya.



Flow of activities for scenario of *Order Clerk creates telephone order*

Main Flow:

1. Customer calls RMO and gets order clerk.
2. Order clerk verifies customer information. If a new customer, invoke *Maintain customer account information* use case to add a new customer.
3. Clerk initiates the creation of a new order.
4. Customer requests an item be added to the order.
5. Clerk verifies the item and adds it to the order.
6. Repeat steps 4 and 5 until all items are added to the order.
7. Customer indicates end of order; clerk enters end of order; system computes totals.
8. Customer submits payment; clerk enters amount; system verifies payment.
9. System finalizes order.

Exception Conditions:

1. If an item is not in stock, then customer can
   a. choose not to purchase item, or
   b. request item be added as a back-ordered item.
2. If customer payment is rejected due to bad-credit verification, then
   a. order is canceled, or
   b. order is put on hold until check is received.

Gambar 2.5 *Intermediate Description* dari *Telephone Order Scenario* untuk *use case Create New Order*

Sumber : (Satzinger, *et al,* 2010, p247)

- **Fully Developed Description**

Fully Developed Description merupakan cara paling umum untuk mendokumentasi *use case*. Walaupun diperlukan kerja ekstra untuk mendefinisikan semua komponen pada level ini guna mendeskripsikan alur internal dari aktivitas sebuah *use case*. Salah

satu kesulitan terbesar yaitu membuat software pengembangan yang digunakan agar dapat mengerti kebutuhan user.

Use Case Name:	Create new order	
Scenario:	Create new telephone order	
Triggering Event:	Customer telephones RMO to purchase items from the catalog.	
Brief Description:	When customer calls to order, the order clerk and system verify customer information, create a new order, add items to the order, verify payment, create the order transaction, and finalize the order.	
Actors:	Telephone sales clerk	
Related Use Cases:	Includes: *Check item availability*	
Stakeholders:	Sales department: to provide primary definition Shipping department: to verify that information content is adequate for fulfillment Marketing department: to collect customer statistics for studies of buying patterns	
Preconditions:	Customer must exist. Catalog, Products, and Inventory items must exist for requested items.	
Postconditions:	Order and order line items must be created. Order transaction must be created for the order payment. Inventory items must have the quantity on hand updated. The order must be related (associated) to a customer.	
Flow of Events:	**Actor**	**System**
	1. Sales clerk answers telephone and connects to a customer.	
	2. Clerk verifies customer information.	
	3. Clerk initiates the creation of a new order.	3.1 Create a new order.
	4. Customer requests an item be added to the order.	
	5. Clerk verifies the item (*Check item availability* use case).	5.1 Display item information.
	6. Clerk adds item to the order.	6.1 Add an order item.
	7. Repeat steps 4, 5, and 6 until all items are added to the order.	
	8. Customer indicates end of order; clerk enters end of order.	8.1 Complete order.
		8.2 Compute totals.
	9. Customer submits payment; clerk enters amount.	9.1 Verify payment.
		9.2 Create order transaction.
		9.3 Finalize order.
Exception Conditions:	2.1 If customer does not exist, then the clerk pauses this use case and invokes *Maintain customer information* use case. 2.2 If customer has a credit hold, then clerk transfers the customer to a customer service representative. 4.1 If an item is not in stock, then customer can a. choose not to purchase item, or b. request item be added as a back-ordered item. 9.1 If customer payment is rejected due to bad-credit verification, then a. order is canceled, or b. order is put on hold until check is received.	

Gambar 2.6 Fully Developed Description dari Telephone OrderScenario untuk use case Create New Order

Sumber : (Satzinger, *et al*, 2010, p248*)*

Gambar 2.6 merupakan contoh Fully Developed Description dari Telephone *Order Scenario* untuk *use case* Create New *Order*. Gambar di atas juga dapat ditampilkan sebagai template standar untuk mendokumentasi fully developed description untuk skenario dan *use case* lainnya.

Bagian pertama dan kedua digunakan untuk identifikasi *use case* dan skenario yang didokumentasikan. Dalam proyek yang lebih besar atau lebih umum, pengidentifikasi unik dapat juga dimasukkan ke

dalam *use case*, dengan identifikasi ekstensi bagian dari skenario tersebut.

Bagian ketiga mengidentifikasi *trigger* (pemacu) yang menginisiasi *use case*. *Trigger* di sini sama dengan *trigger* yang terdapat pada *event table.* Berikut ini terdapat dua hal untuk mendeskripsikan *trigger*. Yang pertama adalah mengidentifikasi kegiatan bisnis yang menginisiasi proses. Yang kedua adalah aktivitas yang menyebabkan sistem ter-otomatisasi untuk menandakan bahwa *use case* dimulai.

Bagian keempat merupakan brief description dari *use case* atau skenario. Seorang analis hanya menduplikasi brief description yang telah dibuat sebelumnya. Bagian kelima mengidentifikasi *actor*. Bagian ini menduplikasi beberapa informasi yang terdapat di *use case* diagram itu sendiri. Bagian keenam mengidentifikasikan *use case* lain dan bagaimana *use case* lain berhubungan dengan *use case* tersebut, seperti relasi <<*include*>>.

Bagian *stakeholders* mengidentifikasi pihak-pihak yang terlibat dalam *use case* atau skenario tersebut. Dapat juga berupa user yang secara tidak langsung terlibat dalam *use case* namun mereka yang terlibat dalam proses dan yang menerima hasil dari *use case* tersebut.

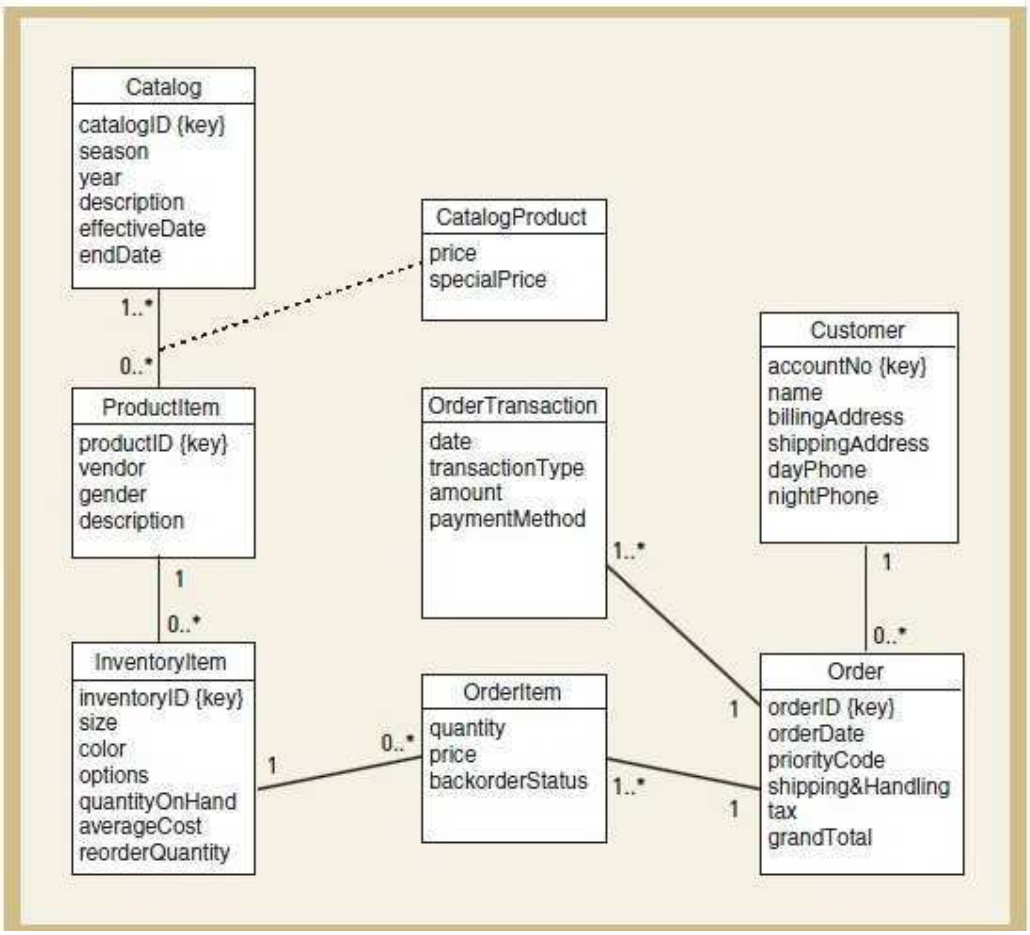
Dua bagian berikutnya menunjukkan informasi penting mengenai status keadaan sistem sebelum dan setelah *use case* dilakukan, disebut preconditions dan postconditions. Preconditions menunjukkan keadaan yang harus terjadi sebelum *use case* dimulai. Postconditions

menunjukkan keadaan yang seharusnya terjadi dalam melengkapi *use case*.

Dua bagian terakhir mendekripsikan alur aktivitas dari *use case* secara terperinci. Dalam bagian ini terbagi menjadi dua kolom versi, yaitu kolom untuk mengidentifikasikan langkah-langkah yang dilakukan oleh *actor* dan respon yang diperlukan oleh sistem. Penomoran membantu dalam menentukan urutan dari langkah-langkah. Alternate activities dan exception conditions dideskripsikan dengan langkah yang spesifik dalam *use case* description.

2.4.2.4 Domain Model Class Diagram

Domain Model Class Diagram Sebuah kelas UML diagram yang menunjukkan hal-hal yang penting dalam pekerjaan pengguna: masalah domain classes, asosiasi, dan atributnya. Problem domain classes bukanlah software classes, meskipun mereka digunakan untuk merancang software classes sebagai rancangan system dan telah terimplementasi (Satzinger, 2010, p195)



Gambar 2.7 ***Domain Model Class diagram (Satzinger,*** **2010, p188)**

Hubungan antar class yang digambarkan dengan garis penghubung disebut *multiplicity of association*. Hubungan antar class ini dapat dibedakan menjadi enam jenis yang digambarkan dalam table sebagai berikut:

Tabel 2.1 Hubungan relasional antar class

Sumber : Satzinger *(*2010, p188)

Hubungan	**Simbol**
Zero to one (optional)	0..1
One and only one (mandatory)	1
One and only one alternate (mandatory)	1..1
Zero or more (optional)	0..*
Zero or more alternate (optional)	*
One or more (mandatory)	1..*

2.4.2.5 Navigation Diagram

Mengacu pada pendapat Mathiassen (2000) *Navigation Diagram* dikemukakan terdiri dari penurunan gambar dari *window* lain, dan panah yang menunjukkan bagaimana menggunakan *button* dan pilihan lain akan mengaktifkan fungsi atau membuka *window* lain.

2.4.3 Interface

Menurut Satzinger et al. (2010, p47) interface adalah tempat dimana sistem informasi menangkap input dan menghasilkan output, serta terjadinya input dan output antara System dan lingkungannya. Ada dua tipe dari interface yaitu user interface dan System interface. User interface bagian dari sistem informasi yang membutuhkan interaksi dari user untuk menghasilkan input dan output.

Shneiderman (2009, p251) mendekripsikan panduan untuk desain interaksi yang baik dalam "*The Eight Golden Rules for Designing Interactive Interface*", yaitu:

1. *Strive for consistency* (konsistensi)

   Konsistensi dilakukan pada urutan tindakan, perintah, dan istilah yang digunakan pada prompt, menu, serta layar bantuan.

2. *Enable frequent users to use shortcuts* (memungkinkan pengguna untuk menggunakan shortcuts)

   Ada kebutuhan dari pengguna yang sudah ahli untuk meningkatkan kecepatan interaksi, sehingga diperlukan singkatan, tombol fungsi, perintah tersembunyi, dan fasilitas makro.

3. *Offer information feedback* (memberikan umpan balik yang informative)

   Untuk setiap tindakan operator, sebaiknya disertakan suatu sistem umpan balik. Untuk tindakan yang sering dilakukan dan tidak terlalu penting, dapat diberikan umpan balik yang sederhana. Tetapi ketika tindakan merupakan hal yang penting, maka umpan balik sebaiknya lebih substansial. Misalnya muncul suatu suara ketika salah menekan tombol pada waktu input data atau muncul pesan kesalahannya.

4. *Design dialogs to yield closure* (merancang dialog untuk menghasilkan suatu penutupan)

   Urutan tindakan sebaiknya diorganisir dalam suatu kelompok dengan bagian awal, tengah, dan akhir. Umpan balik yang informatif akan memberikan indikasi bahwa cara yang dilakukan sudah benar dan dapat mempersiapkan kelompok tindakan berikutnya.

5. *Offer simple error handling* (memberikan penanganan kesalahan yang sederhana)

   Sedapat mungkin sistem dirancang sehingga pengguna tidak dapat melakukan kesalahan fatal. Jika kesalahan terjadi, sistem dapat

mendeteksi kesalahan dengan cepat dan memberikan mekanisme yang sedehana dan mudah dipahami untuk penanganan kesalahan.

6. *Permits easy reversal of actions* (mudah kembali ke tindakan sebelumnya)

   Hal ini dapat mengurangi kekhawatiran pengguna karena pengguna mengetahui kesalahan yang dilakukan dapat dibatalkan; sehingga pengguna tidak takut untuk mengekplorasi pilihan-pilihan lain yang belum biasa digunakan.

7. *Support internal focus of control* (mendukung tempat pengendalian internal)

   Pengguna ingin menjadi pengontrol sistem dan sistem akan merespon tindakan yang dilakukan pengguna daripada pengguna merasa bahwa sistem mengontrol pengguna. Sebaiknya sistem dirancang sedemikan rupa sehingga pengguna menjadi inisiator daripada responden.

8. *Reduce short-term memory load* (mengurangi beban ingatan jangka pendek

   Keterbatasan ingatan manusia membutuhkan tampilan yang sederhana atau banyak tampilan halaman yang sebaiknya disatukan, serta diberikan cukup waktu pelatihan untuk kode, *mnemonic*, dan urutan tindakan.