BAB II
TINJAUAN PUSTAKA

2.1 Konsep Sistem informasi

2.1.1 Sistem

Sistem adalah sekelompok unsur yang erat hubungannya satu dengan yang lain yang berfungsi bersama-sama untuk mencapai tujuan tertentu. Pendekatan sistem yang lebih menekankan pada prosedur menurut Jogiyanto, sistem adalah suatu jaringan kerja prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran tertentu [1].

2.1.2 Karakteristik Sistem

Model umum sebuah sistem terdiri dari input, proses dan output. Hal ini merupakan konsep sebuah sistem yang sangat sederhana. Sebuah sistem memiliki karakteristik sebagai berikut:

1. Komponen *(Components)*

   Sebuah sistem terdiri dari serangkaian komponen yang saling berinteraksi, bekerjasama membentuk satu kesatuan.

2. Batasan sistem *(Boundary)*

   Ruang lingkup sistem merupakan daerah yang membatasi antar sistem yang satu dengan sistem lainnya atau sistem dengan lingkungan luarnya.

3. Lingkungan Luar Sistem *(Environtment)*

   Lingkungan luar sistem merupakan bentnuk apapun yang ada di luar ruang lingkup atau bayasan sistem yang mempengaruhi operasi sitem tersebut.

4. Penghubung Sistem *(Interface).*

   Penghubung sistem adalah media yang menghubungkan sistem dengan subsistem yang lain.

5. Masukan sistem *(Input)*

   Masukan sistem merupakan energi yang dimasukkan kedalam sistem yang dapat berupa pemeliharaan *(maintenance)* dan sinyal *(signal input).*

6. Keluaran Sistem *(Output)*

   Keluaran sistem merupakan hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan merupakan masukan bagi subsistem yang lain.

7. Pengolah Sistem *(Process)*

   Suatu sistem dapat mempunyai suatu proses yang akan mengubah masukan menjadi keluaran.

8. Sasaran Sistem *(Objective)*

   Suatu sistem memiliki tujuan dan sasaran yang pasti dan bersifat *deterministic.* Suatu sistem dikatakan berhasil jika mengenai sasaran dan tujuan yang telah direncanakan.

2.1.3 Informasi

Menurut Gellinas dan Dull, informasi merupakan data yang disajikan dalam suatu bentuk yang berguna terhadap aktifitas pengambilan keputusan.

Menurut Romney dan Steinbart, informasi adalah data yang telah dikelola dan diproses untuk memberikan arti dan memperbaiki proses pengambil keputusan

Menurut Gelinas dan Dull, ada beberapa karakteristik informasi yang berkualitas, yaitu:

1. *Effectiveness*: berkaitan dengan informasi yang relevan dan berkaitan dengan proses bisnis yang disampaikan dengan tepat waktu, benar, konsisten dan dapat digunakan.
2. *Efficiency*: informasi yang berkaitan melalui penyediaan informasi secara optimal terhadap penggunaan sumber daya.
3. *Confidentiality*: karakteristik informasi yang berkaitan dengan keakuratan dan kelengkapan informasi serta validitasnya sesuai dengan nilai-nilai bisnis dan harapan.
4. *Integrity*: karakteristik informasi yang berkaitan dengan perlindungan terhadap informasi yang sensitif dari pengungkapan yang tidak sah.
5. *Availability*: suatu karakteristik informasi yang berkaitan denganinformasi yang tersedia pada saat diperlukan oleh proses bisnis baik sekarang, maupun dimasa

mendatang, hal ini juga menyangkut perlindungan sumber daya yang diperlukan dan kemampuan yang terkait.

6. *Compliance*: yaitu karakteristik informasi yang berkaitan denganmematuhi peraturan dan perjanjian kontrak dimana proses bisnis merupakan subjeknya berupa kriteria bisnis secara internal maupun eksternal.
7. *Reliability*: karakteristik informasi yang berkaitan dengan penyediaan informasi yang tepat bagi manajemen untuk mengoperasikan entitas dan menjalankan tanggung jawab serta tata kelola pemerintahan.

Dengan demikian dapat disimpulkan bahwa informasi adalah data yang diproses menjadi suatu bentuk yang lebih berguna dan berarti bagi yang menerimanya dalam aktivitas pembuatan keputusan [1].

2.1.4 Sistem Informasi

Sistem Informasi adalah suatu sistem didalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian yang mendukung fungsi operasi yang bersifat manajerial dengan kegiatan strategi dari suatu organisasi untuk dapat menyediakan kepada pihak luar tertentu dengan laporan-laporan yang diperlukan [11].

2.2 Analisis Dan Perancangan

Analisis sistem merupakan orang yang melakukan analisis sistem dengan mengidentifikasi permasalahan-permasalahan yang ada pada sistem serta menentukan pemecahan sebagai solusi dari permasalahan tersebut. Pada dasarnya seorang sistem analis harus mengerti tentang bisnis manajemen, mempunyai *skill* komputer [3].

Adapun tugas dari sistem analisis adalah :

1. Bertanggungjawab atas hasil analisis sistem secara keseluruhan.
2. Melakukan analisis terhadap masalah yang timbul dan memberikan solusi dari permasalahan tersebut.
3. Menentukan kebutuhan sistem.
4. Mengimplementasikan sistem.
5. Menyediakan kebutuhan data ketika pembuatan sistem (*software).*

6. Menyiapkan dokumentasi teknis maupun proses dalam pengembangan sistem.

2.3 Kamus

Kamus merupakan daftar kata-kata suatu bahasa. Biasanya, isi kamus tersusun secara alfabetis. Namun, kadang-kadang juga berdasarkan topic, dengan makna atau bentuk yang setara. Sebuah kamus biasanya berisi cara pelafalan, pola suku kata, etimologi (asal-usul kata), dan contoh penggunaan. Daftar sistematik istilah khusus juga disebut kamus. Contoh kamus tersebut adalah kamus singkatan (kamus akronim), kamus dialek Jakarta, asal-usul kata(*etimologi*), atau suatu daftar yang mendefinisikan istilah khusus bidang ilmu tertentu, seperti kamus fisika, kamus geografi, kamus peternakan, kamus matematika, dan kamus filsafat.

Kamus yang pertama diketahui tersimpan dikota Mesopotamia, Elba (sekarang terletak di Syaria). Kamus yang ditulis sekitar tahun 2300 SM itu berbentuk tablet yang terbuat dari keramik. Tulisan kuno yang ada didalamnya dibagi dalam kolom-kolom. Kamus tersebut berisi kata-kata dalam bahasa Sumeria dan padannya dalam bahasa Akkadia [2].

2.4 Kesehatan

Berdasarkan Undang-Undang No. 9 tahun 1960 tentang pokok-pokok kesehatan Bab 1 pasal 2 disebutkan bahwa pengertian sehat atau kesehatan adalah sebagai berikut:

Kesehatan yang meliputi kesehatan fisik, mental, dan sosial dan bukan hanya keadaan bebas dari penyakit, cacat, dan kelemahan". Pengertian sehat tersebut sesuai dengan pengertian sehat yang dikemukakan oleh badan kesehatan dunia, yang dikenal dengan istilah *World Health Organization* (WHO).Berdasarkan pengertian tersebut dapat disimpulkan bahwa kesehatan yang sesungguhnya adalah kesehatan yang sempurna baik jasmani, rohani, maupun sosial.

Maksud dari kesehatan atau kesegaran jasmani yakni seseorang yang dapat melakukan suatu aktivitas atau kegiatan (pekerjaan) yang cukup berat dalam waktu yang cukup lama tampa mengalami kelelahan yang berarti. Artinya, setelah melakukan berbagai kegiatan tersebut ia masih mampu melakukan aktivitas lainnya

secara segar dan bugar (*fitness*). Untuk itu, perlu dijelaskan berbagai jenis aktivitas jasmani yang sering dilakukan oleh sebagian masyarakat kita dan berguna bagi kesehatan, misalnya senam pagi Indonesia, senam tera, aerobik, jalan pagi, *jogging,* renang, tenis meja, tenis lapangan, sepak bola, bola voli, bola basket, dan lain sebagainya [8].

2.5 Alat Bantu Pengembangan Sistem

2.5.1 *Use Case Diagram*

Use case diagram adalah rangkaian atau uraian sekelompok yang saling terkait dan membentuk sistem secara teratur yang dilakukan atau diawasi oleh sebuah *actor*. *Use case* digunakan untuk membentuk tingkah laku benda dalam sebuah model serta direalisasikan oleh sebuah kolaborasi.

Diagram *use case* menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Hal yang ditekankan pada diagram ini adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah aktivitas atas pekerjaan tertentu, misalnya *login* kesistem, meng-*create* sebuah daftar belanja, dan lain sebaginya. Aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu.

Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain. Oleh karena itu, duplikasi fungsionalitas yang *common.* Sebuah *use case* juga daoat meng-*extend* *use case* lain dengan *behavior*-nya sendiri. Hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain [4].

2.5.2 Menyusun Diagram *Use Case*

Langkah-langkah yang dibutuhkan untuk menyusun diagram *use case :*

1. Mengidentifikasi Pelaku Bisnis

   Pelaku bisnis diidentifikasi terlebih dahulu supaya kita dapat berkonsentrasi pada bagaimana sistem itu akan digunakan dan bagaimana sistem itu akan dibangun. Dengan mengidentifikasi pelaku bisnis, lingkup dan batasan sistem dapat kita saring dan identifikasi lebih lanjut. Pelaku juga menentukan kelengkapan persyaratan sistem. Mengenali terlebih dahulu pelaku sistem membuat kita dapat mengidentifikasi kandidat narasumber untuk wawancara. Selain itu, pengenalan

pelaku sistem membantu kita saat observasi untuk menyelesaikan pemodelan *use case*.

2. Mengidentifikasi *Use Case* Persyaratan Bisnis

   Use case persyaratan bisnis mengakap interaksi dengan pengguna menggunakan cara yang bebas dari detail teknologi dan implementasi. *Use* case menggambarkan bagaimana para pelaku sebenarnya berinteraksi dengan sistem. Teknik yang baik untuk mencari *use case* persyaratan adalah menelidiki para pelaku dan bagaimana mereka akan menggunakan sistem tersebut.

3. Membuat Diagram Model *Use Case*

   Setelah *use case* dan pelaku teridentifikasi, diagram model *use case* pun dapat digunakan untuk menggambarkan secara grafis, baik lingkup dan batasan sistem.

4. Mendokumentasikan Naratif *Use Case* Persyaratan Bisnis

   Untuk setiap *use case* tingkat tinggi yang teridentifikasi, kita harus mengembangkannya supaya dapat menyertakan kejadian umum *use case* dan bagian alternatifnya. Bagian kejadian umum *use case* adalah deskripsi langkah demi langkah dimulai dengan pelaku menginisiasi *use case* dan melanjutkannya hingga akhir kejadian bisnis. Dalam bagian ini, kita hanya menyertakan langkah utama yang terjadi pada sebagian besar waktu tersebut (bagian umumnya). Bagian alternatifnya mendokumentasikan *exception* atau *conditional branching* pada *use case*.

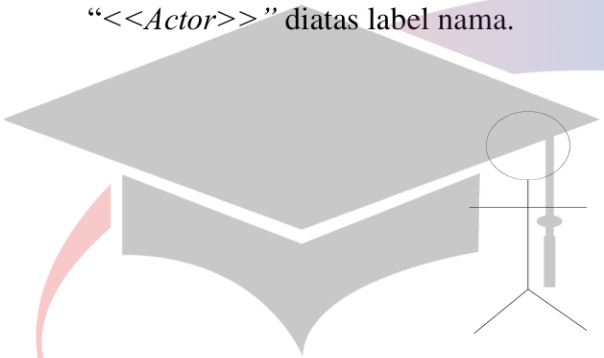
2.5.3 Sistem

Sistem menyatakan batasan sistem dalam relasi dengan *actor-actor* menggunakannya (diluar sistem) dan fitur-fitur yang harus disediakan (dalam sistem). Sistem digambarkan dengan segi empat yang membatasi semua *use case* dalam sistem terhadap pihak mana sistem akan berinteraksi. Sistem disertai label yang menyebutkan nama dari sistem, tapi umumnya tidak digambarkan karena tidak terlalu member arti tambahan pada diagram.

2.5.4 *Actor*

Actor atau aktor dapat berupa manusia, sistem, atau *device* yang memiliki peranan dalam keberhasilan operasi dari sistem. Digambarkan dengan *icon* yang mungkin bervariasi namun konsepnya sama:

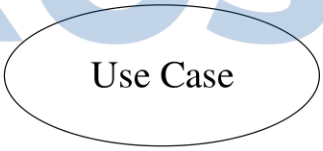
1. Umumnya, untuk orang, digambarkan dengan sosok lengkap seperti, dengan kepala, badan, tangan, dan kaki.
2. Umumnya, untuk sistem, digambarkan dengan segi empat disertai notasi "<<*Actor*>>" diatas label nama.



Gambar 2.1 *Actor*

2.5.5 *Use Case*

Use case mengidentifikasi fitur kunci dari sistem. Tanpa fitur ini, sistem tidak akan memenuhi permintaan *user/actor.* Setiap *use case* mengekspresikan *goal* dari sistem yang harus dicapai. Diberi nama sesuai dengan *goal*-nya dan digambarkan dengan *elips* (dengan nama didalamnya). Fokus tetap pada *goal,* bukan "bagaimana" mengimplementasikannya walaupun *use case* berimplikasi pada prosesnya nanti.



Gambar 2.2 Use Case

2.5.6 *Assosiation*

Mengidentifikasikan interaksi antara setiap *actor* tertentu dengan setiap *use case* tertentu. Digambarkan dengan garis antara *actor* terhadap *use case* yang bersangkutan. Asosiasi bisa berarah (garis dengan anak panah) jika komunikasi satu

arah, namun umumnya terjadi kedua arah (tanpa anak panah) karena selalu diperlukan demikian.

Gambar 2.3 *Association* Tanpa Panah

Gambar 2.4 *Association* Dengan Panah

2.5.7 *Stereotape*

Memungkinkan perluasan UML tanpa memodifikasinya. Berperan sebagai kualifier pada suatu elemen model. Menyediakan informasi lebih banyak mengenai peranan dari elemen tanpa menyebutkan implementasinya. Terutama untuk menggambarkan :

1. *Use case dependency*
2. *Class-Class*
3. *Package-Package*
4. *Classifier*

2.5.8 *Dependency*

Dependensi <<*include*>>

Mengidentifikasi hubungan antar dua *use case* dimana yang satu memanggil yang lain.

1. Jika pada beberapa *use case* terdapat bagian yang memiliki aktivitas yang sama maka bagian aktivitas tersebut biasanya dijadikan *use case* tersendiri dengan relasi dependensi setiap *use case* semula ke *use case* yang baru ini sehingga memudahkan pemeliharaan.
2. Digambarkan dengan garis putus-putus bermata panah dengan notasi <<*include*>> pada garis.
3. Arah mata panah sesuai dengan arah pemanggilan.
4. Dependensi <<*extend*>>
   a. Jika pemanggilan memerlulkan adanya kondisi tertentu maka berlaku dependensi <<*extend*>>

b. *Note :* konsep "*extend"* ini berbeda dengan "*extend"* dalam java!
c. Digambarkan serupa dengan dependensi <<*include*>> kecuali arah panah berlawanan.

2.5.9 *Generalization*

1. Mendefinisikan relasi antara dua *actor* atau dua *use case.* Salah satunya meng-*inherit* dan menambahkan atau override sifat dari yang lainnya.
2. Penggambaran menggunakan garis bermata panah kosong dari yang meng-*inherit* mengarah ke yang di-*inherit.*

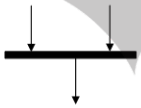
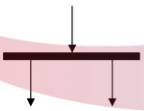
Sebagai acuan dalam membangun diagram use case, dapat menerapkan langkah-langkah:

1. Set konteks dari target sistem.
2. Identifikasi semua *actor.*
3. Identifikasi semua *use case.*
4. Definisikan asosiasi antara setiap *actor* dan setiap *use case.*
5. Evaluasi setiap *actor* dan seiap *use case* untuk mendapatkan kemungkinan *refinement,*
6. Evaluasi setiap *use case* untuk dependensi <<*include*>>.
7. Evaluasi setiap *use case* untuk dependensi <<*extend*>>
8. Evaluasi setiap *actor* dan setiap *use case* untuk generalisasi [4].

2.6 *Activity Diagram*

Activity Diagram memodelkan *workflow* proses bisnis dan urutan aktivitas dalam sebuah proses. Diagram ini sangat mirip dengan *flowchart* karena memodelkan *workflow* dari satu aktivitas ke aktivitas lainnya atau dari aktivitas kestatus. Membuat *activity* diagram pada awal pemodelan proses cukup menguntungkan untuk membantu memahami keseluruhan proses. *Activity Diagram* juga bermanfaat untuk menggambarkan *parallel behavior* atau menggambarkan interaksi antara beberapa *use case* [4].

Berikut adalah simbol-simbol yang sering digunakan pada penggambaran *Activity Diagram.*

No	Gambar	Nama	Keterangan
1		*State*	Aktivitas yang terjadi di dalam *Activity*
2		*Control Flow*	*State* dari sistem yang mencerminkan eksekusi dari suatu aksi.
3		*Initial State*	Proses dimulai pertama kali didalam *Activity*
4		*Final State*	Proses terakhir dalam *Activity*
5		*Transition (Join)*	Menunjukan kegiatan yang digabungkan
6		*Transition (Fork)*	Kegiatan yang dilakukan secara *Parallel*
7		*Decision*	Menggambarkan cabang suatu keputusan

Gambar 2.5 *Activity Diagram*

2.7 PIECES

Untuk mengidentifikasi masalah, harus dilakukan analisis terhadap kinerja, informasi, ekonomi, keamanan aplikasi, efisiensi, dan pelayanan pelanggan. Panduan ini dikenal dengan analisis PIECES *(performance, information, economy, control, efisiency,* dan *services).* Dari analisis ini biasanya didapatkan beberapa masalah utama. Hal ini penting karena biasanya yang muncul di permukaan bukan masalah utama, tetapi hanya gejala dari masalah utama saja [11].

P = Analisis kinerja

Masalah kinerja terjadi ketika tugas-tugas bisnis yang dijalankan tidak mencapai sasaran. Kinerja diukur dengan jumlah produksi dan waktu tanggap.

I = Analisis Informasi

Evaluasi terhadap kemampuan sistem informasi dalam menghasilkan informasi yang bermanfaat perlu dilakukan untuk menyikapi peluang dan menangani masalah yang muncul.

E = Analisis Ekonomi

Alasan ekonomi merupakan motivasi paling umum bagi suatu proyek, pijakan dasar bagi suatu manager adalah biaya atau rupiah. Persoalan ekonomis dan peluang berkaitan dengan biaya. Biaya biasanya terdiri dari biaya tidak diketahui, biaya tidak dapat dilacak ke sumber dan biaya terlalu tinggi.

C = Analisis Keamanan

Tugas bisnis perlu dimonitor dan dibetulkan jika ditemukan kinerja yang dibawah standar. Kontrol dipasang untuk meningkatkan kinerja sistem, mencegah, atau mendeteksi kesalahan sistem, menjamin keamanan data, informasi dan persyaratan.

E = Analisis Efisensi

Efisiensi menyangkut bagaimana menghasilkan output sebanyak-banyaknya dengan input yang sekecil mungkin.

S = Layanan

Kualitas suatu sistem bisa dikatakan buruk apabila sistem menghasilkan produk yang tidak akurat, tidak konsisten, tidak dipercaya, tidak mudah dipelajari, tidak mudah digunakan, sistem canggung digunakan dan tidak fleksibel.

2.8 Basis Data

Basis data *(database)* dalam dunia computer, terutama oleh pemrogram (*programmer*) sudah tidak asing lagi karena seringkali disinggung dan berhubungan langsung. Namun untuk memudahkan memahami apa maksud dengan basis data, ada baiknya dibahas terlebih dahulu apa yang dimaksud dengan basis data.

Basis data merupakan gabungan *file* data yang dibentuk dengan hubungan/relasi yang logis dan dapat diungkapkan dengan catatan serta bersifat indenpen. Adapun basis data adalah: “Tempat berkumpulnya data yang saling berhubungan dalam suatu wadah (organisasi/perusahaan) bertujuan agar dapat mempermudah dan mempercepat untuk pemanggilan atau pemanfaatan kembali data tersebut”.[6]

Dalam pembuatan dan penggunaan basis data, terdapat 4 (empat) komponen dasar sistem basis data, yaitu:

1. Data

   Data yang digunakan dalam sebuah basis data, haruslah mempunyai ciri sebagai berikut:

   a. Data disimpan secara terintegrasi *(integrated)*, yaitu Database merupakan kumpulan dari berbagai macam *file* dari aplikasi-aplikasi yang berbeda yang disusun dengan cara menghilangkan bagian-bagian yang rangkap (redundant).
   b. Data dapat dipakai secara bersama-sama (shared), yaitu masing-masing bagian dari *database* dapat diakses oleh pemakai dalam waktu yang bersamaan, untuk aplikasi yang berbeda.

2. *Hardware* (perangkat keras)

   Terdiri dari semua peralatan perangkat keras computer yang digunakan untuk pengelolaan sistem *database,* seperti:

   a. Pelalatan untuk penyimpanan, disk, drum, dll
   b. Peralatan *input* dan *output*
   c. Peralatan komunikasi data, dll

3. *Software* (perangkat lunak)

   Berfungsi sebagai perantara (*interface)* antara pemakai dengan data fisik pada database, dapat berupa:

   a. *Database Management System* (DBMS)
   b. Program-program aplikasi dan prosedur-prosedur yang lain, seperti Oracle, SQL Server, MySQL, dll

4. *User* (Pengguna)

   Terbagi menjadi 3 klasifikasi:

   1. *Database Administrator* (DBA), yaitu orang/*team* yang bertugas mengelola sistem *database* secara keseluruhan
   2. *Programmer,* yaitu orang/*team* membuat apliksi yang mengakses *database* dengan menggunakan bahasa pemrograman.

3. *End user,* orang yang mengakses *database* melalui terminal dengan menggunakan *query language* atau program aplikasi yang dibuat oleh programmer.

2.9 *Prototyping*

Prototyping adalah suatu cara yang baik untuk mendapatkan umpan balik mengenai sistem yang diajukan dan mengenai bagaimana sistem tersebut tersedia untuk memenuhi kebutuhan informasi para pengguna [7].

Prototyping memberikan ide bagi pembuat maupun pemakai potensial tentang cara sistem berfungsi dalam bentuk lengkapnya. Proses menghasilkan sebuah *prototype* disebut *prototyping* [9].

Prototyping merupakan versi awal dari sistem perangkat lunak yang dapat dipakai untuk mendemonstrasikan konsep, mencoba pilihan desain dan umumnya menemukan lebih banyak mengenai masalah-masalah dan solusinya. Pengembangan *prototype* yang penting dilakukan agar biaya terkontrol dan user dapat beriksperimen dengan *prototype* pada tahap awal proses perangkat lunak [9].

2.9.1 Tahapan-tahapan dalam *Prototyping*

Tahapan *prototyping* adalah sebagai berikut:

1. Pengumpulan kebutuhan

   Pelanggan dan pengembang bersama-sama mendefinisikan format seluruh perangkat lunak, mengidentifikasikan semua kebutuhan, dan garis besar sistem yang akan dibuat.

2. Membangun *prototyping*

   Membangun *prototyping* dengan membuat perancangan sementara yang berfokus pada penyajian kepada pelanggan (misalnya dengan membuat input dan format output).

3. Evaluasi *protoptyping*

   Evaluasi ini dilakukan oleh pelanggan apakah *prototyping* yang sudah dibangun sudah sesuai dengan keinginann pelanggan. Jika sudah sesuai maka langkah 4 akan diambil. Jika tidak *prototyping* direvisi dengan mengulangu langkah 1, 2 , dan 3.

4. Mengkodekan sistem

   Dalam tahap ini *prototyping* yang sudah di sepakati diterjemahkan ke dalam bahasa pemrograman yang sesuai.

5. Menguji sistem

   Setelah sistem sudah menjadi suatu perangkat lunak yang siap pakai, harus dites dahulu sebelum digunakan. Pengujian ini dilakukan dengan pengujian Black Box.

6. Evaluasi Sistem

   Pelanggan mengevaluasi apakah sistem yang sudah jadi sudah sesuai dengan yang diharapkan . Jika ya, langkah 7 dilakukan; jika tidak, ulangi langkah 4 dan 5.

7. Menggunakan sistem

   Perangkat lunak yang telah diuji dan diterima pelanggan siap untuk digunakan [10].

2.9.2 Kelebihan dan Kelemahan *Prototyping*

Ada tiga kelebihan utama *prototyping* yang potensial untuk mengubah sistem lebih dini dalam masa perkembangannya, peluang untuk menghentikan pengembangan suatu sistem yang lebih mendekati apa yang dibutuhkan yang tidak berfungsi dan kemungkinan mengembangkan suatu sistem yang mendekati apa yang dibutuhkan dan diharapkan pengguna. Ketiga kelebihan-kelebihan tersebut saling berkaitan satu sama lain.

1. Mengubah Sistem Sejak Dini dalam Masa Perkembangannya

   Prototyping yang berhasil tergantung pada umpan balik pengguna sejak awal dan sering diajukan yang dapat digunakan untuk membantu memodifikasi sistem dan membuatnya menjadi lebih responsive terhadap apa yang benar-benar dibutuhkan. Sama-halnya dengan sistem-sistem lainnya, perubahan sejak dini tidak terlalu memakan biaya disbanding perubahan yang terlambat dilakukan dalam perkembangan proyek. Karena *prototype* bisa diubah beberapa kali dan karena kemampuan fleksibilitas dan adaptasinya ibarat jantung *prototyping,* umpan balik yang menyebabkan dilakukannya perubahan dalam sistem. Saat mengubah *prototype,* penganalisis tidak perlu mengkhawatirkan waktu tenaga kerja dan pemrograman yang dihabiskan dalam upayanya mengembangkan

sebuah sistem yang lengkap hanya untuk menemukan bahwa *prototype* memerlukan modifikasi-modifikasi tertentu.

2. Membatalkan Sistem-Sistem yang tidak Diharapkan

   Kemungkinan dilakukannya pembatalan sistem yang tidak sesuai harapan pengguna dan penganalisis, penghapusan sistem *prototype* secara permanen dilakukan bila sistem tersebut tidak berguna dan tidak mampu memenuhi syarat-syarat(dan tujuan-tujuan lain) yang telah ditetapkan sebelumnya. Meskipun pembatalan pembatalan *prototype* merupakan suatu keputusan yang tidak mudah dibuat, namun lebih baik dari pada menghamburkan waktu dan uang untuk proyek yang nyata-nyata tidak berfungsi.

3. Merancang Sebuah Sistem yang Sesuai dengan Kebutuhan dan Harapan Pengguna

   Kelebihan ketiga ialah sistem yang sedang dikembangkan harus lebih bisa memenuhi kebutuhan dan harapan pengguna. Beberapa studi mengenai sistem informasi yang tidak berhasil menuntut interval panjang antara penetapan syarat-syarat dan presentasi sistem yang sudah jadi, sistem-sistem ini gagal karena sudah biasa bagi penganalisis sistem untuk mengembangkan sistem terpisah dari pengguna selama periode yang kritis ini. Lebih baik berinteraksi dengan pengguna melalui siklus hidup pengembangan sistem. Bila tim anda membuat komitmen melibatkan pengguna pada seluruh fase proyek, *prototype* bisa digunakan sebagai perangkat yang interaktif yang menentukan sistem final untuk merefleksikan secara akurat syarat-syarat pengguna, salah satu cara untuk membantu pengguna semacam ini adalah dengan melibatkan mereka secara aktif dalam fase *prototyping*. Keputusan untuk menjaga agar *prototype* tetap berfungsi dibuat bila *prototype* masih sesuai dengan waktu pemrograman dan penganalisis yang dianggarkan, bila pengguna menganggap sistem tersebut bermanfaat dan bila memenuhi syarat-syarat informasi serta tujuan-tujuam yang telah ditetapkan [7].

Ada beberapa kelemahan dalam melakukan *prototyping*. Yang pertama ialah sulitnya mengatur *prototyping* sebagai suatu proyek dalam sistem yang lebih besar. Kedua, pengguna dan penganalisis bisa mengadopsi *prototype* sebagai suatu sistem

yang lengkap bila pada kenyataannya sistem tersebut tidak cukup memadai dan tidak pernah dimaksudkan sebagai suatu sistem yang sudah jadi.

1. Mengelolah Proyek

   Meskipun beberapa iterasi *prototype* diperlukan, perpanjangan *prototype* secara tak terbatas juga menciptakan masalah-masalah tersendiri. Penting bahwa tim penganalisis sistem merencanakan dan kemudian menjalankan rencana itu mengenai bagaimana umpan balik terhadap *prototype* akan dikumpulkan, dianalisis dan diinterpretasikan. Penetapan periode waktu yang spesifik dimana Anda dan para pembuat keputusan menggunakan umpan balik tersebut untuk mengevaluasi bagaimana baiknya kinerja *prototype.* Meskipun *prototype prized for* sifat evolusionernya, penganalisis tidak dapat mengizinkan *prototype to overtake fase-fase* lain dalam Siklus hidup pengembangan sistem. Dapatkan umpan balik dari pengguna secara berkala, tidak hanya sekali dan meminta mereka member saran-saran bagi peningkatan ataupun perubahan yang harus dilakukan dengan memuaskan. Umpan balik diarahkan keanggota tim penganalisis sistem untuk mendapatkan reaksi mereka dan kemungkinan modifikasi *prototype* agar lebih bisa memenuhi kebutuhan pengguna dengan lebih baik. Ingat bahwa modifikasi-modifikasi terhadap *prototype* harus diatur dengan jadwal yang ketat kira-kira hanya dalam satu atau dua hari lewat beberapa iterasi.

2. Memakai Sistem yang Belum Selesai Seola-olah Sebagai Sistem yang Selesai

   Kelemahan kedua adalah bila seandainya sistem yang dibutuhkan *badly* dan *welcomed readily, prototype* bisa diterima sebagai sistem yang belum selesai dan ditekankan pada layanannya saja tanpa perlu *refinement-refinement* yang diperlukan. Meskipun secara dangkal metode ini lebih tampak sebagai cara yang menarik untuk memotong upaya pengembangan, namun menjadi tidak menguntungkan perusahaan dan tim. Pengguna akan mengembangkan pola-pola interaksi dengan sistem *prototype* yang tidak sesuai dengan apa yang seharusnya terjadi dengan sistem yang selesai. Selain itu, sebuah *prototype* tidak akan mampu menampilkan semua fungsi yang diperlukan. Akhirnya, ketika pengguna menemukan kekurangan-kekurangan tersebut, reaksi yang tidak baik dari pengguna akan semakin berkembang bila *prototype* dipakai secara salah dan

diintegrasikan kedalam bisnis seolah-olah merupakan sistem yang sudah selesai [7].

2.10 *Android*

Android adalah sistem operasi untuk telepon seluler yang berbasis Linux. *Android* menyediakan platform terbuka bagi para pengembang buat menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Awalnya, Google Inc. membeli *Android* Inc. pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengembangkan *Android*, dibentuklah *Open Handset Aliance*, konsorsium dari 34 perusahaan peranti keras, perangkat lunak dan telekomunikasi, termasuk Google, HTC, Intel, Motorala, Qualcomm, *T-Mobile* dan Nvidia.

Pada saat perilisan perdana *Android*, 5 November 2007, Android bersama *Open Handset Alliance* menyatakan mendukung pengembangan standar terbuka pada perangkat *seluler*. Di lain pihak, Google merilis kode-kode *Android* di bawah ini lisensi *Apache*, sebuah lisensi perangkat lunak dan standar terbuka perangkat *seluler*.

Di dunia terdapat dua jenis distributor sistem operasi *Android*. Pertama yang mendapat dukungan penuh dari Google atau Google Mail Services (GMS) dan kedua adalah yang benar-benar bebas distribusinya tanpa dukungan langsung Google atau dikenal sebagai *Open Handset Distribution* (OHD).

Pada Juli 2000, Google bekerjasama dengan *Android* Inc., perusahaan yang berada di Palo Alto, California Amerika Serikat. Para pendiri Android Inc. bekerja pada Google, diantaranya Andy Rubin, Rich Miner, Nick Sears dan Chris White. Saat itu banyak yang menganggap fungsi Android Inc. hanyalah sebagai perangkat lunak pada telepon seluler. Sejak itu muncul rumor bahwa Google hendak memasuki pasar telepon seluler. Diperusahaan Google, tim yang dipimpin Rubin bertugas mengembangkan program perangkat seluler yang didukung oleh kernel Linux. Hal ini menunjukkan indikasi bahwa Google sedang bersiap menghadapi persaingan dalam pasar telepon seluler.

Telepon pertama yang memakai sistem operasi Android adalah HTC Dream, yang dirilis pada 22 oktober 2008. Pada penghujung tahun 2009 diperkirakan di

dunia ini paling sedikit terdapat 18 jenis telepon seluler yang menggunakan *Android* [5].

1. Android versi 1.1

   Pada 9 Maret 2009, Google merilis Android versi 1.1 ini dilengkapi dengan pembaruan estetis pada aplikasi, jam alarm, *voice search* (pencarian suara), pengiriman pesan dengan Gmail dan pemberitahuan email.

2. Android versi 1.5 (Cupcake)

   Pada pertengahan Mei 2009, Google kembali merilis telepon seluler dengan menggunakan Android dan SDK (*Software Development Kit*) dengan versi 1.5 (*Cupcake*). Terdapat beberapa pembaruan termasuk juga penambahan beberapa fitur dalam seluler versi ini yakni kemampuan merekam dan menonton video dengan modus kamera, mengunggah video ke Youtube dan gambar ke Picasa langsung dari telepon, dukungan Bluetooth A2DP, kemampuan terhubung secara otomatis ke headset Bluetooth, animasi layar dan keyboard pada layar yang dapat disesuaikan dengan sistem.

3. Android versi 1.6 (Donut)

   Donut (versi 1.6) dirilis pada September dengan menampilkan proses pencarian yang lebih baik disbanding sebelumnya, penggunaan baterai indicator dan control applet VPN. Fitur lainnya adalah galeri yang memungkinkan pengguna untuk memilih foto yang akan dihapus; kamera, camcorder dan galeri yang diintergrasikanl CDMA / EVDO, 802.1x, VPN, Gestures dan text to change speech(tidak tersedia pada semua ponsel; pengadaan resolusi VWGA).

4. Android versi 2.0/2.1(Éclair)

   Pada 3 Desember 2009 kembali diluncurkan ponsel Android dengan versi 2.0/2.1 (Eclair), perubahan yang dilakukan adalah pengoptimalan hardware, peningkatan Google Maps 3.1.2, perubahan UI dengan browser baru dan dukungan HTML 5, daftar kontak yang baru, dukungan flash untuk kamera 3,2 MP, digital Zoom dan Bluetooth 2.1. Untuk bergerak cepat dalam persaingan perangkat generasi berikut, Google melakukan investasi dengan mengadakan kompetisi aplikasi mobile terbaik (killer apps – aplikasi unggulan). Kompetisi ini berhadiah $25.000 bagi setiap pengembang aplikasi terpilih. Kompetisi diadakan selama dua tahap yang

tiap tahapnya dipilih 50 aplikasi terbaik. Dengan semakin berkembangnyadan semakin bertambahnya jumlah handset Android, semakin banyak pihak ketiga yang berminat untuk menyalurkan aplikasi mereka kepada sistem operasi Android. Aplikasi terkenal yang diubah ke dalam sistem operasi Android adalah Shazam, Backgrounds dan WeatherBug. Sistem operasi Android dalam situs Internet juga dianggap penting untuk menciptakan aplikasi Android asli, contohnya oleh *MySpace* dan *Facebook.*

5. Android versi 2.2 (Froyo: Frozen Yoghurt)

   Pada 20 Mei 2010, Android versi 2.2(*Froyo*) diluncurkan. Perubahan-perubahan umumnya terhadap versi-versi sebelumnya antara lain dukungan Adobe Flash 10.1, kecepatan kinerja dan aplikasi 2 sampai 5 kali lebih cepat, integrasi V8 Javascript engine yang dipakai Google Chrome yang mempercepat kemampuan rendering pada browser, pemasangan aplikasi dalam SD Card, kemampuan Wifi Hotspot portable dan kemampuan auto update dalam aplikasi Android Market.

6. Android versi 2.3 (Gingerbread)

   Pada 6 Desember 2010, Android versi 2.3 (*Gingerbread*) diluncurkan. Perubahan-perubahan umum yang didapat dari android versi ini antara lain peningkatan kemampuan permainan (*gaming*), peningkatan fungsi copy paste, layer antar muka(*User Interface*) didesain ulang, dukungan format video VP8 dan WebM, efek audio baru (reverb, equalization, headphone virtualization dan bass boost), dukungan kemampuan Near Field Communication (NFC) dan dukungan jumlah kamera yang lebih dari satu.

7. Android versi 3.0/31 (Honeycomb)

   Android Honeycomb dirancang khusus untuk tablet. Android versi ini mendukung ukuran layar yang lebih besar. User interface pada Honeycomb juga berbeda karena sudah didesain untuk tablet. Honeycomb juga mendukung multi prosesor dan juga akselerasi perangkat keras(hardware) untuk grafis. Tablet pertama yang dibuat dengan menjalankan Honeycomb adalah Motorala Xoom. Perangkat tablet dengan platform Android 3.0 akan segera hadir di Indonesia. Perangkat tersebut bernama Eee Pad Transformer produksi dari Asus. Rencana masuk pasar Indonesia pada mei 2011.

8. Android versi 4.0 (ICS : Ice Cream Sandwich)

   Diumumkan pada tanggal 10 Oktober 2011, membawa fitur Honeycomb untuk smartphone dan menambahkan fitur baru termasuk membuka kunci dengan pengenalan wajah, jaringan data pemantauan penggunaan dan control, terpadu kontak jaringan sosial, perangkat tambahan fotografi, mencari email secara offline dan berbagi informasi dengan menggunakan NFC.

Fitur yang tersedia di Android adalah:

1. Kerangka aplikasi: itu memungkinkan penggunaan dan penghapusan komponen yang tersedia.
2. Dalvik mesin virtual: mesin virtual dioptimalkan untuk perangkat mobile.
3. Grafik: grafik di 2D dan grafis 3D berdasarkan pustaka OpenGL.
4. SQLite: untuk penyimpanan data.
5. Mendukung media: audio, video dan berbagai format gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).
6. GSM, Bluetooth, EDGE, 3G dan WiFi (*Hardware dependent*).
7. Kamera, *Global Positioning System* (*GPS*), kompas dan *accelerometer* (tergantung *hardware*).

9. Android versi 4.1 (Jelly Bean)

   Android Jelly Bean diluncurkan pertama kali pada Juli 2012, dengan berbasis Linux Kernel 3.0.31. Terdiri dari Android 4.1 API Level 16, Android 4.2 API Level 17 , Android 4.3 API Level 18. Penamaan Jelly Bean mengadaptasi nama sejenis permen dalam beraneka macam rasa buah. Ukurannya sebesar kacang merah. Permen ini keras di luar tapi lunak di dalam serta lengket bila di gigit.

10. Android versi 4.4 (KitKat)

    Android 4.4 Kitkat API level 19.Google mengumumkan Android KitKat (dinamai dengan izin dari Nestle dan Hershey) pada 3 september 2013. Dengan tanggal rilis 31 Oktober 2013. KitKat merupakan merk sebuah coklat yang dikeluarkan oleh Nestle. Rilis berikutnya setelah nama KitKat diperkirakan banyak pengamat akan diberi nomor 5.0 dan dinamai ‘Key Lime Pie’.

11. Android versi 5.0 (Lolipop)

    pada 25 Juni 2014 resmi diluncurkan dengan versi stabil terbaru dari sistem operasi Android yang dikembangkan oleh Google,

yang pada saat ini mencakup versi antara 5.0 dan 5.1. Diresmikan pada 25 Juni 2014 saat Google I / O, dan tersedia secara resmi melalui Over-The-Air (OTA) update pada tanggal 12 November 2014, untuk memilih perangkat yang menjalankan distribusi Android dilayani oleh Google (seperti perangkat Nexus dan Google Play Edition). Kode sumbernya dibuat tersedia pada 3 November 2014.Salah satu perubahan yang paling menonjol dalam rilis Lollipop adalah user interface yang didesain ulang dan dibangun dengan yang dalam bahasa desain disebut sebagai "material design". Perubahan lain termasuk perbaikan pemberitahuan, yang dapat diakses dari lockscreen dan ditampilkan pada banner di bagian atas screen. Google juga membuat perubahan internal untuk platform, dengan Android Runtime (ART) secara resmi menggantikan Dalvik untuk meningkatkan kinerja aplikasi, dan dengan perubahan yang ditujukan untuk meningkatkan dan mengoptimalkan penggunaan baterai, yang dikenal secara internal sebagai Project Volta.

12. Android versi 6.0 (Marshmallow)

    Android 6.0 atau disebut dengan “Marshmallow” secara resmi dirilis pada bulan Oktober 2015. Pertama kali diperkenalkan Mei 2015 di bawah codename “Android M””, Android Marshmallow memperkenalkan model izin yang didesain ulang: sekarang ada hanya delapan kategori izin, dan aplikasi yang tidak lagi secara otomatis diberikan semua hak akses mereka ditentukan pada waktu instalasi. Sebuah sistem opt-in sekarang digunakan, di mana pengguna akan diminta untuk memberikan atau menolak izin individu (seperti kemampuan untuk mengakses kamera atau mikrofon) untuk aplikasi ketika mereka dibutuhkan. Aplikasi mengingat hibah izin mereka, dan mereka dapat disesuaikan oleh pengguna setiap saat. Model izin baru akan digunakan hanya oleh aplikasi yang dikompilasi untuk Marshmallow menggunakan kit pengembangan perangkat lunak(SDK) tersebut, sementara semua aplikasi lainnya akan terus menggunakan model izin sebelumnya.

13. Android versi 7.0 (Nougat)

    Nougat adalah versi Android termutakhir yang baru diperkenalkan pada ajang kumpul developer Google I/O, pertengahan 2016 ini. Beberapa lama setelahnya, Google menghadirkan Nougat secara resmi untuk publik. Pembaruan paling

mendasar pada versi Nougat adalah kehadiran Google Assistant yang menggantikan Google Now. Asisten digital tersebut lebih bisa diandalkan untuk. Fitur-fitur baru lainnya mencakup layar split-screen saat dipakai multitasking, serta fitur Doze yang telah dikenalkan di versi Android Marshmallow namun telah ditingkatkan. Android Nougat juga memiliki dukungan terhadap platform virtual reality terbaru Google.

14. Android versi 8.0 (Oreo)

dirilis sebagai preview pengembang pada Maret 2017, dengan gambar pabrik untuk perangkat Nexus dan Pixel saat ini. Pratinjau pengembang terakhir dirilis pada Juli 2017, dengan versi stabil yang dirilis pada Agustus 2017. Versi: 8.0 Fitur:

1. Project Treble, arsitektur modular yang membuatnya lebih mudah dan lebih cepat bagi pembuat perangkat keras untuk menghadirkan pembaruan Android.
2. Dukungan untuk emoji Unicode 10.0 dan penggantian semua emo berbentuk gumpalan dengan yang bulat dengan gradien dan garis besar..
3. Pengaturan dan Pengaturan Cepat yang didesain ulang dengan latar belakang putih dan hitam masing-masing.
4. Pengaturan yang direstrukturisasi dengan mengelompokkan kembali dalam beberapa bagian entri yang serupa [5].

Tabel 2.1 Statistik Pengguna *Android* per Versi

No	Versi	Nama Versi (*Code Name*)	Tahun Dikeluarkan
1	Versi 1.0	Versi 1.0	2008
2	Versi 1.5	CupCake	2009
3	Versi 1.6	Donut	2009
4	Versi 2.0	Éclair	2009
5	Versi 2.2	Froyo	2010
6	Versi 2.3	Gingerbread	2010
7	Versi 3.0	Honeycomb	2011
8	Versi 4.0	Ice Cream Sandwich	2011

9	Versi 4.1	Jelly Bean	2012
10	Versi 4.4	Kitkat	2013
11	Versi 5.0	Lollipop	2014
12	Versi 6.0	Marshmallow	2015
13	Versi 7.0	Nougat	2016
14	Versi 8.0	Oreo	2017