BAB II

TINJAUAN PUSTAKA

2.1 Definisi *User Centered Design*

UCD (*user Centered Design*) merupakan paradigma baru dalam pengembangan sistem berbasis web. UCD didefinisikan sebagai *"efficiency" The practice of designing a product so that users can perform required operation, service, and supportive tasks with a minimum of stress and maximum of efficiency*. Perancangan berbasis pengguna (*User Centered design = User Centered Design =* UCD) adalah istilah yang yang digunakan untuk untuk menggambarkan filosofi perancangan. Konsep dari UCD adalah pengguna sebagai pusat dari proses pengembangan sistem, dan tujuan/sifat-sifat, konteks dan lingkungan sistem semua didasarkan dari pengalaman pengguna. [4]

Prinsip yang harus diperhatikan dalam UCD adalah:

1. Fokus pada pengguna

   Perancangan harus berhubungan langsung dengan pengguna sesungguhnya atau calon pengguna, misalnya melalui *interview*, *survey*. Tujuannya adalah untuk memahami kognisi, karakter, dan sikap pengguna serta karakteristik.. Aktivitas utamanya mencakup pengambilan data, analisis dan integrasinya ke dalam informasi perancangan dari pengguna tentang karakteristik tugas, lingkungan teknis atau organisasi.

2. Perancangan terintegrasi

   Perancangan harus mencakup antarmuka pengguna, sistem bantuan (cara penggunaan), dan dukungan teknis seperti kebutuhan perangkat lunak dan perangkat keras yang dibutuhkan.

3. Dari awal berlanjut pada penggujian pengguna

   Satu-satunya pendekatan yang sukses dalam perancangan sistem yang berpusat pada pengguna adalah secara empiris dibutuhkan observasi tentang kelakuan pengguna, evaluasi umpan-balik yang cermat, wawasan pemecahan terhadap masalah yang ada, dan motivasi yang kuat untuk mengubah rancangan.

4. Perancangan interaktif.

   Sistem yang sedang dikembangkan harus didefinisikan, dirancang, dan melakukan beberapa kali test. Berdasarkan hasil test kelakuan dari fungsi, maka juga akan dapat ditarik kesimpulan mengenai tingkat keberhasilan produk (aplikasi) tersebut.

UCD adalah tentang partisipasi dan pengalaman manusia dalam proses perancangan. Pengguna adalah orang yang akan menggunakan sistem. Pengguna langsung biasa disebut pengguna akhir (*end user*) yang menggunakan sistem untuk menyelesaikan pekerjaannya. Pengguna tidak langsung adalah pengguna yang

menggunakan sistem untuk penggunaan yang lain seperti *system administrators*, *installers*, dan *demonstrators*.

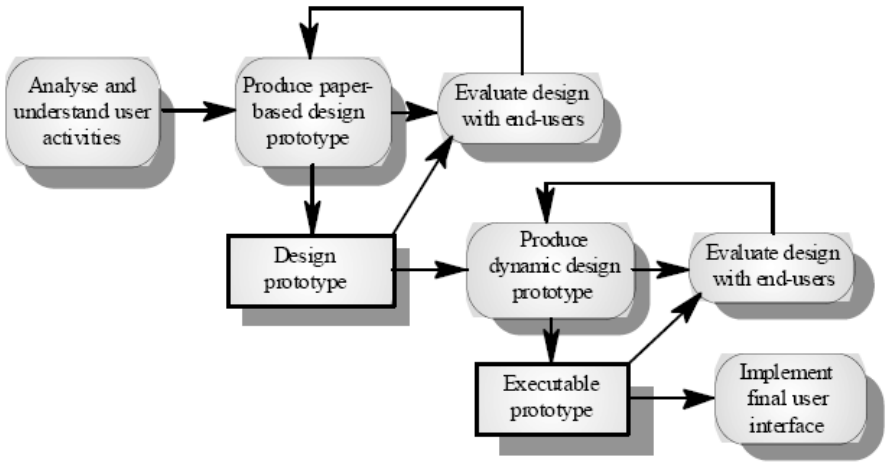
2.1.1 Konsep UCD

"*In user user-centered design, the users are the center focus."*

Maksud dari pernyataan di atas adalah tujuan/sifat sifat-sifat, konteks, penguna, sifat konteks dan lingkungan produk semua didasarkan dari pengalaman penguna Selanjutnya ditetapkan model pekerjaan pengguna yang akan didukung sistem sistem. Prinsip–prinsip dalam merancang *user interface* adalah sebagai berikut :

1. *User familiarity* / mudah dikenali : menggunakan istilah yang lazim digunakan dan dikenali oleh pengguna secara umum, misalnya: untuk sistem perkantoran gunakan istilah *letters, documents, folders* bukan *directories, file, identifiers*.
2. *Consistency* : Konsisten dalam operasi dan istilah yang digunakan di seluruh sistem sehingga tidak membingungkan.
3. *Minimal surprise* / Tidak membuat pengguna terkejut : Operasi yang terjadi dapat diprediksi oleh pengguna sesuai dengan perintah yang ada.
4. *Recoverability/* pemulihan : *Recoverability* ada dua macam yaitu : *confirmation of destructive action* (konfirmasi terhadap aksi yang merusak) atau ketersediaan fasilitas pembatalan (*undo).*
5. *User guidance*/ bantuan : Adanya cara penggunaan sistem sehingga pengguan lebih mudah dalam menjalankan sistem.

Prinsip perancangan UCD digambarkan pada gambar 2.1 .



Gambar 2.1 Prinsip Perancangan UCD

2.1.2 Aturan Dalam UCD

Karena telah mendefinisikan hak pengguna untuk mentransformasi budaya yang terdapat dalam perancangan, pengembangan, dan pembuatan sistem teknologi informasi, dan untuk memastikan bahwa produk hasilnya akan tepat seperti harapan pelanggan.[3]

Aturan dalam UCD (*User Centered Design*)

1. Perspektif

   Pengguna selalu benar. Jika terdapat masalah dalam penggunakan sistem, maka masalah ada pada sistem dan bukan pengguna.

2. Installasi

Pengguna mempunyai hak untuk dapat menginstall atau mengun-install perangkat lunak dan perangkat keras sistem secara mudah tanpa ada konsekuensi negatif.

3. Pemenuhan

   Pengguna mempunyai hak untuk mendapatkan sistem dapat bekerja seperti yang dijanjikan.

4. Instruksi

   Pengguna mempunyai hak untuk dapat menggunakan instruksi secara mudah untuk memahami dan menggunakan sistem untuk mencapai tujuan yang diinginkan secara efisien.

5. Umpan Balik

   Pengguna mempunyai hak terhadap sistem untuk menyediakan informasi yang jelas, dapat dimengerti, dan akurat tentang tugas yang dilakukan dan kemajuan yang dicapai.

6. Keterkaitan

   Pengguna mempunyai hak untuk mendapatkan informasi yang jelas tentang semua prasyarat yang dibutuhkan sistem untuk memperoleh hasil terbaik.

7. Batasan

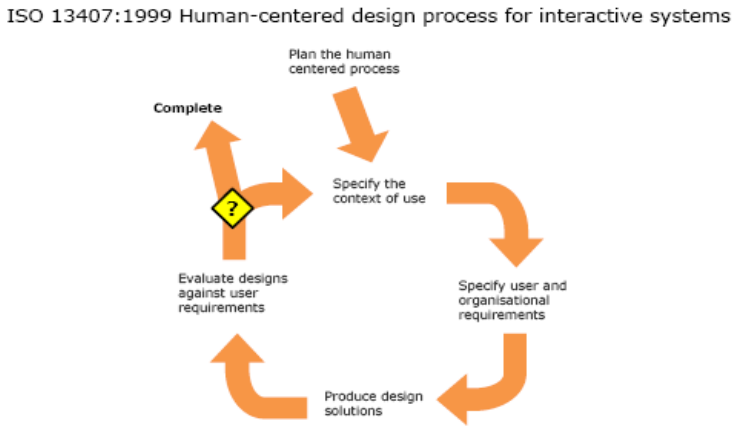
   Pengguna mempunyai hak untuk mengetahui batasan kemampuan sistem.

8. *Usability*

   Pengguna harus dapat menjadi penguasa teknologi perangkat lunak dan perangkat keras, dan bukan sebaliknya. Sistem harus dapat digunakan secara alami dan ituitif.

2.1.3 Proses UCD

Gambar 2.2 merupakan gambaran mengenai proses UCD (*user centered design*)



Gambar 2.2 Proses UCD

Keterangan gambar:

1. Memahami dan menentukan konteks pengguna
2. Menentukan kebutuhan pengguna
3. Solusi perancangan yang dihasilkan
4. Evaluasi perancangan terhadap kebutuhan pengguna.

2.1.4 Metode UCD

Metode yang digunakan dalam UCD adalah dengan melakukan aktivitas sebagai berikut :

1. Kuisioner

   Kuisioner merupakan salah satu teknik pengumpulan data yang dilakukan dengan cara memberikan seperangkat pertanyaan atau pernyataan tertulis kepada pengguna. Kuisioner menjadi efisien bila peneliti sudah dapat mengetahui dengan pasti variabel yang akan diukur dan tahu apa yang bisa diharapkan dari pengguna. Dengan hasil yang didapat dari kuisioner, maka peneliti dapat mengetahui kebutuhan pengguna.

2. *Interview*

   Interview dilakukan untuk berinteraksi dengan pengguna dengan maksud untuk mencari tahu apa yang dibutuhkan pengguna dari sistem yang akan dibangun. Hal ini dilakukan sebagai studi pendahuluan untuk menemukan permasalahan yang akan diteliti. *Interview* dapat dilakukan secara terstruktur dan tidak terstruktur, dan dapat dilakukan melalui tatap muka maupun menggunakan alat komunikasi seperti telepon.

3. *Task Modelling*

   Proses menganalisis dan menggambarkan bagaimana pengguna melaksanakan tugas tugasnya pada sistem, apa saja dapat yang dilakukan dan hal-hal apa saja yang perlu diketahui oleh pengguna. Memeriksa tugas-tugas

pengguna untuk mengetahui dengan baik apa yang dibutuhkan pengguna dari *interface* dan bagaimana pengguna akan menggunakannya.

4. *Prototyping*

   Prototipe adalah suatu proses untuk membangun solusi perancangan yang kongkrit yang berawal dari pengguna dan kebutuhan pengguna. Mewakili model produk yang akan dibangun, mensimulasikan struktur, fungsionalitas, atau operasi sistem. Dimungkinkan untuk mengimplementasikan sembarang fungsionalitas yang riil. Dapat berupa Low-fidelity atau high-fidelity. Memberi gambaran tentang keseluruhan produk atau bagian-bagiannya.

2.1.5 Prototype

Prototipe merupakan suatu metode dalam pengembangan sistem yang menggunakan pendekatan untuk membuat sesuatu program dengan cepat dan bertahap sehingga segera dapat dievaluasi oleh pemakai. Perancangan prototipe juga merupakan alah satu bagian dari proses UCD (*User Centered Design*). [5]

Keunggulan Prototipe :

1. Adanya komunikasi yang baik antara pembangun perangkat lunak dan pengguna.
2. Pembangun perangkat lunak dapat bekerja lebih baik dalam menentukan kebutuhan pengguna.
3. Lebih menghemat waktu dalam pengembangan sistem.

4. Penerapan menjadi lebih mudah karena pengguna mengetahui apa yang diharapkannya.

Kelemahan Prototipe :

1. Pengguna kadang tidak melihat atau menyadari bahwa perangkat lunak yang ada belum mencantumkan kualitas perangkat lunak secara keseluruhan dan juga belum memikirkan kemampuan pemeliharaan untuk jangka waktu lama.
2. Pembangun perangkat lunak biasanya ingin cepat menyelesaikan aplikasi. Sehingga menggunakan algoritma dan bahasa pemrograman yang sederhana untuk membuat *prototyping* lebih cepat selesai tanpa memikirkan lebih lanjut bahwa program tersebut hanya merupakan cetak biru sistem .
3. Hubungan pengguna dengan komputer yang disediakan mungkin tidak mencerminkan teknik perancangan yang baik .

2.1.5.1 Dimensi *Prototype*

1. Penyajian

1. Bagaimana desain dilukiskan atau diwakili.
2. Dapat berupa uraian tekstual atau dapat visual dan diagram.

2. Lingkup

Apakah hanya interface atau apakah mencakup komponen komputasi.

3. *Executability* (Dapat dijalankan)

1. Dapatkah prototipe tersebut dijalankan.
2. Jika dikodekan, akan ada periode saat prototipe tidak dapat dijalankan.

4. *Maturation* (Pematangan)

Apakah tahapan-tahapan produk ini mengikuti *revolusioner* dan *evolusioner*.

1. *Revolusioner* : mengganti yang lama.
2. *Evolusioner :* terus melakukan perubahan pada perancangan yang sebelumnya dengan mengembangkan tanpa mengurangi kinerja sistem yang lama.

2.1.5.2 Terminologi *Prototype*

1. Prototipe Horisontal

   Mengerjakan atau menunjukkan sebagian besar *interface*, tetapi tidak mendalam.

2. Prototipe Vertikal

   Lebih sedikit aspek atau fitur dari *interface* yang disimulasikan, tetapi dilaksanakan dengan rincian yang sangat baik.

3. *Early Prototyping* (prototipe cepat)
4. *Late Prototyping* (prototipe lambat)
5. *Low-fidelity Prototyping* (prototipe dengan tingkat ketepatan yang rendah)

   1. Gambaran cepat dari sistem final
   2. Mempunyai fungsi atau interaksi yang terbatas
   3. Lebih menggambarkan konsep , perancangan, alternativ, dan layout layar dibanding model interaksi pengguna dengan sistem.

4. Tidak untuk memperlihatkan secara rinci bagaimana operasi sistem aplikasi.
5. Digunakan pada awal siklus perancangan
6. Memperlihatkan konsep pendekatan secara umum tanpa harus membuang banyak tenaga, biaya dan waktu.

6. *Mid-fidelity prototyping* (prototipe dengan tingkat ketepatan sedang)
   - Form skematik.
   - Navigasi dan fungsi yang disimulasikan dan biasanya bersumber pada apa yang tampil pada layar .
7. *High-fidelity prototyping* (prototipe dengan tingkat ketepatan yang tinggi)

2.1.6 *Usability*

Definisi *usability* menurut Shackel adalah :

> *The capability of human functional terms to be used easily and efiectively by the specified range of users, given specific training and user support, to fulfil the specified range of tasks, within the specified range of environment scenario.*

Definisi tersebut masih kental dari sudut pandang *anthropometric* ataupun *biomekanis* . Artinya kenyamanan fisik lebih dititik beratkan sehingga Chapanis mendefinisikan *usability* lebih jauh sebagai :

The usability of computer is measured by how easily and efiectively the computer can be used by a specific set of users, given particular kinds of supports, to carry out.

Dari kedua definisi tersebut jelas suatu pengukuran usabilitas tidak dilakukan ketika pengguna baru memakai sistem. Tetapi dilakukan setelah pengguna melalui masa pelatihan dan mendapat dukungan yang cukup (manual dan sebagainya). Pada sudut pandangan teknologi informasi *usability* dapat diformulasikan sebagai berikut. [17]

Usability is emergent quality of an optimum design which is reected itthe efective and satisfying use of the IT

Menurut Jacob Nielsen karateristik *usability* terbagi menjadi 5 yaitu :

1. *Learnibility,* yaitu bagaimana waktu yang digunakan pengguna untuk dapat mengerti dan menggunakan aplikasi.
2. *Efficiency,* yaitu seberapa cepat pengguna dapat mengerti dan menggunakan *task.*
3. *Memorability,* yaitu bagaimana pengguna dapat mengingat mengingat cara kerja sistem setelah tidak berinteraksi.
4. *Erorrs,* yaitu berapa banyak eror yang dilakukan oleh pengguna.
5. *Satisfaction,* yaitu bagaimana kepuasan yang diterima oleh pengguna setelah berinteraksi dengan sistem yang digunakannya.

2.1.7 *User Interaction*

User interface yang baik harus menyatukan interaksi pengguna *(user interaction*) dan penyajian informasi *(information presentation*).

5 tipe utama interaksi untuk *user interaction*:

1. *Direct manipulation* (pengoperasian secara langsung) : interaksi langsung dengan objek pada layar. Misalnya *delete file* dengan memasukkannya ke trash.

   - Kelebihan: Waktu pembelajaran pengguna sangat singkat, *feedback* langsung diberikan pada tiap aksi sehingga kesalahan terdeteksi dan diperbaiki dengan cepat.

   - Kekurangan : *Interface* tipe ini rumit dan memerlukan banyak fasilitas pada sistem komputer, cocok untuk penggambaran secara visual untuk satu operasi atau objek.

2. Menu selection dengan pilihan berbentuk menu. Memilih perintah dari daftar yang disediakan. Misalnya saat klik kanan dan memilih aksi yang dikehendaki.

   - Kelebihan : pengguna tidak perlu mengingat nama perintah, minimal pengetikan , tingkat kesalahan rendah.

   - Kekurangan : Tidak ada logika AND atau OR, perlu ada struktur menu jika banyak pilihan, menu dianggap lambat oleh *expert user* dibanding *command language.*

3. *Form fill-in* (pengisian form) : Mengisi area-area pada *form.*

- Kelebihan : Masukan data yang sederhana dan mudah dipelajari.

- Kekurangan : Memerlukan banyak tempat di layar, harus menyesuaikan dengan form manual dan kebiasaan user.

4. *Command language* (perintah tertulis). Menuliskan perintah yang sudah ditentukan pada program contoh: *operating system.*

- Kelebihan : Perintah diketikan langsung pada sistem. Misal UNIX, DOS command, dapat diterapkan pada terminal yang murah, kombinasi perintah dapat dilakukan, misalnya *copy file* dan *rename* nama *file.*

- Kekurangan: Perintah harus dipelajari dan diingat cara penggunaannya, tidak cocok untuk pengguna biasa, kesalahan memakai perintah sering terjadi, perlu ada sistem pemulihan kesalahan dan kemampuan mengetik.

5. *Natural language* (perintah dengan bahasa alami). Menggunakan bahasa alami untuk mendapatkan hasil.

- Kelebihan: Perintah dalam bentuk bahasa alami dengan kosa kata yang terbatas (singkat) , misalnya kata kunci yang kita tentukan untuk dicari oleh *search engine*.

- Kekurangan: Tidak semua sistem cocok menggunakan ini jika digunakan maka akan memerlukan banyak pengetikan.

2.2 Media Sosial

Media sosial adalah sebuah media online dimana para penggunanya bisa dengan mudah berpartisipasi, berbagi, dan menciptakan isi meliputi blog, sosial *network* atau jejaring sosial, wiki, forum dan dunia virtual. Blog, jejaring sosial dan wiki mungkin merupakan bentuk media sosial yang paling umum digunakan oleh masyarakat di seluruh dunia.

Sementara jejaring sosial merupakan situs dimana setiap orang bisa membuat web page pribadi, kemudian terhubung dengan teman-teman untuk berbagi informasi dan berkomunikasi. Jejaring sosial terbesar antara lain *Facebook, Myspace,* dan *Twitter.* Jika media tradisional menggunakan media cetak dan media *broadcast*, maka media sosial menggunakan *internet*. Media sosial mengajak siapa saja yang tertarik untuk berpertisipasi dengan memberi kontribusi dan *feedback* secara terbuka, memberi komentar, serta membagi informasi dalam waktu yang cepat dan tak terbatas. [13]

Saat teknologi internet dan *mobile phone* makin maju maka media sosial pun ikut tumbuh dengan pesat. Kini untuk mengakses *facebook* atau *twitter* misalnya, bisa dilakukan dimana saja dan kapan saja hanya dengan menggunakan sebuah PC yang terkoneksi dengan *internet* ataupun menggunakan *mobile phone*. Demikian cepatnya orang bisa mengakses media sosial mengakibatkan terjadinya fenomena besar terhadap arus informasi tidak hanya di negara-negara maju, tetapi juga di Indonesia.

Karena kecepatannya media sosial juga mulai tampak menggantikan peranan media massa konvensional dalam menyebarkan berita-berita.

Pesatnya perkembangan media sosial kini dikarenakan semua orang seperti bisa memiliki media sendiri. Jika untuk memiliki media tradisional seperti televisi, radio, atau koran dibutuhkan modal yang besar dan tenaga kerja yang banyak, maka lain halnya dengan media. Seorang pengguna media sosial bisa mengakses menggunakan social media dengan jaringan internet bahkan yang aksesnya lambat sekalipun, tanpa biaya besar, tanpa alat mahal dan dilakukan sendiri tanpa karyawan. Kita sebagai pengguna social media dengan bebas bisa mengedit, menambahkan, memodifikasi baik tulisan, gambar, video, grafis, dan berbagai model content lainnya.

Intinya, menggunakan media sosial menjadikan kita sebagai diri sendiri. Selain kecepatan informasi yang bisa diakses dalam hitungan detik, menjadi diri sendiri dalam media sosial adalah alasan mengapa media sosial berkembang pesat.

Jika dalam kehidupan sehari-hari kita tidak bisa menyampaikan pendapat secara terbuka karena satu dan lain hal, maka tidak jika kita menggunakan media sosial. Kita bisa menulis apa saja yang kita mau atau kita bebas mengomentari apapun yang ditulis atau disajikan orang lain. Ini berarti komunikasi terjalin dua arah. Komunikasi ini kemudian menciptakan komunitas dengan cepat karena ada ketertarikan yang sama akan suatu hal.

2.3 Mikroblog

Mikroblog adalah suatu bentuk blog yang memungkinkan penggunanya untuk menulis teks pembaharuan singkat yang biasanya kurang dari 140 karakter dan mempublikasikannya, baik untuk dilihat semua orang atau kelompok terbatas yang dipilih oleh pengguna tersebut. Pesan-pesan ini dapat dikirim melalui berbagai cara yaitu melalui SMS (*Short Message Service*), pesan instan, surat elektronik, digital *audio* atau *web.* Yang membedakan mikroblog dengan blog lainnya adalah mikroblog memiliki ukuran yang lebih kecil dari ukuran blog sebenarnya. Akan tetapi tujuannya tetap sama yaitu pengguna menulis topik tertentu. [14]

Pengguna mikroblog juga dapat memberikan komentar kepada teks pembaharuan yang dibuat oleh pengguna lain dan begitu juga sebaliknya. Layanan ini dapat diunduh secara gratis. Tidak ada kriteria-kriteria khusus untuk menggunakan layanan ini, semua orang di seluruh dunia dapat melakukan aktivitas menulis teks pembaharuan singkat dengan fasilitas yang diberikan oleh mikroblog. Mikroblog telah mengalami perkembangan dan menyediakan banyak fasilitas yang bisa pengguna manfaatkan. Layanan ini memiliki banyak macam berikut kelebihan dan kekurangan yang bisa dijadikan pilihan bagi pengguna.

2.4 Mikroblog *Twitter*

Twitter adalah sebuah situs mikroblog dan situs web jejaring sosial yang memberikan fasilitas bagi pengguna untuk mengirimkan "pembaharuan" berupa tulisan teks dengan panjang maksimum 140 karakter melalui SMS, pengirim pesan instan, surat elektronik, atau aplikasi seperti *Twitterrific* dan *Twitbin*. *Twitter* didirikan pada Maret 2006 oleh perusahaan rintisan Obvious Corp. Kata *twitter* secara harfiah berarti 'berkicau'. Situs ini mempunyai konsep blog mikro dalam penggunaannya. Di Indonesia situs jejaring sosial ini mempunyai pengguna aktif yang cukup banyak. [7]

Logo yang digunakan oleh twitter adalah seperti pada gambar 2.3 .



Gambar 2.3 Logo Twitter

Kesuksesan *Twitter* membuat banyak situs lain meniru konsepnya, kadang menawarkan layanan spesifik lokal suatu negara atau menggabungkan dengan layanan lainnya.

Beberapa istilah yang dipergunakan di *Twitter* :

1. Tweet - sesuatu yang ditulis di status *Twitter*
2. @ - penanda yang ditulis didepan nama pengguna lain agar *tweet* yang dimasukkan diterima oleh pengguna tersebut

3. Reply – balasan terhadap *tweet*
4. Following - pengguna lain yang *tweet* nya akan selalu diterima
5. Follower - pengguna lain yang akan selalu menerima *tweet*
6. ReTweet (RT) adalah melakukan posting kembali *tweet* dari user lain
7. Direct Messages - pesan personal
8. Favorites - menyimpan *tweet* tertentu yang diinginkan
9. Hash Tags (#) - penanda yang ditulis di depan topik tertentu agar pengguna lain bisa mencari topik yang sejenis yang ditulis oleh orang lain juga
10. Trending Topics - topik yang sedang banyak dibicarakan banyak pengguna dalam suatu waktu yang bersamaan

2.4.1 Kegunaan *Mikroblog Twitter*

Beberapa studi, terutama oleh Harvard Business School dan Sysomos, telah mencoba menganalisis perilaku penggunaan layanan mikroblog. Kesimpulan dari penelitian yang mereka lakukan menunjukkan bahwa salah satu aktivitas masyarakat sekarang adalah menulis teks pembaharuan singkat. Dalam surveinya yang didasarkan pada lebih dari 11 juta pengguna,Sysomos menunjukkan bahwa 10% dari mereka memiliki akun Twitter.

Mikroblog memungkinkan pengguna menulis teks pembaharuan singkat tentang kehidupan pengguna di mana saja dan mengirimnya ke pengguna lain yang dapat dilakukan setiap waktu. Selain itu mikroblog dapat dijadikan juga sebagai media pengiriman berita. Perusahaan media seperti BBC, *The New York Times* dan Al Jazeera sedang mencoba mikroblog khususnya Twitter sebagai cara untuk mengirim dan menerima berita. Baru-baru ini mikroblog juga sering dijadikan media kampanye dan promosi bagi calon presiden atau kontestan sebuah ajang untuk mencari dukungan. Calon presiden atau kontestan tersebut memiliki profil pada mikroblognya dan pengguna lain dapat memeriksa teks pembaharuannya tiap waktu. Layanan mikroblog juga muncul sebagai sumber yang penting untuk mengetahui berita dan situasi terbaru yang terjadi di dunia. [12]

2.4.2 Kelebihan dan Kelemahan *Twitter*

Kelebihan *twitter :*

1. Mudah dinavigasi dan memperbarui, "*link to*" dan mempromosikan apapun.
2. Menjangkau lebih luas tidak hanya antara teman.
3. Satu *feed* untuk semua pengguna dan siapa pun dapat mengikuti orang lain kecuali diblokir.
4. Alat komunikasi yang cukup interaktif dan cepat tanggap.

Kelemahan *twitter* :

1. Kemampuan terbatas yaitu, menemukan orang-orang, mengirim pesan singkat, balasan langsung terhadap '*tweet*' yang diposting.
2. Dibatasi sampai 140 karakter per *update*.
3. Belum ada strategi keuangan yang mudah dan jelas.
4. Lebih menekankan pada perhitungan *follower.*

2.5 API (*Application Programming Interface*)

API adalah sekumpulan perintah, fungsi, dan protokol yang dapat digunakan saat membangun perangkat lunak untuk sistem operasi tertentu. API memungkinkan programmer untuk menggunakan fungsi standar untuk berinteraksi dengan sistem operasi. API atau *Application Programming Interface* juga merupakan suatu dokumentasi yang terdiri dari antar muka, fungsi, kelas, struktur untuk membangun sebuah perangkat lunak. Dengan adanya API, maka memudahkan seorang programmer untuk membongkar suatu *software* untuk kemudian dapat dikembangkan atau diintegrasikan dengan perangkat lunak yang lain. API dapat dikatakan sebagai penghubung suatu aplikasi dengan aplikasi lainnya. Suatu rutin standar yang memungkinkan *developer* menggunakan *system function*. Proses ini dikelola melalui *operating system*. Keunggulan dari API ini adalah memungkinkan suatu aplikasi dengan aplikasi lainnya untuk saling berinteraksi.[8]

Keuntungan dengan menggunakan API adalah sebagai berikut:

1) Portabilitas.

Developer yang menggunakan API dapat menjalankan programnya dalam sistem operasi mana saja asalkan sudah ter- *install* API tersebut.

2) Lebih Mudah Dimengerti

API menggunakan bahasa yang lebih terstruktur dan mudah dimengerti daripada bahasa *system call*. Hal ini sangat penting dalam hal *editing* dan pengembangan.

System call interface ini berfungsi sebagai penghubung antara API dan *system call* yang dimengerti oleh sistem operasi. *System call interface* ini akan menerjemahkan perintah dalam API dan kemudian akan memanggil *system calls* yang diperlukan. Untuk membuka suatu *file* tersebut *user* menggunakan program yang telah dibuat dengan menggunakan bantuan API, maka perintah dari *user* tersebut diterjemahkan dulu oleh program menjadi perintah *open()*. Perintah *open()* ini merupakan perintah dari API dan bukan perintah yang langsung dimengerti oleh kernel sistem operasi. Oleh karena itu, agar keinginan pengguna dapat dimengerti oleh sistem operasi, maka perintah *open()* tadi diterjemahkan ke dalam bentuk *system call* oleh *system call interface*. Implementasi perintah *open()* tadi bisa bermacam-macam tergantung dari sistem operasi yang digunakan.[15]

Cara menggunakan API :

1. Dilakukan dengan mengimpor package/kelas
2. Ada beberapa kelas bernama sama dipackage yang berbeda, yaitu :

   – import salah satu dan gunakan nama lengkap untuk yang lain

   – gunakan nama lengkap semua kelas

Kebanyakan Sistem Operasi seperti Windows, menyediakan fasilitas API sehingga programmer dapat melakukan aktivitas programming dengan lebih konsisten. Meskipun API didesain untuk programer, namun API juga baik untuk *user* karena setidaknya dapat menjamin bahwa program tersebut memiliki *interface* yang sama, sehingga lebih mudah untuk dipelajari.

2.6 *Twkl.in*

Twkl.in adalah salah satu servis internet yang menyediakan jasa penyingkatan URL. Penyingkatan URL (bahasa Inggris: *URL shortening*) adalah suatu teknik pada WWW yang membuat suatu laman situs dapat diakses dengan menggunakan suatu URL yang sangat pendek selain alamat aslinya. Gambar 2.4 adalah logo yang digunakan oleh twkl.in.



Gambar 2.4 Logo twkl.in

Tujuannya antara lain adalah untuk menghemat ruang, terutama dengan semakin populernya layanan mikroblog seperti *Twitter* yang memiliki keterbatasan jumlah karakter yang bisa dikirim. Contoh layanan penyingkatan URL lain yang terkenal antara lain adalah TinyURL, ow.ly dan yang versi buatan lokal seperti twkl.in.

2.7 *Img.ly*

Img.ly adalah salah satu layanan internet yang menyediakan jasa penyimpanan gambar (*Image hosting)* . Sebuah jasa penyimpanan gambar memungkinkan pengguna untuk mengunggah gambar ke sebuah website internet. Gambar tersebut lalu disimpan di server penyedia layanan dan selanjutnya bisa dilihat atau digunakan untuk keperluan yang beragam. Gambar 2.5 adalah logo yang digunakan oleh img.ly.



Gambar 2.5 Logo img.ly

2.8 *Google Translate*

Google Translate adalah salah satu layanan internet yang menyediakan jasa untuk menerjemahkan beberapa jenis bahasa . Situs yang diluncurkan pada 2007 ini memuat lebih dari 50 bahasa yang dipakai di beberapa negara di seluruh dunia.

Pengguna bisa melakukan penerjemahan misalnya dari Bahasa Indonesia ke bahasa lain, seperti Bahasa Jepang ataupun sebaliknya. Atau dari Bahasa Inggris ke bahasa asing lain, seperti Bahasa Arab, *Russian*, *French*, *Spanish* dan sebaliknya. Selain itu, juga terdapat fonetik atau cara baca bahasa dengan gaya penulisan tertentu seperti Bahasa Jepang atau Cina. Selanjutnya *google translate* menambah jumlah bahasa dan fonetik dari tahun ke tahun sejak diluncurkan pertama kali. Pada September 2010 lalu, fonetik yang ditambahkan adalah untuk Bahasa Latin, Arab, Rusia, Yunani, Persia, dan Bahasa Urdu. [9]

Gambar 2.6 adalah logo yang digunakan oleh *google translate.*



Gambar 2.6 Logo *Google Translate*

2.9 UML

Unified Modeling Language (UML) adalah keluarga notasi grafis yang didukung oleh meta-model tunggal , yang membantu pendeskripsian dan desain sistem perangkat lunak , khususnya sistem yang dibangun menggunakan pemrograman berbasis objek (OO). Definisi ini merupakan definisi yang sederhana. Pada kenyataannya , pendapat orang-orang tentang UML berbeda satu sama lain. Hal

ini dikarenakan oleh sejarahnya sendiri dan oleh perbedaan persepsi tentang apa yang membuat sebuah proses rancang-bangun perangkat lunak efektif.

2.9.1 Sejarah Unified Modeling Language (UML)

Tahun 1994, Grady Boch dan James Rumbaugh bergabung untuk menggunakan metode berorientasi objek. Ivan Jacobson bergabung pada tahun 1995, dan mereka bertiga fokus membuat suatu bahasa pemodelan objek standar sebagai ganti dari pendekatan atau metode objek standar. Berdasarkan kerja mereka dan hasil kerja lainnya pada industri, *Unified Modeling Language* (UML) versi 1.0 dirilis pada tahun 1997.

Unified Modeling Language (UML) tidak menentukan metode untuk sistem-sistem pengembangan, tetapi sudah diterima luas sebagai standar untuk pemodelan objek. *Object Management Gorup*/OMG, badan standar industri, mengadopsi UML pada bulan November 1997 dan terus bekerja sama untuk meningkatkannya berdasarkan kebutuhan industri. Pada saat ini, salah satu industri telah merilis sebuah sofware yang mendukung UML yaitu Visual Paradigm 6.4 Interprise edition. Berbagai industri juga bermunculan dan mendukung penggunaan UML dengan berbagai produk, diantaranya *Rational Rose, SmartDraw*, dan lain-lain.

2.9.2 Definisi *Unified Modeling Language* (UML)

Unified Modeling Language (UML) merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem software yang terkait dengan objek. Sementara menurut Henderi Unified Modeling Language (UML**)** adalah sebuah bahasa pemodelan yang telah menjadi standar dalam industri software untuk visualisasi, merancang, dan mendokumentasikan sistem perangkat lunak. *Unified Modeling Language* (UML) biasa digunakan untuk :

1. Menggambarkan batasan sistem dan fungsi-fungsi sistem secara umum, dibuat dengan *use case* dan *actor*
2. Menggambarkan kegiatan atau proses bisnis yang dilaksanakan secara umum, dibuat dengan *interaction diagrams*
3. Menggambarkan representasi struktur statik sebuah sistem dalam bentuk *class diagrams*
4. Membuat model *behavior* ”yang menggambarkan kebiasaan atau sifat sebuah sistem” dengan *state transition diagrams*
5. Menyatakan arsitektur implementasi fisik menggunakan *component and development diagrams*
6. Menyampaikan atau memperluas *fungsionality* dengan *stereotypes* (Ziga Turck, 2007)

UML merupakan salah satu alat bantu yang sangat handal dalam bidang pengembangan sistem berorientasi objek karena UML menyediakan bahasa

pemodelan visual yang memungkinkan pengembang sistem membuat *blue print* atas visinya dalam bentuk yang baku. UML berfungsi sebagai jembatan dalam mengkomunikasikan beberapa aspek dalam sistem melalui sejumlah elemen grafis yang bisa dikombinasikan menjadi diagram. UML mempunyai banyak diagram yang dapat mengakomodasi berbagai sudut pandang dari suatu perangkat lunak yang akan dibangun. Diagram-diagram tersebut digunakan untuk :

1. Mengkomunikasikan ide
2. Melahirkan ide-ide baru dan peluang-peluang baru
3. Menguji ide dan membuat prediksi
4. Memahami struktur dan relasi-relasinya

2.9.3 Diagram Dasar dalam *Unified Modeling Language* (UML)

Pemodelan menggunakan *Unified Modeling Language* merupakan metode pemodelan berorientasi objek dan berbasis visual. Karenanya pemodelan menggunakan UML merupakan pemodelan objek yang fokus pada pendefinisian struktur statis dan model sistem informasi yang dinamis daripada mendefinisikan data dan model proses yang tujuannya adalah pengembangan tradisional. UML menawarkan diagram yang dikelompokan menjadi lima perspektif berbeda untuk memodelkan suatu sistem. Berikut ini adalah penjelasan mengenai berbagai diagram UML serta tujuannya:

1. *Model Use Case Diagram*

Use Case Diagram secara grafis menggambarkan interaksi antara sistem, sistem eksternal, dan pengguna. Dengan kata lain *Use Case* diagram secara grafis mendeskripsikan siapa yang akan menggunakan sistem dan dalam cara apa pengguna mengharapkan interaksi dengan sistem itu. *Use Case* secara naratif digunakan untuk secara tekstual menggambarkan sekuensi langkah-langkah dari setiap interaksi.

2. Diagram Struktur Statis

UML menawarkan dua diagram untuk memodelkan struktur statis sistem informasi, yaitu:

*a. Class Diagram***:** menggambarkan struktur object sistem. Diagram ini menunjukan *class object* yang menyusun sistem dan juga hubungan antara *class object* tersebut.

*b. Object Diagram***:** serupa dengan *class* diagram, tetapi *object diagram* memodelkan *isntance object actual* dengan menunjukan nilai-nilai saat ini dari *atribut instance*. Object Diagram menyajikan *"snapshot/potret*" tentang objek sistem pada point waktu tertentu. Diagram ini tidak digunakan sesering *Class Diagram*, tetapi saat digunakan dapat membantu seorang *developer* memahami struktur sistem secara lebih baik.

3. Diagram Interaksi

Diagram interaksi memodelkan sebuah interaksi, terdiri dari satu set objek, hubungan-hubungannya, dan pesan yang terkirim di antara objek. Model diagram ini

memodelkan behavior (kelakuan) sistem yang dinamis dan UML memiliki dua diagram untuk tujuan ini, yaitu:

a. Diagram rangkaian/*Sequence* Diagram**:** secara grafis menggambarkan bagaimana objek berinteraksi dengan satu sama lain melalui pesan pada sekuensi sebuah *use case* atau operasi. Diagram ini mengilustrasikan bagaimana pesan terkirim dan diterima di antara objek dan dalam sekuensi atau timing apa.

*b. Diagram kolaborasi/Collaboration Diagram***:** serupa dengan diagram rangkaian/sekuensi, tetapi tidak fokus pada timing atau sekuensi pesan. Diagram ini justru menggambarkan interaksi (atau kolaborasi) antara objek dalam sebuah format jaringan.

Diagram rangkaian maupun diagram kolaborasi merupakan *isomorphic* artinya kita dapat mengubah dari satu diagram ke diagram lain.

4. Diagram *State/State Diagram*

UML memiliki sebuah diagram untuk memodelkan behavior objek khusus yang kompleks (*statecahrt*) dan sebuah diagram untuk memodelkan *behavior* dari sebuah *use case* atau sebuah metode, yaitu:

*a. Diagram statechart**:*** digunakan untuk memodelkan *behavior* objek khusus yang dinamis. Diagram ini mengilustrasikan siklus hidup objek-berbagai keadaan yang dapat diasumsikan oleh objek dan *event-event* (kejadian) yang menyebabkan objek beralih dari satu state ke state lain.

b. Diagram aktivitas/Activity Diagram: secara grafis digunakan untuk menggambarkan rangkaian aliran aktivitas baik proses bisnis maupun *use case*. *Activity* diagram dapat juga digunakan untuk memodelkan *action* yang akan dilakukan saat sebuah operasi dieksekusi, dan memodelkan hasil dari *action* tersebut.

5. Diagram Implementasi

Diagram implementasi juga memodelkan struktur sistem informasi, yaitu:

a. Diaram komponen/Component Diagram: digunakan untuk menggambarkan organisasi dan ketergantungan komponen-komponen *software* sistem. Komponen diagram dapat digunakan untuk menunjukan bagaimana kode pemrograman dibagi menjadi modul-modul (atau komponen).

b. Diagram penguraian/Deployment: digunakan untuk mendeskripsikan arsitektur fisik dalam istilah ”node” untuk hardware dan *software* dalam sistem. Diagram ini menggambarkan konfigurasi komponen-komponen software *real-time,* prosesor, dan peralatan yang membentuk arsitektur sistem.

2.10 *OAuth*

OAuth adalah suatu protokol terbuka yang memungkinkan pengguna untuk berbagi sumber pribadi mereka (mis. foto, video, daftar alamat) yang disimpan di suatu situs web dengan situs lain tanpa perlu menyerahkan nama pengguna dan kata

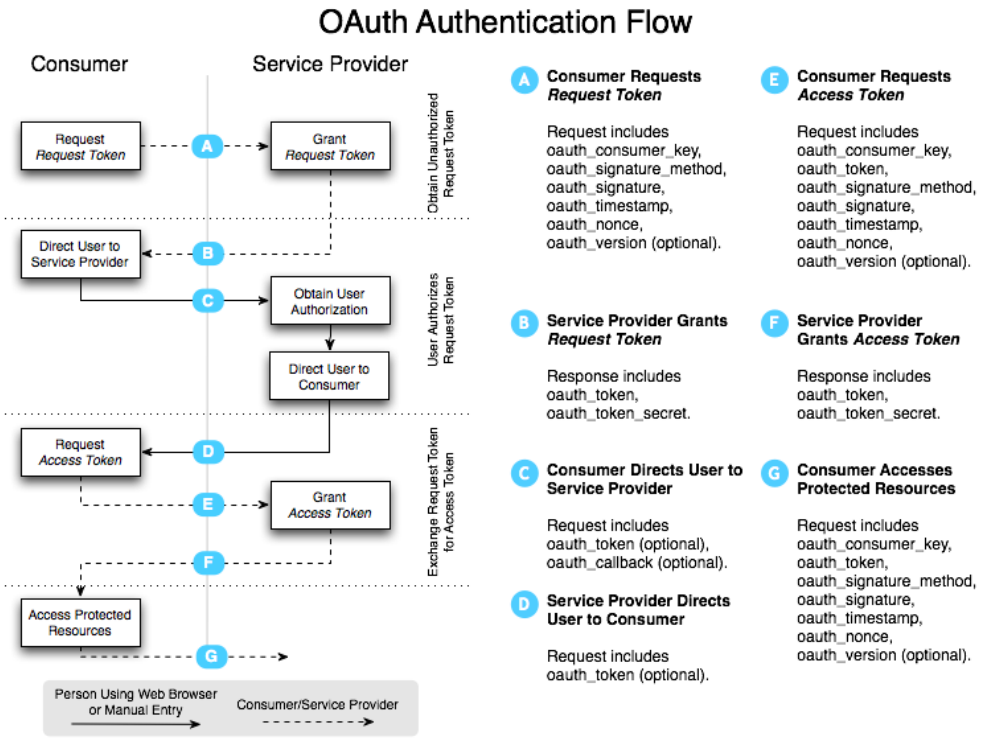
sandi mereka. Proses ini dilakukan dengan memberikan *token*, bukan nama pengguna dan kata sandi, untuk data mereka yang diinangi oleh suatu penyedia jasa tertentu. Setiap *token* memberikan akses untuk suatu situs spesifik terhadap suatu sumber spesifik selama durasi tertentu. Gambar 2.7 adalah merupakan logo dari Oauth.



Gambar 2.7 Logo OAuth

OAuth mengizinkan seorang mengguna untuk memberikan akses kepada situs pihak ketiga untuk mengakses informasi mereka yang disimpan di penyedia layanan lain tanpa harus membagi izin akses atau keseluruhan data mereka. [16]

Gambar 2.8 merupakan gambaran cara kerja *oauth.*



Gambar 2.8 *OAuth Authentication Flow*

2.10.1 Cara Kerja *OAuth* di *Twitter*

Ketika pengunjung datang ke halaman *login* , aplikasi mengirimkan request GET ke server *twitter* melalui sebuah set kode *request.* Kode *request* ini digunakan untuk verifikasi pengguna pada website *twitter*.

Pengguna lalu di dibawa ke halaman *twitter* untuk melakukan *allow* atau *deny* untuk akses aplikasi. Jika pengguna melakukan *allow* pada aplikasi , maka pengguna akan dibawa kembali ke alamat aplikasi. URL tempat pengguna akan dipindahkan telah

berisi *token OAuth* yang akan digunakan di langkah selanjutnya. Urutan cara kerja *OAuth* pada *twitter* adalah sebagai berikut:

1. Pengguna melakukan akses pada aplikasi , aplikasi menghubungi *twitter* untuk mendapatkan *one-time Request Token* seperti pada gambar 2.9 .



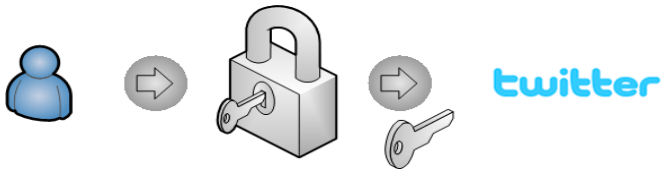
Gambar 2.9 Cara kerja *OAuth-01*

2. *Twitter* membalas dengan *Request Token* dan *Request Token Secret Key*. *Request Token* digunakan untuk membuat *authorization lin*k, dan *Request Token Secret* digunakan untuk memberitahu twitter bahwa user adalah yang asli melakukan *request* untuk *Request Token* seperti pada gambar 2.10 .



Gambar 2.10 Cara Kerja *OAuth-02*

3. User melakukan klik pada *authorization link* yang baru dibangkitkan, dibuat dengan *Request Token. Link* melakukan redirect ke *Twitter*, dan *API call* melakukan pass *Request Token Secret* seperti pada gambar 2.11 .



Gambar 2.11 Cara Kerja *OAuth-03*

4. *Twitter* memberi prompt kepada user untuk melakukan *allow* untuk akses aplikasi. Jika pengguna melakukan konfirmasi , *Twitter* mengirimkan pengguna kembali ke alamat aplikasi dengan *Access Token* dan *Access Token Secret key* . *Token key/pair* adalah *unique* untuk setiap pengguna dan tidak berubah. *Token key/pair* ini bisa disimpan di database untuk keperluan mendatang seperti pada gambar 2.12.



Gambar 2.12 Cara Kerja *OAuth-04*

2.10.2 Mendaftarkan Aplikasi di *Twitter*

Untuk memulai menggunakan API *twitter* , diharuskan untuk mendaftarkan terlebih dahulu dengan langkah sebagai berikut :

1. Melakukan akses ke http://dev.twitter.com/apps/new
2. Mengisi hal-hal yang diperlukan yaitu :
   a. Application name : nama aplikasi
   b. Description : deskripsi aplikasi
   c. Application Website : alamat website dari aplikasi
   d. Organization : organisasi pemilik aplikasi
   e. Application type : tipe aplikasi yaitu Client atau Browser
   f. Callback URL : alamat URL untuk di *redirect* setelah akses disetujui
   g. Default Access Type : berupa *"Read & Write"* atau *"Read Only"*
   h. Application Icon : Icon untuk aplikasi
   i. Captcha : form yang harus diisi untuk memastikan bahwa yang membuat adalah manusia
3. Setelah pembuatan/pendaftara aplikasi selesai maka akan diberikan *Consumer key* , *Consumer secret* , *API key* , *Access Token* dan *Access Token Secret*.

2.11 PHP

PHP adalah bahasa pemrograman *script* yang paling banyak dipakai saat ini. PHP banyak dipakai untuk memrogram situs web dinamis, walaupun tidak tertutup kemungkinan digunakan untuk pemakaian lain.

Contoh terkenal dari aplikasi PHP adalah phpBB dan MediaWiki (software di belakang Wikipedia). PHP juga dapat dilihat sebagai pilihan lain dari ASP.NET/C#/VB.NET Microsoft, ColdFusion Macromedia, JSP/Java Sun Microsystems , dan CGI/Perl. Contoh aplikasi lain yang lebih kompleks berupa CMS yang dibangun menggunakan PHP adalah Mambo, Joomla!, Postnuke, Xaraya, dan lain-lain.

2.11.1 Sejarah PHP

Pada awalnya PHP merupakan kependekan dari *Personal Home Page* (Situs Personal). PHP pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu PHP masih bernama FI (*Form Interpreted*), yang wujudnya berupa sekumpulan *script* yang digunakan untuk mengolah data *form* dari web. Logo dari PHP adalah seperti gambar 2.13 .



Gambar 2.13 Logo PHP

Selanjutnya Rasmus merilis kode sumber tersebut untuk umum dan menamakannya PHP/FI. Dengan perilisan kode sumber ini menjadi *open source*, maka banyak programmer yang tertarik untuk ikut mengembangkan PHP.

Pada November 1997, dirilis PHP/FI 2.0. Pada rilis ini interpreter PHP sudah diimplementasikan dalam program C. Dalam rilis ini disertakan juga modul-modul ekstensi yang meningkatkan kemampuan PHP/FI secara signifikan.

Pada tahun 1997, sebuah perusahaan bernama Zend menulis ulang interpreter PHP menjadi lebih bersih, lebih baik, dan lebih cepat. Kemudian pada Juni 1998, perusahaan tersebut merilis interpreter baru untuk PHP dan meresmikan rilis tersebut sebagai PHP 3.0 dan singkatan PHP dirubah menjadi akronim berulang *PHP: Hypertext Preprocessing*.

Pada pertengahan tahun 1999, Zend merilis interpreter PHP baru dan rilis tersebut dikenal dengan PHP 4.0. PHP 4.0 adalah versi PHP yang paling banyak dipakai pada awal abad ke-21. Versi ini banyak dipakai disebabkan kemampuannya untuk membangun aplikasi web kompleks tetapi tetap memiliki kecepatan dan stabilitas yang tinggi.

Pada Juni 2004, Zend merilis PHP 5.0. Dalam versi ini, inti dari interpreter PHP mengalami perubahan besar. Versi ini juga memasukkan model pemrograman berorientasi objek ke dalam PHP untuk menjawab perkembangan bahasa pemrograman ke arah paradigma berorientasi objek.

2.11.2 Kelebihan PHP

1. Bahasa pemrograman PHP adalah sebuah bahasa *script* yang tidak melakukan sebuah kompilasi dalam penggunaanya.

2. Web Server yang mendukung PHP dapat ditemukan dimana - mana dari mulai apache, IIS, Lighttpd, nginx, hingga Xitami dengan konfigurasi yang relatif mudah.

3. Dalam sisi pengembangan lebih mudah, karena banyaknya milis - milis dan developer yang siap membantu dalam pengembangan.

4. Dalam sisi pemahamanan, PHP adalah bahasa *scripting* yang paling mudah karena memiliki referensi yang banyak.

5. PHP adalah bahasa open source yang dapat digunakan di berbagai mesin (Linux, Unix, Macintosh, Windows) dan dapat dijalankan secara runtime melalui console serta juga dapat menjalankan perintah-perintah sistem. [6]

2.12 JavaScript

JavaScript adalah bahasa *script* yang digunakan pada halaman HTML. *JavaScript* adalah varian dari *java* yang sangat terkenal dalam lingkungan pemrograman *web*. *JavaScript* mempunyai keunggulan pada sisi *client*, maupun diakses lebih cepat dari *script server*. Tetapi kelemahannya pada *script* ini belum tentu mampu diproses oleh *browser client* tergantung kompetibilitas *browser*

tersebut. Cara untuk menjalankan j*avascript* hanya dengan *javascript-enabled* yaitu *browser* yang mampu menjalankan *javascript* seperti *Netscape Navigator* (versi 2.0 keatas) atau *internet ekplorer* (MSIE-versi 3.0 ke atas). [11]

Beberapa hal mengenai JavaScript :

1) *JavaScript* didesain untuk menambah interaktif suatu *web*.
2) *JavaScript* merupakan sebuah bahasa *scripting* dengan bahasa pemrograman yang ringan.
3) *JavaScript* biasanya disisipkan (*embedded*) dalam halaman HTML.
4) *JavaScript* adalah bahasa interpreter (yang berarti skrip dieksekusi tanpa proses kompilasi.

Ada beberapa cara untuk menuliskan JavaScript yaitu :

a) *Script* pada bagian *head*

Skrip ini dieksekusi ketika dipanggil (biasanya berbentuk *function*) atau dipanggil berdasarkan *event* tertentu. Peletakan *script* di *head* akan menjamin *script* di-*load* terlebih dahulu sebelum digunakan.

```
<html>
<head>
<script type=”text/javascript”>
Function message ()
{
```

```
Alert (“Alert ini dipanggil saat even dijalankan”);
}
<script>
</head>
</html>
```

b) *Script* pada *Body*

Skrip akan dieksekusi ketika halaman di-*load* sampai dibagian <body> dan ketika ditempatkan berarti antara isi dan *javacript* dijadikan satu bagian.

```
<html>
<head>
</head>
<body>
<script type=”text/javascript”>
Document.write(“Pesan ini akan tampil”);
</script>
</body>
</html>
```

c) External JavaScript

Untuk memudahkan menulis *script* yang berulang-ulang di setiap halaman maka *JavaScript* dapat ditulis di *file* secara *ekternal*. Jadi, antara HTML dan *JavaScript* dipisah, kemudian *file* tersebut dipanggil dari dokumen HTML. *File JavaScript* tersebut disimpan dengan ektensi .js.

```
<html>
<head>
</head>
<body>
<script src=”xxx.js”>
</script>
</body>
</html>
```