BAB II
LANDASAN TEORI

2.1 Tinjauan Pustaka

Naskah publikasi (Afriani & Ibrahim, 2015) dijelaskan bahwa penelitian dan pengembangan sistem yang dibuat dengan metode *Fast* (*Framework for the Aplication of System Thinking*) ini dimaksudkan untuk memudahkan penyampaian informasi di Fakultas Ilmu Komputer Unsri yang semula masih menggunakan sistem undangan dalam bentuk kertas. Selain itu sistem tersebut dibuat untuk efisiensi waktu dalam penyampaian informasi atau perintah karena informasi dikirimkan langsung ke nomor *handphone* bagian terkait.

Dalam sebuah skripsi (Santika, 2014) membahas dan merancang sebuah aplikasi berbasis web yang berfungsi untuk membantu sistem kerja pada perusahaan PCP, dimana aplikasi tersebut memanfaatkan kecanggihan teknologi informasi guna menunjang kegiatan operasional pengiriman barang yang ada di PCP cabang Tanjung Pinang. Dengan dibuatnya aplikasi ini diharapkan dapat meningkatkan efektifitas dan efisiensi yang akhirnya akan meningkatkan pelayanan ke pelanggan. Penelitian ini menggunakan metode *waterfall* sebagai metode pengembangan sistem.

Dalam skripsi lain (Ciputra, 2013) membahas dan membuat aplikasi yang mampu memberikan kemudahan bagi pelanggan untuk mengetahui keberadaan barang yang dikirim dan dapat membantu admin dalam menjalankan proses bisnisnya. Metode pengembangan sistem yang digunakan adalah metode *waterfall*.

Penelitian skripsi (Rahmadi, 2010) yang dilakukan di PD. Guna Pratama sebuah perusahaan yang bergerak di bidang jasa pengiriman barang khususnya pengiriman barang melalui jalur darat mengembangkan sebuah aplikasi SMS *gateway* yang memungkinkan pelanggan dan supir yang telah terdaftar dapat memberikan dan mengakses informasi dengan cepat dan mudah. Dengan adanya aplikasi ini diharapkan dapat meningkatkan pelayanan PD. Guna Pratama dan

proses pengolahan data pengiriman barang lebih efektif dan efisien. Pada skripsi ini Randi Rahmadi menggunakan metode RAD (*Rapid Application Development*) sebagai alur pengembangan sistem.

2.2 Kerangka Pemikiran

Kerangka pemikiran yang dijalankan dalam penelitian ini dijabarkan sebagai berikut:

1) Latar belakang masalah

   Tahap ini merupakan tahap awal dimana observasi permasalahan yang ada dilakukan kemudian berkembang dengan mengkaji latar belakang permasalahan tersebut.

2) Rumusan masalah

   Hasil dari pengkajian latar belakang permasalahan yang ada dapat dirumuskan sebuah permasalahan mendasar untuk kemudian diangkat menjadi tema penelitian.

3) Penguasaan dasar menggunakan PHP, HTML, CSS, MySQL dan SMS *Gateway*

   Tahap untuk mempelajari dasar-dasar dari PHP, HTML, CSS, MySQL serta SMS *Gateway* yang nantinya akan digunakan dalam pembuatan aplikasi.

4) Pengumpulan Data

   Pengumpulan data dilakukan dengan cara observasi dan wawancara langsung di lapangan serta mencari referensi dari buku, jurnal, dan sumber lainnya yang relevan.

5) Analisis dan perancangan aplikasi berbasis objek

6) Implementasi Sistem manajemen *report* pengiriman *order customer* dengan integrasi SMS *gateway* pada PT. Genusa Media Artha Sukoharjo

7) Pengujian Sistem

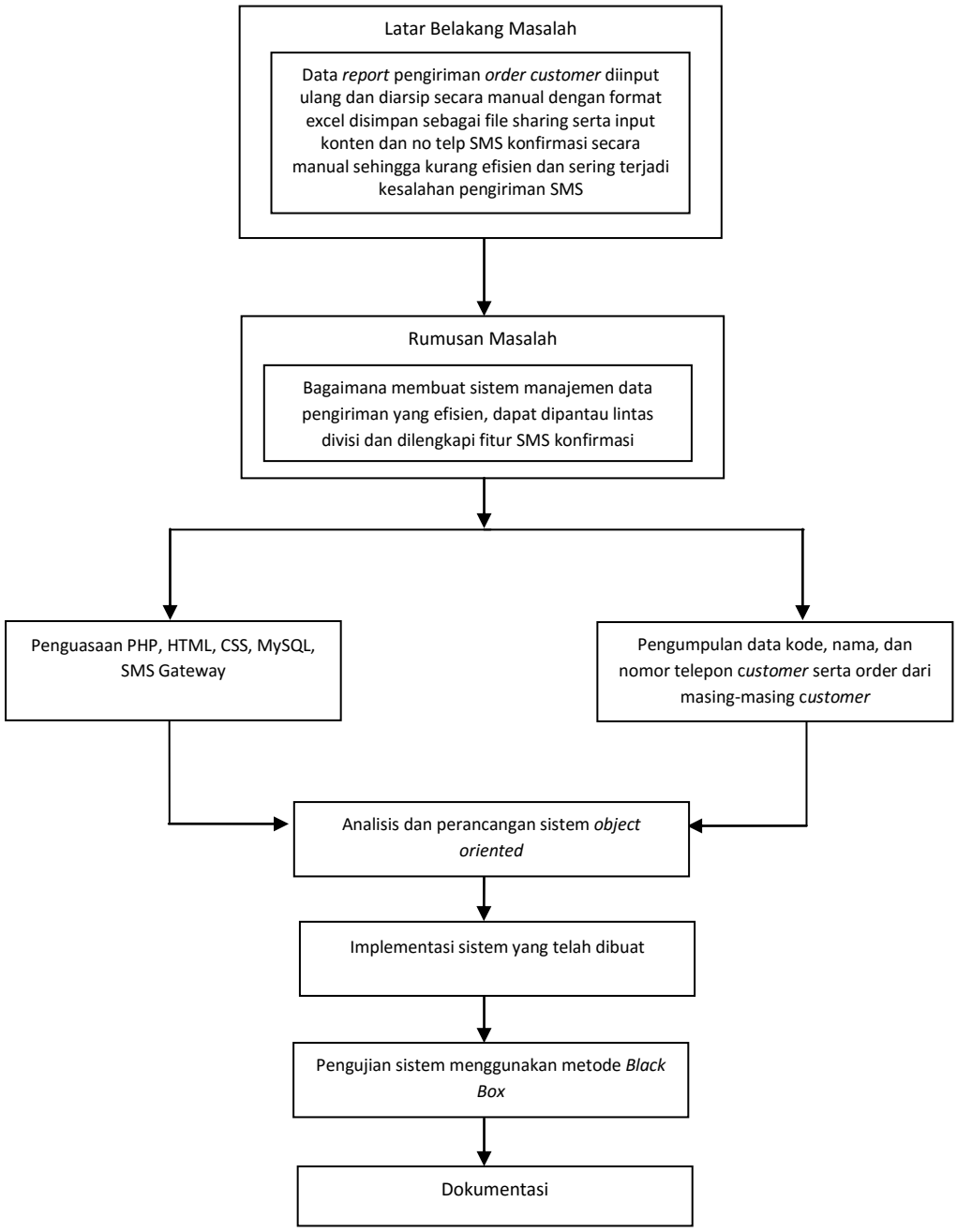
   Pengujian sistem akan dilakukan dengan metode *black box testing* serta metode angket untuk mengetahui tingkat kepuasan *user*.

8) Dokumentasi

   Tahap dokumentasi merupakan tahap terakhir dimana dilakukan

pendokumentasian terhadap penelitan yang telah dilakukan kemudian seluruh proses disusun menjadi laporan tugas akhir.

Berdasarkan penjabaran dari kerangka pemikiran di atas, maka dapat dibuatkan bagan alur seperti pada Gambar 2.1.



Gambar 2.1 Diagram Kerangka Pemikiran

2.3 Landasan Teori Pendukung

2.3.1 Sistem Informasi

Sistem didefinisikan sebagai sekumpulan prosedur yang saling berkaitan dan saling terhubung untuk melakukan usatu tugas bersama-sama. Secara garis besar, sebuah sistem terdiri dari tiga komponen utama. Ketiga komponen utama tersebut mencakup *software, hardware,* dan *brainware*. Ketiga komponen tersebut saling berkaitan satu sama lain (Pratama, 2014).

Masih dalam buku yang sama, I Putu Agus Eka Pratama menjelaskan bahwa informasi merupakan hasil pengolahan data dari satu atau berbagai sumber yang kemudian diolah sehingga memberikan nilai, arti, dan manfaat. Pada proses pengolahan data, untuk dapat menghasilkan informasi, juga dilakukan proses verifikasi secara akurat, spesifik, dan tepat waktu. Hal ini penting agar informasi dapat memberikan nilai dan pemahaman kepada pengguna. Pengguna dalam hal ini mencakup pembaca, pendengar, penonton, bergantung pada bagaimana cara pengguna tersebut menikmati sajian informasi dan melalui media apa informasi tersebut disajikan.

Berdasarkan uraian dari masing-masing definisi diatas, I Putu Agus Eka Pratama menjelaskan bahwa sistem informasi merupakan gabungan dari empat bagian utama. Keempat bagian utama tersebut mencakup perangkat lunak (*software*), perangkat keras (*hardware*), infrastruktur, dan sumber daya manusia (*brainware*) yang terlatih. Keempat bagian utama ini saling berkaitan untuk menciptakan sebuah sistem yang dapat mengolah data menjadi informasi yang bermanfaat. Didalamnya juga termasuk proses perencanaan, kontrol, koordinasi, dan pengambilan keputusan. Sehingga sebagai sebuah sistem yang mengolah data menjadi informasi yang akan disajikan dan digunakan oleh pengguna, maka sistem informasi merupakan sebuah sistem yang kompleks. Bukan hanya komputer saja yang bekerja (beserta *software* dan *hardware* yang dimiliki), namun juga manusia (dengan *brainware* yang dimiliki). Manusia atau pengguna dalam hal ini menggunakan seluruh ide, pemikiran, perhitungan, untuk dituangkan ke dalam sistem informasi yang digunakan.

2.3.2 Aplikasi

Aplikasi adalah penggunaan atau penerapan suatu konsep yang menjadi pokok pembahasan. Aplikasi dapat diartikan juga sebagai program komputer yang dibuat untuk menolong manusia dalam melaksanakan tugas tertentu (Rosa & Shalahuddin, 2011).

Aplikasi juga dapat didefinisikan sebagai program siap pakai yang dapat digunakan untuk menjalankan perintah-perintah dari pengguna aplikasi tersebut dengan tujuan mendapatkan hasil yang lebih akurat sesuai dengan tujuan dari pembuatan aplikasi tersebut.

Aplikasi komputer dapat dikategorikan menjadi beberapa kategori, yaitu :

1) *Enterprise*. Aplikasi jenis ini digunakan untuk organisasi yang cukup besar dengan maksud menghubungkan aliran data dan kebutuhan informasi antar bagian seperti IT *Helpdesk*, *Travel Management*, dan sebagainya,
2) *Enterprise – support*. Aplikasi jenis ini merupakan aplikasi pendukup dari *enterprise*, contohnya *Database Management*, *email server*, serta *networking system*,
3) *Individual worker*. Jenis aplikasi ini merupakan aplikasi yang biasa digunakan untuk mengolah atau mengedit data oleh penggunanya. Contoh dari jenis aplikasi ini antara lain Ms. Office, Photoshop, dan lain sebagainya,
4) Aplikasi akses konten. Merupakan aplikasi yang biasanya digunakan untuk mengakses konten tanpa kemampuan untuk mengolah atau mengedit data. Akses hanya terbatas pada melakukan kustomisasi. Contoh dari aplikasi ini antara lain *Media Player*, *Games*, dan Web Browser,
5) Aplikasi pendidikan merupakan aplikasi yang mengandung konten pembelajaran,
6) Aplikasi simulasi merupakan aplikasi yang memuat konten penelitian, pengembangan, dan lain sebagainya,
7) Aplikasi pengembangan media. Aplikasi ini berfungsi untuk mengolah atau mengembangkan media. Umumnya aplikasi ini digunakan untuk kepentingan komersial, hiburan, dan pendidikan,

8) Aplikasi mekanika dan produk. Aplikasi ini dibuat sebagai pelaksana atau pengolah data yang spesifik untuk kebutuhan tertentu seperti *Computer Aided Design* (CAD).

2.3.3 Manajemen *Report System*

Istilah manajemen berasal dari bahasa Latin *manus* yang berarti "tangan" (*Online Etymology*), dalam bahasa Italia *maneggiare* berarti "mengendalikan", kemudian bahasa Perancis *management* yang berarti "seni melaksanakan dan mengatur" (*Oxford English Dictionary*), sedangkan dalam bahasa Inggris istilah manajemen berasal dari kata *to manage* yang berarti mengatur. Pengaturan yang dilakukan melalui proses aktivitas dan diatur berdasarkan urutan dan fungsinya dianamakan *Manajemen*. Jadi manajemen itu merupakan suatu proses untuk mewujudkan keinginan yang hendak dicapai atau yang diinginkan oleh sebuah organisasi, baik organisasi bisnis, organisasi sosial, organisasi pemerintah, dan sebagainya (Usman, 2015).

Sedangkan definisi *report* sendiri adalah teks yang berisi gambaran sesuatu secara umum, dimana benda atau kejadian tersebut bersifat umum atau jamak. *Report* berfungsi untuk melaporkan hasil kegiatan operasional perusahaan selama satu periode atau pada tanggal tertentu. Tujuan pembuatan *report* adalah untuk memberikan informasi tentang kegiatan operasional dari perusahaan sebagai dasar untuk pengambilan keputusan manajemen (Ashari, 2005).

Dari definisi di atas dapat ditarik kesimpulan bahwa manajemen *report* merupakan sebuah sistem yang buat untuk mengendalikan, mengatur, atau memantau suatu kegiatan berdasarkan *report* masuk yang berisi hasil kegiatan operasional.

2.3.4 Pengiriman *Order Customer*

Dalam Kamus Besar Bahasa Indonesia (KBBI) *online* disebutkan bahwa *order* berarti perintah untuk melakukan sesuatu atau merupakan suatu bentuk pesanan. Secara lebih luas *order* dapat diartikan sebagai sebuah proses pembelian baik barang atau jasa yang dilakukan oleh konsumen kepada penjual sebelum konsumen mendapatkan barang. Kata *order* saat ini sering digunakan untuk

transaksi jual beli meskipun dulu masyarakat lebih sering menggunakan istilah pesanan daripada *order* ketika membeli barang.

Customer adalah orang yang membeli produk, baik itu perorangan, perusahaan, ataupun lembaga pemerintah. Dalam dunia bisnis, perusahaan menyadari bahwa memenangkan hati *customer*s sangatlah penting karena hal ini akan mempengaruhi kelangsungan hidup perusahaan di masa depan. Tujuan perusahaan bukan hanya semata-mata membuat *customer* puas, tetapi lebih dari itu perusahaan ingin membuat *customer* setia (Yunarto, 2006).

Pengiriman *order customer* sendiri merupakan salah satu bagian dari sirkulasi bisnis. Bagian ini merupakan bentuk pelayanan dari sebuah perusahaan atau penjual terhadap *order customer* yang masuk. Pelayanan yang baik, respon dan pengiriman yang cepat merupakan salah satu upaya untuk meningkatkan kepuasan serta kepercayaan *customer* sehingga dapat terbentuklah *customer* yang setia.

2.3.5 HTML

HTML merupakan singkatan dari *HyperText Markup Language*. HTML diartikan file teks murni yang dapat dibuat dengan media editor teks kompatibel untuk dikonversi menjadi sebuah program serta menggunakan kode-kode pemrograman (Buana, 2012).

HTML dibuat oleh Tim Berners-Lee pada tahun 1989 dan pertama kali dipopulerkan dengan *browser* Mosaic. Selama awal tahun 1990 HTML mengalami perkembangan yang sangat pesat. Tahun 1990 muncul HTML versi 1.0 dengan fitur *heading*, *paragraph*, *hypertext*, list, cetak tebal dan miring teks, serta *image wrappring*.

Pada tanggal 14 Januari 1996, HTML versi 2.0 resmi dirilis. Versi ini merupakan cikal bakal adanya *homepage* interaktif, karena dibenamkannya kemampuan menampilkan *form* pada dokumen sehingga dapat memasukkan nama, alamat, serta kritik maupun saran.

HTML versi 3.0 dirilis tanggal 18 Desember 1997. Versi ini telah mengadopsi fasitilas *table* dan disebut juga sebagai HTML+ namun segera

direlease HTML versi 3.2 karena adanya beberapa kasus yang timbul pada pengembang *browser*.

HTML versi 4.0 resmi dirilis pada 24 Desember 1999. HTML versi ini telah mengadopsi penambahan *link*, meta *imagemaps*, dan lain sebagainya sebagai penyempurna HTML versi 3.2. dan menjadi versi paling lama digunakan sebelum kemunculan HTML5.

HTML5 mulai mencuat pada April 2010 setelah CEO Apple Inc mengatakan bahwa dengan pengembangan HTML5, Adobe Flash sudah tidak dibutuhkan lagi untuk menyaksikan video atau menyaksikan konten apapun di web. Versi ini menambahkan fitur kanvas, video dan elemen audio, elemen konten yang lebih spesifik seperti *header*, *footer*, *navigations*, dan *section*. Dan bila dikombinasikan dengan CSS3 akan menjadi sebuah website yang dinamis.

HTML5 menawarkan kelebihan yang luar biasa dibandingkan generasi sebelumnya yakni HTML4 dan element yang digunakan pada HTML5 lebih memiliki arti atau lebih mudah dipahami saat dibaca atau menuliskannya.

2.3.6 CSS

CSS atau *Cascade Style Sheet* adalah sebuah fitur yang diperkenalkan sejak HTML versi 4.0 dan berfungsi untuk menangani masalah tampilan pada HTML seperti jenis, ukuran dan warna *font*, posisi teks, batas tulisan atau *margin*, warna *background*, dan sebagainya (Setiawan, 2008).

Konsep dari CSS mirip dengan template pada Microsoft Word. Dalam Microsoft Word perubahan penampilan pada dokumen Word dapat dilakukan dengan mengubah format pada *style* dokumen, begitu pula pada penampilan halaman sebuah website hanya dengan mengubah format tag HTML tertentu melalui CSS untuk selanjutnya menggantikan spesifikasi *default* dari *browser* untuk tag-tag tersebut. Beberapa kelebihan yang ditawarkan dari penggunaan CSS antara lain kemudahan pemformatan tambahan, kontrol yang lebih baik, serta perubahan yang lebih mudah.

Suatu ketika seorang web *developer* ingin merenovasi tampilan website yang telah dibuatnya, namun website tersebut terdiri dari halaman yang cukup

banyak cukup merepotkan jika harus melakukan editing per-*page*. Disinilah penggunaan CSS benar-benar membantu, web *developer* tersebut hanya perlu mengedit *syntax* yang terdapat dalam file .css sesuai dengan keinginan.

CSS1 dirilis pada tahun 1996 yang telah mendukung format, warna *font* teks, dan lain-lain. Kemudian pada tahun 1998 CSS2 dirilis yang didalmnya terdapat fungsi pengaturan letak elemen. Kemudian dilirislah CSS3 dan merupakan versi terakhir yang dirilis dengan penyempurnaan kemampuan dari versi sebelumnya. Banyak orang berpendapat kombinasi CSS3 dan HTML5 telah menggeser posisi JQuery dalam tampilan sebuah website.

2.3.7 PHP

PHP merupakan bahasa pemrograman web *server-side* yang bersifat *open source*. PHP adalah *script* yang terintegrasi dengan HTML dan berada pada server dan digunakan untuk membuat halaman website yang dinamis. PHP sudah menjadi bahasa *scripting* umum yang banyak digunakan di kalangan *developer* web. Mempunyai banyak kelebihan menjadi alasan utama kenapa PHP lebih dipilih sebagai basis umum dalam membuat sebuah aplikasi web (Hidayatullah & Jauhari, 2014).

PHP pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995 dan diberi nama FI (*Form Interpreted*) yang digunakan untuk mengelola *form* dari web. Karena berbasis *open source* pengembangan PHP cukup signifikan. Pada tahun 1997 dirilislah PHP 2.0 yang telah terintegrasi dengan bahasa pemrograman C serta dilengkapi dengan modulnya sehingga kualitas kerja PHP meningkat secara signifikan. Pada tahun ini juga sebuah perusahaan bernama Zend merilis ulang PHP dengan lebih bersih, baik, dan cepat. Tahun 1998 muncullah PHP 3.0.

PHP 4.0 dirilis tahun 1999 dan merupakan versi paling banyak digunakan pada awal abad 21 karena mampu membangun web komplek dengan stabilitas kecepatan yang tinggi. Kemudian pada tahun 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 kearah paradigma berorientasi objek.

Beberapa kelebihan dari PHP adalah sebagai berikut :

1) Bersifat *open source* yang artinya dapat digunakan dan didistribusikan oleh siapapun secara gratis,
2) Bahasa pemrograman PHP lebih kompatibel yakni dapat dijalankan di semua *platform* OS baik *desktop* maupun *mobile* yang mempunyai web *browser*,
3) Mendukung lebih banyak paket *database* seperti MySQL, Oracle, PostgrSQL, dan lain sebagainya,
4) Aplikasi PHP lebih cepat dibandingkan dengan ASP maupun Java,
5) Pengembangan aplikasi PHP lebih mudah karena banyak dokumentasi, referensi serta sifatnya yang *open source* menjadi PHP memiliki cukup banyak *developer*.

2.3.8 *Database* MySQL

Database adalah sekumpulan *file* data yang satu sama lainnya saling berhubungan yang diorganisasikan sedemikian rupa sehingga memudahkan untuk mendapatkan dan memproses data tersebut. Lingkungan sistem *database* menekankan pada data yang tidak tergantung (*independent*) pada aplikasi yang akan menggunakan data tersebut. Data adalah kumpulan fakta dasar (mentah) yang terpisah. Data menggambarkan suatu organisasi (Masrur, 2016).

Dalam buku lain dijelaskan bahwa *database* adalah suatu aplikasi yang menyimpan sekumpulan data. Setiap *database* mempunyai API tertentu untuk membuat, mengakses, mengatur, mencari, dan menyalin data yang ada di dalamnya (Enterprise, 2014).

MySQL merupakan salah satu jenis *database*. Dalam sebuah buku dijelaskan bahwa MySQL sebagai salah satu *Relational Database Management* yang bersifat *Open Source*. Struktur *database* disimpan dalam tabel-tabel yang saling berelasi. Karena sifat *open source*, MySQL dapat dipergunakan dan didistribusikan baik untuk kepentingan individu maupun *corporate* secara gratis tanpa memerlukan lisensi dari pembuatnya. MySQL dapat dijalankan dalam

berbagai *platform* sistem operasi seperti Windows, Linux, Unix, Sun OS, dan lain-lain. *Source* dan dokumentasi lengkap dapat diperoleh melalui situs *www.mysql.com* (Masrur, 2016).

2.3.9 UML (*Unified Modelling Language*) Diagram

Pada umumnya metode-metode yang ditujukan untuk pembangunan aplikasi berorientasi objek menggunakan UML untuk memodelkan berbagai artefak dari pengkat lunak. Dalam sebuah buku dijelaskan bahwa UML adalah sekumpulan simbol dan diagram untuk memodelkan *software*. Dengan menggunakan UML, desain *software* dapat diwujudkan dalam bentuk simbol dan diagram. Desain dalam bentuk simbol dan diagram, kemudian dapat diterjemahkan menjadi kode program. Telah tersedia *tools* yang dapat membuat kode program berdasarkan UML *Class* Diagram. Implementasi kode program dari UML dapat menggunakan bahasa pemrograman apa saja dengan syarat bahasa pemrograman tersebut mendukung pemrograman berorientasi objek (OOP). Pada dasarnya UML dibuat untuk mempermudah dan memvisualisasikan sistem yang seaman mungkin dengan cara yang sesederhana mungkin (Aziz, 2005).

UML juga dapat diartikan sbagai sebuah bahasa yang menentukan, visualisasi, konstruksi dan mendokumentasikan *artifact* (bagian dari informasi yang digunakan maupun dihasilkan dalam suatu proses pembuatan perangkat lunak. *Artifact* dapat berupa model, deskripsi, atau perangkat lunak) dari sistem perangkat lunak, seperti pada pemodelan bisnis atau sistem non perangkat lunak lainnya.

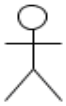
UML merupakan bahasa standar untuk penulisan *blueprint software* yang digunakan untuk visualisasi, spesifikasi, pembentukan dan pendokumentasian alat-alat dari sistem perangkat lunak.

2.3.10.1 *Use Case* Diagram

Use case diagram menampilkan sekumpulan *use case* dan aktor (pelaku) serta hubungan diantara *use case* dan aktor tersebut. *Use case* diagram digunakan untuk penggambaran *use case* static dari suatu sistem. *Use case* merupakan komponen wajib dalam program, dengan adanya *use case* pengguna dapat melihat

gambaran dari kegunaan aplikasi tersebut. Adapun simbol dan fungsi simbol yang digunakan dalam *use case* diagram (Rosa A. S. & Shalahuddin, 2013) dapat dilihat pada Tabel 2.1.

Tabel 2.1 Simbol *Use Case* Diagram

GAMBAR	NAMA	KETERANGAN
	*Actor*	Orang, proses atau sistem lain yang berinteraksi dengan sistem inforamsi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri.
	*Use Case*	Fungsional yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.
	Association	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
<<include>  user 	*Include*	Relasi *use case* tambahan ke sebuah *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini.

2.3.10.2 *Class* Diagram

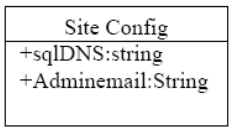
Class diagram digunakan untuk menampilkan kelas-kelas dan paket-paket di dalam sistem. *Class* diagram memberikan gambaran untuk sistem secara statis dan relasi antar kelas. Umumnya, dibuat beberapa *class* diagram untuk sistem tunggal. Beberapa diagram akan menampilkan subset dari kelas-kelas dan relasinya. Dapat juga dibuat beberapa diagram sesuai dengan yang diinginkan untuk mendapatkan gambaran lengkap terhadap sistem yang dibangun.

Class diagram adalah alat perancangan terbaik untuk tim pengembang. Diagram tersebut membantu pengembang mendapatkan struktur sistem sebelum

kode ditulis, dan membantu untuk memastikan bahwa sistem adalah desain terbaik.

Class diagram juga dapat diartikan sebagai kumpulan objek-objek dengan data yang mempunyai struktur umum, *behavior* umum, relasi umum, dan semantic atau kata yang umum. Sebuah *class* digambarkan seperti sebuah bujur sangkar dengan tiga bagian ruangan. *Class* sebaiknya diberi nama menggunakan kata benda sesuai dengan domain, bagian atau kelompoknya. Simbol *class diagram* (Rosa A. S. & Shalahuddin, 2013) dapat dilihat pada Tabel 2.2.

Tabel 2.2 Simbol *Class* Diagram

GAMBAR	NAMA	KETERANGAN
	*Class*	*Class* adalah blok - blok pembangun pada pemrograman berorientasi obyek.
	Association	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*.

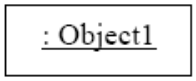
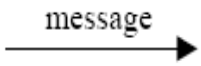
2.3.10.3 *Sequence* Digram

Sequence diagram merupakan salah satu jenis diagram pada UML yang menjelaskan interaksi objek yang berdasarkan urutan waktu. *Sequence* diagram juga dapat menggambarkan urutan atau tahapan yang harus dilakukan untuk dapat menghasilkan sesuatu seperti pada *use case* diagram.

Sequence diagram digunakan untuk menggambarkan perilaku pada sebuah skenario. Diagram jenis ini memberikan kejelasan sejumlah objek dan pesan-pesan yang diletakkan diantaranya di dalam sebuah *use case*. Komponen utamanya adalah objek yang digambarkan dengan kota segi empat, *message* yang digambarkan dengan garis penuh, dan waktu yang ditunjukkan dengan *progress vertical*.

Secara mudahnya *sequence* diagram adalah gambaran tahap demi tahap yang seharusnya dilakukan untuk menghasilkan sesuatu sesuai dengan *use case* diagram. Simbol *sequence* diagram (Rosa A. S. & Shalahuddin, 2013) tersaji pada Tabel 2.3.

Tabel 2.3 Simbol *Sequence* Diagram

GAMBAR	NAMA	KETERANGAN
	*Actor*	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri.
	*Object*	Merupakan *instance* dari sebuah class dan dituliskan tersusun secara horizontal.
	Lifeline	Menyatakan kehidupan suatu objek.
	*Activation*	Menyatakan objek dalam keadaan aktif dan berinteraksi pesan. Semua yang berhubungan dengan waktu aktif ini adalah sebuah tahapan yang dilakukan didalamnya.
	*Message*	Message mengindikasikan komunikasi antara object-object.

2.3.10.4 *Activity* Diagram

Activity diagram atau diagram aktivitas merupakan salah satu jenis diagram pada UML yang dapat memodelkan proses-proses apa saja yang terjadi pada sistem. *Activity* diagram juga dapat diartikan sebagai sebuah diagram yang menggambarkan alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi dan bagaimana akan berakhir. *Activity* diagram juga dapat menggambarkan proses paralel yang

mungkin terjadi pada beberapa eksekusi. Simbol dan fungsi simbol yang digunakan dalam *activity* diagram (Rosa A. S. & Shalahuddin, 2013)dapat dilihat pada Tabel 2.4.

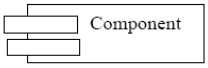
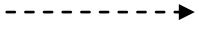
Tabel 2.4 Simbol *Activity* Diagram

GAMBAR	NAMA	KETERANGAN
	Status Awal	Status awal aktivitas sistem
	Aktivitas	Aktivitas yang dilakukan sistem biasanya diawali dengan kata kerja.
	Status Akhir	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.

2.3.10.5 *Component* Diagram

Component diagram dibuat untuk menunjukkan organisasi dan ketergantungan diantara kumpulan komponen dalam sebuah sistem. *Component* diagram fokus pada komponen sistem yang dibutuhkan dan ada di dalam sistem. symbol *component* diagram (Rosa A. S. & Shalahuddin, 2013) tersaji pada Tabel 2.5.

Tabel 2.5 Simbol *Component* Diagram

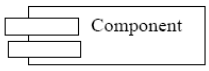
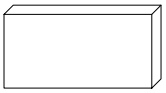
GAMBAR	NAMA	KETERANGAN
	*Component*	Sebuah komponen melambangkan sebuah entitas software dalam sebuah sistem.
	*Dependency*	Sebuah Dependency digunakan untuk menotasikan relasi antara dua komponen.

2.3.10.6 *Deployment* Diagram

Deployment diagram merupakan salah satu diagram pada UML yang menunjukkan tata letak atau sistem secara fisik, dapat juga dikatakan untuk

menampilkan bagian-bagian *software* yang terdapat *hardware* dan digunakan untuk menerapkan suatu sistem dan hubungan antara komponen *hardware*. *Deployment* diagam juga dapat diartikan sebagai salah satu jenis UML yang digunakan untuk memvisualisasikan, menspesifikan, dan mendokumentasikan proses yang terjadi pada suatu sistem perangkat lunak berbasis *object oriented* yang akan dibagun. Symbol *deployment* diagram (Rosa A. S. & Shalahuddin, 2013) tersaji pada Tabel 2.6.

Tabel 2.6 Simbol *Deployment* Diagram

GAMBAR	NAMA	KETERANGAN
	*Component*	Pada deployment diagram, komponen-komponen yang ada diletakkan didalam node untuk memastikan keberadaan posisi mereka
	*Node*	Menggambarkan bagian-bagian *hardware* dalam sebuah sistem.
	Association	Menghubungkan dua *node* yang mengindikasikan jalur komunikasi antara elemen-emelen dengan *hardware*

2.3.10 SMS *Gateway*

SMS, salah satu teknologi yang meskipun sudah cukup lama diluncurkan namun fitur ini belum bisa sepenuhnya tergeser oleh aplikasi-aplikasi *messaging* dengan fitur serupa atau bahkan lebih unggul. Kenyataannya teknologi ini masih menjadi alternatif yang sangat bisa diandalkan ketika aplikasi-aplikasi *social messaging* yang ada saat ini tidak dapat digunakan karena adanya gangguan koneksi internet. Koneksi internet bisa dikatakan sebagai nyawa bagi aplikasi-aplikasi *social messaging* saat ini, karena itu bila tanpa adanya koneksi internet aplikasi-aplikasi tersebut sama sekali tidak dapat digunakan. Selain itu fitur SMS masih menjadi andalan untuk berkomunikasi bagi orang-orang yang awam

terhadap teknologi berbasis intenet misalnya para orang tua karena memang cara penggunaannya yang relatif sederhana. Selain itu fitur SMS dengan sistem *broadcast* dengan memanfaatkan SMS *gateway* juga merupakan salah satu media promosi yang masih cukup dinikmati oleh *corporate*. Sebagai contoh adalah ketika seseorang mendaftarkan nomor telepon yang digunakan disebuah perusahaan asuransi misalnya, para member dari perusahaan asuransi tersebut akan menerima SMS informasi apabila ada promo atau penawaran baru dari perusahaan asuransi tersebut.

Definisi SMS *gateway* sendiri dalam sebuah buku disebutkan sebagai suatu teknologi pengolahan dilakukan secara komputerisasi dan memanfaatkan layanan SMS itu sendiri untuk berbagai keperluan serta tujuannya masing-masing. Dewasa ini, masyarakat lebih mengartikan SMS *gateway* sebagai jembatan komunikasi yang menghubungkan perangkat keras (dalam hal ini ponsel) dengan perangkat komputer yang membuat aktivitas SMS menjadi lebih mudah dan menyenangkan (Maulana, 2015).

2.3.11.1 Gammu

Gammu merupakan salah satu *tools* untuk mengembangkan aplikasi SMS *Gateway* yang cukup mudah diimplementasikan dan bersifat *free download*. Dalam website resminya dijelaskan bahwa Gammu adalah sebuah nama *project* serta perangkat lunak yang digunakan untuk membangun aplikasi, *script* dan *drivers* yang dapat digunakan untuk semua fungsi telepon seluler maupun alat sejenisnya. Kelebihan Gammu antara lain :

a) *Compatible* dengan Windows maupun Linux,
b) Menggunakan *database* MySQL,
c) Dapat membaca, menghapus dan mengirim SMS,
d) Dapat menggunakan *interface* web-*based*.

2.3.11 *Black Box Testing*

Pengembangan perangkat lunak mengikuti siklus hidup tertentu yang dimulai dari menentukan solusi untuk masalah yang ditemukan dan mengimplementasikannya. Pengujian sistem perangkat lunak atau *software testing*

adalah bagian dari siklus hidup tersebut yang melibatkan verifikasi apakah setiap unit yang dikembangkan telah memenuhi kebutuhan sistem yang dikembangkan telah memenuhi kebutuhan sistem yang didefinisikan pada tahapan sebelumnya. Pengujian sistem merupakan proses mengeksekusi sistem perangkat lunak untuk menentukan apakah sistem perangkat lunak tersebut cocok dengan spesifikasi sistem dan berjalan sesuai dengan lingkungan yang diinginkan, pengujian sistem sering diasosiasikan dengan pencarian *bug*, ketidaksempurnaan program, kesalahan pada baris program yang menyebabkan kegagalan pada eksekusi sisem perangkat lunak

Black box testing merupakan salah satu metode dalam unit *testing*. *Black box testing* terfokus pada apakah unit program memenuhi kebutuhan atau *requirement* yang disebutkan dalam spesifikasi pada *black box testing*, cara pengujian hanya dilakukan dengan menjalankan atau mengeksekusi unit atau modul, kemudian diamati apakah hasil dari unit tersebut sesuai dengan proses bisnis yang diinginkan (Fatta, 2007). Kelebihan dari *black box testing* antara lain :

a) Spesifikasi program dapat ditentukan di awal,
b) Dapat digunakan untuk menilai konsistensi program,
c) Testing dilakukan berdasarkan spesifikasi,
d) Tidak perlu melihat kode program secara detail.