**BAB II**

**LANDASAN TEORI**

**2.1. Tinjaun Pustaka**

**A. Konsep Dasar Sistem**

Menurut McLeod dalam Yakub (2012:1) mengemukakan bahwa “sistem adalah sekelompok elemen-elemen yang terintegrasi dengan tujuan yang sama untuk mencapai tujuan”.

**B. Model Pengembangan Sistem *waterfall***

Model pengembangan perangkat lunak dikenal dengan nama Siklus Hidup Pengembangan Sistem (SDLC). *WaterFall Model* menggambarkan sekuensial linier untuk rekayasa perangkat lunak, model ini muncul pertama kali tahun 1970 sehingga sering dianggap kuno, tetapi model ini yang paling banyak digunakan dalam rekayasa perangkat lunak.

Model SDLC air terjun *(waterfall)* sering juga disebut model sekuensial linier atau alur hidup klasik. Model air terjun menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut dimulai dari analisis, desain, pengodean, pengujian, dan tahap pendukung. Menurut Rosa dan Shalahuddin (2013:28-30) model waterfall adalah sebagai berikut:

**Sistem/Rekayasa informasi**

Analisis

Pengujian

Pengodean

Desain

Sumber : Rosa dan Shalahudin (2013:28-30)

Gambar II.1

Ilustrasi Model *Waterfall*

1. Analisis kebutuhan perangkat lunak

Proses pengumpulan kebutuhan dilakukan secara intensif untuk menspesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh *user*. Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu untuk didokumentasikan.

1. Desain

Desain perangkat lunak adalah proses multi langkah yang focus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antarmuka, dan prosedur pengkodean. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan ke representasi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya. Desain perangkat lunak yang dihasilkan pada tahap ini juga perlu didokumentasikan.

1. Pembuatan kode program

Desain harus ditranslasikan ke dalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan desain yang telah dibuat pada tahap desain.

1. Pengujian

Pengujian fokus pada perangkat lunak dari segi lojik dan fungsional dan memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalisir kesalahan (*error*) dan memastikan keluaran yang dihasilkan sesuai dengan yang diinginkan.

1. Pendukung (*support*) atau pemeliharaan (*maintenance*)

Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikirimkan ke *user*. Perubahan bisa terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak harus beradaptasi dengan lingkungan baru. Tahap pendukung atau pemeliharaan dapat mengulangi proses pengembangan mulai dari analisis spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru.

Model air terjun sangat cocok digunakan, kebutuhan pelanggan sudah sangat dipahami dan kemungkinan terjadinya perubahan kebutuhan selama pengembangan perangkat lunak kecil. Hal positif dari model air terjun adalah struktur tahap pengembangan sistem jelas, dokumentasi dihasilkan di setiap tahap pengembangan, dan sebuah tahap dijalankan setelah tahap sebelumnya selesai dijalankan (tidak ada tumpang tindih pelaksanaan tahap).

**C. *UML* (*Unified Modeling Language*)**

UML adalah “salah satu standar bahasa yang banyak di gunakan di dunia industri untuk mendefinisikan requirement, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek. Menurut Sukamto dan Shalahudin (2013:133).”

Dalam merancang program dibutuhkan peralatan pendukung yang dapat digunakan untuk membantu meringankan proses perancangan program aplikasi. *UML* yang berarti bahasa pemodelan standar merupakan alat komunikasi yang konsisten dalam mensuport para pengembang sistem yang memungkinkan para anggota *team* untuk bekerja sama dengan model yang sama dalam merancang suatu sistem yang akan dibuat, model yang digunakan untuk membangun dalam pengembangan dalam pemrograman terstruktur adalah:

1. *Use Case Diagram*

*Use case* atau diagram use case merupakan permodelan untuk kelakuan *(behavior)* sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar*, use case* digunakan untuk mengetahui fungsi apa saja yang ada didalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Menurut Sukamto dan Shalahudin (2013:155).

1. *Activity Diagram*

Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu di perhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor,jadi aktivitas yang dapat dilakukan oleh sistem. Menurut Sukamto dan Shalahudin (2013:161).

1. *Component Diagram*

Menurut Sukamto dan Shalahudin (2013:149) mengemukakan bahwa “Diagram komponen atau *component diagram* dibuat untuk menunjukan organisasi dan ketergantungan diantara kumpulan komponen dalam sebuah sistem. Diagram komponen fokus pada komponen sistem yang dibutuhkan dan ada didalam sistem*”*.

1. *Deployment Diagram*

Diagram *deployment* atau *deployment diagram* menunjukan konfigurasi komponen dalam proses eksekusi aplikasi. *Diagram deployment* juga dapat di gunakan untuk memodelkan sistem tambahan (*embedded system*),sistem *client*(*server*),sistem terdistribusi murni dan rekayasa ulang aplikasi. Menurut Rosa dan Shalahudin (2013:154).

**D. Konsep Pemrograman**

Menurut Sukamto dan Shalahuddin (2013:67-68) mengemukakan bahwa “pemrograman terstruktur adalah konsep atau paradigma atau sudut pandang pemrograman yang membagi-bagi program berdasarkan fungsi-fungsi atau prosedur-prosedur yang dibutuhkan program komputer”. Modul-modul (pembagian program biasanya dibuat dengan mengelompokkan fungsi-fungsi dan prosedur-prosedur yang diperlukan sebuah proses tertentu. Adapun konsep pemrograman terstruktur yang akan dibahas yaitu :

1. Modular

Modular *programming* (pemrograman modular) merupakan suatu teknik yang digunakan untuk menulis program yang berukuran besar”. Program dibagi menjadi menjadi beberapa bagian kecil. Tiap bagian disebut modul, yang melakukan suatu tugas tertentu. Umumnya setiap modul program terdiri dari sekumpulan pernyataan yang memiliki fungsi/keguanaan tertentu di dalam program. Setiap modul tersebut diberi nama sehingga untuk menyatakannya cukup dengan menyebut namanya. Program yang didefinisikan dengan baik akan mudah dibaca dan dimengerti oleh pemakai serta menjadi lebih efisien karena modul yang sama mungkin dipakai pada beberapa tahapan program.

1. *Top Down*

Pendekatan top-down ini sangat berguna dalam perencanaan pemrograman modular. Dalam pemrograman top-down yang pertama kita definisikan adalah modul utama. Modul utama yang dimaksud adalah modul yang pertama kali dijalankan, yang memanggil modul lain dan juga modul yang mengakhiri program. Keuntungan dari teknik top-down adalah bahwa setiap tahap pemrograman yang ada menjadi sederhana karena setiap tingkat mengabaikan detail dari tingkat yang lebih rendah.

1. *Bottom Up*

Bila ada masalah yang kompleks, maka pemecahan masalah dilakukan dengan menggabungkan prosedur-prosedur yang ada menjadi satu kesatuan program guna menyelesaikan masalah tersebut. Pendekatan ini bertitik tolak pada tujuan program secara umum/menyeluruh dan bukan bagaimana cara mencapainya.

**E. *Entity Relationship Diagram (ERD)***

Menurut Sukamto dan Shalahudin (2013:50-52). “Pemodelan awal basis data yang paling banyak digunakan adalah menggunakan *Entity Relationship Diagram* (ERD). ERD dikembangkan berdasarkan teori himpunan dalam bidang matematika. ERD digunakan untuk pemodelan basis data relasional. Sehingga jika penyimpanan basis data menggunakan ERD. ERD memiliki beberapa aliran notasi seperti notasi Chen (dikembangkan oleh Peter Chen), Barker (dikembangkan oleh Richard Barker,Ian Palmer,Harry Ellis), notasi Crow’s Foot, dan beberapa notasi lain. Namun yang banyak di gunakan adalah notasi Chen.

ERD biasanya memiliki hubungan binary (satu relasi menghubungkan dua buah entitas). Beberapa metode perancangan ERD menoleransi hubungan relasi *ternary* (satu relasi menghubungkan tiga buah relasi) atau N-ary (satu relasi menghubungkan banyak entitas),tapi banyak metode perancangan ERD yang tidak mengizinkan hubungan *ternary* atau *N-ary*. ERD untuk memodelkan struktur data dan hubungan antar data, untuk menggambarkannya digunakan beberapa notasi dan simbol. Pada dasarnya ada tiga simbol yang digunakan, yaitu :

Entitas (*Entity)*

Merupakan objek yang mewakili sesuatu yang nyata dan dapat dibedakan dari sesuatu yang lain. Simbol dari *entity* ini biasanya digambarkan dengan persegi panjang. Entitas menunjukkan objek-objek dasar yang terkait didalam sistem. Objek dasar dapat berupa orang, benda atau hal lain yang keterangannya perlu disimpan dalam basis data.

* 1. Atribut (*Attribute*)

Setiap entitas pasti mempunyai elemen yang disebut atribut yang berfungsi untuk mendeskripsikan karakteristik dari entitas tersebut. Macam-macam atribut sebagai berikut:

1. *Simple attribute* yaitu atribut yang terdiri atas satu komponen tunggal dengan keberadaannya yang independent dan tidak dapat dibagi lagi ke komponen yang lebih kecil. *Simple attribute* dikenal juga dengan nama *atomic attribute*.
2. *Composite attribute* yaitu atribut yang memiliki berbagai komponen dimana semua komponennya memiliki keberadaan yang independen.
3. *Single value attribute* yaitu sebuah atribut yang mempunyai nilai tunggal untuk setiap kejadian.
4. *Multi value attribute* yaitu sebuah atribut yang mempunyai beberapa nilai untuk setiap kejadian pada sebuah entitas.
5. *Derived attribute* yaitu atribut yang memiliki nilai yang dihasilkan dari satu atau beberapa atribut lainnya dan tidak harus dari satu entitas.
   1. Derajat Relasi (*Relationship*)

Relasi didefinisikan sebagai hubungan yang terjadi antar entiti. Representasi diagram relasi adalah sebuah garis lurus yang menghubungkan dua buah *entity*. Jenis-jenis atau hubungan yang biasa terjadi antar satu *entity* dengan *entity* lain dalam sebuah basis data meliputi :

*One to One* / Satu ke Satu

Hubungan relasi satu ke satu yaitu setiap entitas pada himpunan entitas A berhubungan paling banyak dengan satu entitas pada himpunan entitas B. Contoh hubungan antara *entity* Mahasiswa dengan Kartu Mahasiswa. Seorang mahasiswa hanya boleh memiliki satu kartu mahasiswa, satu kartu mahasiswa hanya dimiliki oleh satu orang mahasiswa.

*One to Many* / Satu ke Banyak

Setiap entitas pada himpunan entitas A dapat berhubungan dengan banyak entitas pada himpunan B, tetapi setiap entitas pada entitas B dapat berhubungan dengan satu entitas pada himpunan entitas A. Contohnya hubungan yang terjadi antara entiti Konsumen dengan Motor. Seorang konsumen dapat memiliki satu atau lebih motor. Sedangkan satu motor hanya dapa dimiliki oleh seorang konsumen.

*Many to Many* / Banyak ke Banyak

Setiap entitas pada himpunan entitas A dapat berhubungan dengan banyak entitas pada himpunan entitas B.

* 1. Keterkaitan (*Cardinality Ratio*)

Menjelaskan batasan jumlah keterhubungan suatu entitas dengan entitas lainnya. Terdapat tiga jenis *cardinality ratio* yaitu satu ke satu,satu ke banyak dan banyak ke banyak.

* 1. *Participation Constraint*

Menjelaskan apakah keberadaan suatu entitas tergantung pada hubungannya dengan entitas lain. Terdapat dua macam *Participation Constrain* :

1. *Total Participation*

Menjelaskan keberadaan suatu entitas tergantung pada hubungannya dengan entitas lain.

1. *Partial Participation*

Menjelaskan keberadaan suatu entitas tidak tergantung pada hubungannya dengan entitas lain.

**F. *LRS* (*Logical Record Structure*)**

Menurut Sukamto dan Shalahudin (2013:58) “LRS merupakan hasil dari pemodelan *Entity* *Relationship (ER)* beserta atributnya sehingga bisa terlihat hubungan-hubungan antar entitas”.

Dalam pembuatan LRS terdapat tiga hal yang dapat mempengaruhi, menurut Sukamto dan Shalahudin (2013:58) yaitu:

* + 1. Jika tingkat hubungan *(cardinality)* satu pada satu *(one-to-one)*, maka digabungkan dengan entitas yang lebih kuat *(strong entity)*, atau digabungkan dengan entitas yang memiliki atribut yang lebih sedikit.
    2. Jika tingkat hubungan *(cardinality)* satu pada banyak *(one-to-many)*, maka hubungan relasi atau digabungkan dengan entitas yang tingkat hubungannya banyak.
    3. Jika tingkat hubungan *(cardinality)* banyak pada banyak *(many-to-many)*, maka hubungan relasi tidak akan digabungkan dengan entitas manapun, melainkan menjadi *LRS*.

**2.2. Penelitian Terkait**

Menurut Nurul Azizah Yaoma Ramadhani (2011:35) menjelaskan bahwa:

Penerimaan siswa baru merupakan salah satu proses yang ada di instansi pendidikan seperti sekolah yang berguna untuk menyaring calon siswa yangterpilih sesuai kriteria yang ditentukan oleh sekolah tersebut untuk menjadi siswa didiknya. Pada umumnya proses penerimaan siswa baru dilakukan melalui tahap pendaftaran, tes seleksi dan pengumuman penerimaan siswa. Dalam penelitian ini penerapannya pada SMK Al-Irsyad Tegal yang selama ini dilakukan secara manual atau sudah terkomputerisasi menggunakan *Microsoft Office Excel*, yang memungkinkan masih banyak kekurangan data, penghitungan nilai tes secara manual yang memungkinkan terjadinya kesalahan. Untuk itu dibutuhkan suatu system informasi yang dapat membantu dalam proses penerimaan siswa baru. Rumusan masalah dari penelitian ini adalah bagaimana proses membangun system informasi penerimaan siswa baru di SMK Al-Irsyad Tegal sehingga dapat menyajikan informasi yang akurat serta efisien. Adapun tujuan dari penelitian ini adalah menghasilkan suatu program system aplikasi. Manfaat dari penelitian ini adalah memberikan kemudahan bagi admin maupun siswa itu sendiri. Metode penelitian yang digunakan adalah pustaka, observasi, wawancara, analisis, perancangan, uji coba dan implementasi. Kesimpulan hasil dari penelitian ini adalah diharapkan dapat terbangunnya system informasi penerimaan siswa baru yang berbasis *web*.

Menurut Iwan Ady Prabowo (1693-1173) menjelaskan bahwa:

SMAN 1 Ngemplak adalah salah satu dari sekian banyak sekolah yang ada di Boyolali, sehingga keberadaannya sangat dikenal oleh masyarakat. Selama ini pengumuman tentang penerimaan peserta didik baru (PPDB) masih menggunakan edaran brosur. Penyampaian informasi pendaftaran belum menggunakan teknologi yang saat ini sedang berkembang yaitu pelayanan *online* dengan pemanfaatan teknologi informasi internet. Padahal kalu dilihat dari sudut pandang konsumen, teknologi ini sangat *user friendly.* Hal ini sangat menguntungkan bagi SMAN 1 Ngemplak untuk bias memberikan layanan informasi kepada calon siswa baru selaku konsumen di Ngemplak dan masyarakat umum dalam rangka mempromosikan SMAN1 Ngemplak. Oleh karena itu agar masyarakat luas lebih mengetahui tentang SMAN 1 Ngemplak sendiri maka diperlukan sebuah sistem informasi sebagai sarana penyampaian informasi kepada masyarakat. Perumusan masalahnya adalah bagaimana menciptakan suatu aplikasi yang dapat membantu menyampaikan informasi Penerimaan Peserta Didik Baru (PPDB) di SMAN 1 Ngemplak. Tujuan dari penelitian ini adalah membuat sistem informasi penerimaan peserta didik baru (PPDB) berbasis *web* ke dalam aplikasi komputer. Sehingga sistem yang ada nanti lebih efektif dan efisien dapat memudahkan dalam operasional pelayanan penerimaan peserta didik baru.