# BAB II

# TINJAUAN PUSTAKA

## A. Penelusuran Referensi

Dalam penelitian ini penulis memaparkan dua penelitian terdahulu yang relevan dengan permasalahan yang akan diteliti tentang perancangan sistem informasi desa.

Muhammad Rizal Husain Muttaqien (2014) jurnalnya yang berjudul "Analisis dan Perancangan Sistem Informasi Kependudukan Pada Pemerintah Desa Bangunjiwo Berbasis Web" Mempaparkan bagaimana kemudahan dalam mengakses informasi - informasi yang ada di desa.

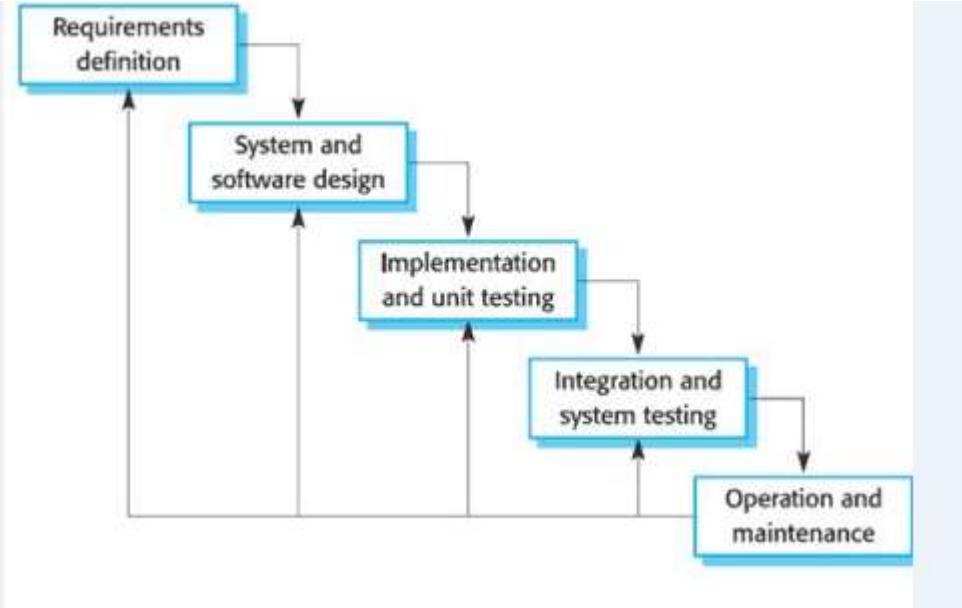
Ardi Prihananto (2012) jurnalnya yang berjudul "Analisis dan Perancangan Sitem Informasi Berbasis Web pada Desa Denggungan Sebagai Media Promosi Boyolali" Mempaparkan kesimpulan setelah adanya web tersebut, dapat membantu masyarakat luas untuk mengetahui tempat – tempat pariwisata yang ada di boyolali terutama di desa Denggungan.

Keunggulan dari Sistem Informasi yang akan dibuat ini dibandingkan dengan jurnal sebelumnya adalah desain *interface* lebih menarik, informasi yang didapat lebih banyak, dan memudahkan masyarakat dalam mengurus surat menyurat.

## B. Metode Perancangan Sistem

Metode perancangan sistem yang ada hingga saat ini terdapat beberapa macam model. Mulai dari air terjun (*waterfall*), model *evolusioner*, model *spiral* dan beberapa model lainnya. Namun model yang digunakan untuk perancangan sistem pada tugas akhir ini menggunakan model air terjun atau

*waterfall* model. Menurut (Al- Bahra bin Ladjamudin B;2006;16) model *waterfall* adalah model yang diperoleh dari proses *engineering* (proses rekasa) lain. Inti dari metode *waterfall* adalah pengerjaan dari suatu sistem dilakukan secara berurutan atau secara linear. Jadi jika langkah ke-1 belum dikerjakan, maka langkah 2 tidak dapat dikerjakan. Jika langkah ke-2 belum dikerjakan maka langkah ke-3 juga tidak dapat dikerjakan, begitu seterusnya. Secara otomatis langkah ke-3 akan bisa dilakukan jika langkah ke-1 dan ke-2 sudah dilakukan. Seperti terlihat dibawah ini :




Sumber (Al Bahra Ladjamudin B, 2006:18)

Gambar 2.1 siklus hidup *watterfall*

Model *waterf*all merupakan model yang sering digunakan untuk membangun sistem yang praktis. Metode ini menawarkan cara pembuatan perangkat lunak secara lebih nyata. Langakah-langkah yang penting dalam model ini meliputi :

1. Definisi Persyaratan

   Batasan dan tujuan sistem ditentukan melalui konsultasi dengan *user* sistem. Persyaratan ini kemudian didefinisikan secara rinci dan berfungsi

sebagai spesifikasi sistem., kemudian semuanya itu dibuat dalam bentuk yang dapat dimengerti oleh *user*.

2. Desain sistem dan perangkat lunak

   Proses perancangan sistem membagi persyaratan dalam sistem perangkat lunak atau perangkat keras. Kegiatan ini menentukan arsitektur secara keseluruhan.

3. Implementasi dan ujicoba unit

   Selama tahap ini perangkat lunak disadari sebagai sebuah program lengkap atau unit program. Ujicoba unit termasuk pengujian bahwa setiap unit sesuai spesifikasi.

4. Integrasi dan ujicoba unit

   Unit program diintegrasikan dan diuji menjadi sistem yang lengkap untuk menyakinkan bahwa persyaratan perangkat lunak telah dipenuhi. Setelah ujicoba, sistem disampaikan kepada *user*.

5. Pemeliharaan

   Normalnya pada tahap ini merupakan tahap yang memerlukan waktu yang paling panjang. Setelah sistem dipasang dan digunakan berikut adalah tahap pemeliharaan yang meliputi koreksi beberapa *error* pada tahap-tahap terdahulu, perbaikan atas implementasi unit sistem dan pengembangan layanan sistem, sementara persyaratan baru ditambahkan.

## C. Desa Sidoharjo

Desa berasal dari bahasa sansekerta dhesi yang berarti “Tanah kelahiran”. Desa identik dengan kehidupan agraris dan kesederhanaannya. Menurut kamus besar bahasa Indonesia desa merupakan kesatuan wilayah

yang dihuni oleh sejumlah keluarga yang mempunyai sistem pemerintahan sendiri (dikepalai oleh seorang kepala desa) atau desa merupakan kelompok rumah diluar kota yang merupakan kesatuan.

Desa Sidoharjo adalah sebuah desa di wilayah Kecamatan Pulung Kabupaten Ponorogo Provinsi Jawa Timur. Desa Sidoharjo terdiri dari beberapa dukuh , Dukuh Sukun Etan, Dukuh Sukun Kulon, Dukuh Sidoharjo, Dukuh Segajah. Sebagian besar mata pencaharian penduduk bercocok tanam.

## D. Sistem Informasi

1. Sejarah Sistem Informasi

Sistem informasi adalah kombinasi dari manusia , fasilitas atau alat teknologi, media, prosedur dan pengendalian yang bermaksud menata jaringan komunikasi yang penting menurut John F. Nash (2003). Menurut kusnadi dkk. (2008) perkembangan sistem operasi sangat dipengaruhi oleh Perkembangan Hardware, yaitu :

a. Generasi ke-nol 1940

Komponen utama tabung hampa udara. Sistem computer belum menggunakan sistem operasi. Semua operasi computer dilakukan secara manual melalui *plugboards,* dan hanya bisa digunakan untuk menghitung (+,-, dan *).

b. Generasi ke-pertama 1950

Komponen utama adalah transistor. Sistem operasi berfungsi terutama sebagai pengatur pergantian antar *job* agar waktu instalasi *job* berikutnya lebih efisien. Input menggunakan *punch card.*

c. Generasi ke-dua 1960

Komponen pembentuk utama adalah IC. Berkembangnya konsep-konsep seperti *Multiprograming, Multiprocessing, Spooling (Simultaneous Peripheral Operation On Line), Device Independence, Time Sharing atau Multitasking,* Dan *Real Time system.*

d. Generasi ke-tiga 1970

Komponen pembentuk utama adalah VLSI (*Very Large Scale Integrated Circuit*). Perkembangannya ditandai dengan konsep *general purpose system*, sehingga sistem informasi menjadi sangat kompleks, mahal dan sulit dipelajari.

e. Generasi ke-empat (pertengahan 1970-an hingga sekarang)

Personal Komputer semakin popular. Berkembangnya sistem informasi untuk jaringan computer dengan tujuan: data *sharing*, *hardware sharing* dan program *sharing*. *User interface* semakin *user friendly* tanpa harus mengorbangkan unjuk kerjanya.

1. Fungsi Sistem Operasi

Dari sejarah pengembangan sistem operasi terlihat bahwa sistem operasi dirancang sedemikian rupa untuk :

a. Sebagai mesin abstraksi (*extended machine*)

Sistem operasi melakukan abstraksi pengaksesan sumber daya computer sehingga tersedia lingkungan (antarmuka) dan layanan yang nyaman dan mudah bagi program aplikasi maupun pengguna.

Komputer terdiri atas berbagai macam perangkat keras seperti prosesor, memori, pewaktu, media penyimpanan, dan piranti I/O lainnya. Pada awal perkembembangan computer, dimana belum ada sistem operasi, semua operasi computer dilakukan secara manual menggunakan panel control oleh para ahli saja.

Sistem operasi kemudian dikembangkan secara bertahap untuk mengotomatisasi hal-hal yang tadinya harus dilakukan secara manual. Selain itu, sistem operasi menyembunyikan detil operasi perangkat dari keras serta menyediakan antarmuka natar operasi yang lebih sederhana dan mudah dimengerti oleh pengguna. Dengan kata lain, sistem operasi melakukan abstraksi bagi pengguna terhadap pengaksesan sumber daya komputer.

Sistem operasi melakukan abstraksi terhadap pengaksesan terhadap sumber daya computer sehingga pengoperasian komputer menjadi mudah bagi pengguan. Dengan adanya sistem operasi, pengguna hanya melihat sebagai mesin yang dapat menjalankan tugas-tugas yang diberikan kepadanya, seperti memutar lagu, melakukan perhitungan matematis, membaca basis data dengan cara yang sangat mudah. Pengguna tidak perlu tahu bagaimana sesungguhnya mekanisme kerja internal computer seperti menyalin kode intruksi program ke memori, melakukan perhitungan di prosesor, menyimpan data ke suatu alamat di disk, dan bagaimana carnya data-data dikirim ke *printer* lewat kabel *parallel* serta detail operasi lainnya.

b. Sebagai pengolah sumber daya manusia (*resource manager*)

Sistem operasi mengolah seluruh sumber daya computer sehingga terpakai secara efisien,efektif, dan aman. Pada sistem computer dimugkingkan beberapa job dijadwal secara bergantian untuk menggunakan prosesor. Sumber daya computer dibagi pakai oleh job-job tersebut. Caranya adalah dengan melakukan multipleks (*multiplexing*) pada penggunaan sumber daya, baik dari segi waktu maupun ruang.

Contoh sumber daya yang dimultipleks secara waktu adalah prosesor dan printer. Multi programming pada prosesor tunggal dicapai dengan melakukan alokasi prosesor kepada masing-masing job secara bergantian. Contoh sumber daya yang dimultipleks secara ruang adalah memori dan disk.ruang memoriu dipartisi untuk menyimpan beberapa job secara bersama. Sistem operasi harus memastikan program yang satu tidak menulis atau memodifikasi partisi memori job lainnya.

## E. PHP

Kasiman Peranginangin (2006) mengatakan bahwa, PHP merupakan singkatan dari *PHP Hypertext Preprocessor.* PHP digunakan sebagai bahasa *script server-side* dalam pengembangan Web yang disisipkan pada dokumen HTML. Penggunaan PHP memungkinkan Web dapat dibuat dinamis sehingga *maintenance* situs Web menjadi lebih mudah dan efisien. PHP ditulis menggunakan bahasa C. PHP memiliki banyak kelebihan yang tidak dimiliki oleh bahasa *script* sejenis. PHP difokuskan pada pembuatan *script server-side,* yang bisa melakukan apa saja yang dilakukan oleh CGI, seperti mengumpulkan data dari form, menghasilkan isi halaman web dinamis, dan

kemampuan mengirim serta menerima *cookies*, bahkan lebih daripada kemampuan CGI. PHP tidak terbatas pada hasil keluaran HTML *(HyperText Markup Language).* PHP juga memiliki kemampuan untuk mengolah gambar, file PDF, dan movie flash. PHP juga dapat menghasilkan teks seperti XHTML dan file XML lainnya. Salah satu fitur yang dapat diandalkan oleh PHP adalah dukungannya terhadap banyak database, salah satunya adalah MySQL.

## F. My SQL

My SQL adalah salah satu jenis database server yang sangat terkenal (madcoms,2007) .Data adalah bagian penting dari pemrograman modern sehingga keseluruhan bahasa program menyediakan fungsi untuk mengakses *database.* Standar utama untuk bahasa *database* adalah *Structured Query Language* (SQL). SQL distandarisasi sebagai bahasa untuk menciptakan *database,* menyimpan informasi ke dalam *database,* dan mendapatkan kembali informasi darinya. Aplikasi khusus dan lingkungan pemrograman mengkhususkan diri untuk menginterpretasikan data SQL.

Seorang programmer akan memulai dengan menciptakan suatu struktur data di dalam SQL dan kemudian menulis suatu program dalam bahasa (PHP) untuk mengakses data tersebut. Program PHP kemudian bisa memformulasikan permintaan atau memperbaharui data tersebut, yang dilewatkan ke interpreter SQL.

a. Konsep *Database*

1) Entitas dan *Relationship*

Entitas adalah berbagai hal dalam dunia nyata yang informasinya disimpan dalam *database*. Sebagai contoh, kita dapat menyimpan informasi pegawai dan bekerja untuk departemen tertentu. Dalam kasus ini, pegawai merupakan suatu entitas dan departemen juga merupakan entitas.

*Relationship* adalah hubungan antar entitas. Sebagai contoh, seorang pegawai bekerja untuk suatu departemen. Bekerja untuk adalah *relationship* antara entitas pegawai dan entitas departemen.

*Relationship* terdiri dari tiga derajat berbeda, yakni *one-to-one*, *one-to-many (many-to-one)*, dan *many-to-many*.

*One-to-one* menghubungkan secara tepat dua entitas dengan satu kunci (*key*). Misalnya, dalam suatu perusahaan satu orang pegawai memiliki satu komputer saja.

*One-to-many (many-to-one)* merupakan hubungan antar entitas dimana kunci (*key*) pada satu tabel muncul berkali-kali dalam table lainnya. Misalnya, banyak pegawai bekerja untuk satu perusahaan.

*Many-to-many* merupakan hubungan yang sering menyebabkan permasalahan dalam prakteknya. Dalam hubungan *many-to-many*, kunci utama (*primary key*) dari tabel kedua dapat muncul beberapa kali pada tabel pertama. Misalnya, dalam suatu perusahaan, banyak pegawai bekerja untuk banyak departemen. Untuk mengatasi permasalahan tersebut, dibutuhkan tabel antara.

2) Relasi atau Tabel

*Database* terdiri dari sekumpulan relasi atau tabel. Relasi dan tabel memiliki arti yang sama. Perhatikan contoh tabel pegawai pada tabel berikut.

Tabel 2.1 Tabel Pegawai

| No Pegawai | Nama | Pekerjaan | Kode Departemen |
|---|---|---|---|
| **234** | Rina Erlinda | Programmer | 14 |
| **567** | Nora | Programmer | 14 |
| **456** | Ricco | DBA | 12 |
| **678** | Kasiman | Programmer | 14 |

Sumber Madcoms,2007

3) Kolom atau *Attribute*

Dalam tabel database, setiap kolom atau *attribute* menjelaskan beberapa bagian *record* data yang disimpan dalam tabel. Kolom adalah bagian dari tabel, sedangkan suatu *attribute* berkaitan dengan entitas dunia nyata yang merupakan pemodelan tabel. Seperti pada tabel pegawai sebelumnya, dapat dilihat bahwa setiap pegawai memiliki satu NoPegawai, Nama, Pekerjaan, dan KodeDepartemen yang merupakan kolom, tetepi sering juga disebut *attribute* dari table pegawai.

4) Baris, *Record*, dan *Tuple*

Pada tabel pegawai sebelumnya, setiap baris pada table mewakili suatu *record* pegawai. Setiap baris dalam tabel sering juga disebut *record* atau *tuple* yang terdiri dari suatu nilai untuk setiap kolom dalam tabel.

5) Kunci (*Key*)

Suatu *superkey* adalah suatu kolom yang dapat digunakan untuk mengidentifikasi suatu baris dalam tabel. Suatu *key* adalah suatu minimal *superkey*. Sebagai contoh, pada tabel pegawai sebelumnya, kita dapat

menggunakan NoPegawai dan Nama secara bersama-sama untuk mengidentifikasi baris-baris dalam tabel. Kita juga dapat menggunakan seluruh klom sebagai *superkey*.

Namun, kita tidak membutukan seluruh kolom tersebut untuk mengidentifikasi suatu baris. Kita hanya butuh NoPegawai. Ini adalah suatu minimal *superkey* yang merupakan suatu minimal kolom yang dapat digunakan untuk mengidentifikasi suatu baris tunggal, maka NoPegawai adalah suatu *key*.

Kita dapat mengidentifikasi seorang pegawai dengan Nama atau dengan NoPegawai yang merupakan dua *key* yang disebut *candidate key*. Disebut *candidate key* karena dengan kedua kolom tersebut kita akan memilih salah satu yang menjadi *primary key*. *Primary key* adalah sutu kolom atau sekumpulan kolom yang akan digunakan untuk mengidentifikasi secara tunggal setiap baris dari suatu tabel. Dalam hal ini, kita akan membuat NoPegawai sebagai *primary key* karena secara umum Nama mungkin ada yang sama.

Foreign *key* menyatakan hubungan antar tabel. Sebagai contoh, pada tabel pegawai dapat dilihat bahwa KodeDepartemen menyimpan suatu nomor departemen yang akan disimpan dalam suatu tabel terpisah dengan *primary key*-nya adalah Kode Departemen.

b. MySQL

Ada sejumlah paket *Relational Database Management System* (RDBMS) yang tersedia. Program tersebut bervariasi dalam hal kemampuan, fleksibilitas dan harga. Namun pada dasarnya, semua bekerja dengan cara yang sama. Salah satunya adalah *database* MySQL. MySQL sangat cocok berpasangan dengan bahasa pemrograman PHP.

MySQL merupakan program dengan lisensi *open source* dan tersedia secara cuma-cuma. MySQL mampu bekerja pada berbagai sistem informasi, dan banyak bahasa. MySQL bekerja dengan cepat dan baik dengan data yang besar. Selain itu, PHP juga menyediakan banyak fungsi untuk mendukung *database* MySQL.

## G. UML

"UML *(Univied Modeling Language)* adalah bahasa pemodelan untuk system atau perangkat lunak yang berparadigma berorientasi objek" (Adi Nugroho,2010:6) Sehingga dapat disimpulkan, UML *(Univied Modeling Language)* adalah pemodelan objek yang digunakan untuk menyajikan suatu sistem yang berorientasi pada objek.

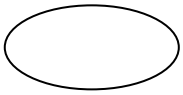
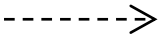
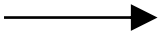
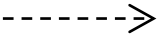
UML merupakan sintak umum untuk membuat model logika dari suatu sistem dan digunakan untuk menggambarkan sistem agar dipahami selama fase analisis dan desain.UML biasanya disajikan dalam bentuk diagram/gambar yang meliputi *class* beserta atribut dan operasinya.

### *1. Use Case*

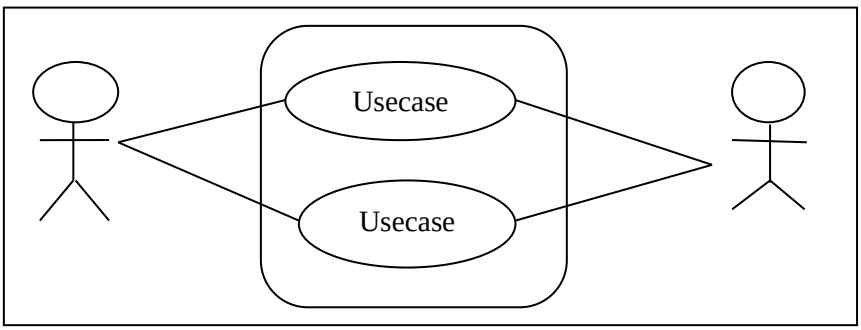
*Use case* menurut Satzinger et al., (2010, p.242) "merupakan suatu aktivitas yang dilakukan oleh sistem, biasanya merupakan sebuah respon untuk permintaan dari pengguna sistem". Satzinger et al., (2010, p.243) "menjelaskan bahwa aktor tidak selalu sama dengan sumber dari peristiwa di *event table* karena actor di *use case* merupakan orang yang berinteraksi dengan sistem yang mana sistem harus meresponnya".

Tabel 2.2 Relasi – relasi Pada *Usecase*

| relasi | Fungsi | Notasi |
|---|---|---|

| | | |
|---|---|---|
| **Aktor** | Digunakan untuk menggambarkan seseorang atau sesuatu. |  |
| ***Use case*** | Digunakan untuk pemberian nama dalam kegunaan sistem yang akan dilakukan. |  |
| **Asosiasi (association)** | Lintasan komunikasi antara actor dengan usecase |  |
| **Extend** | Penambahan perilaku ke suatu use case dasar |  |
| **Generalisasi use case** | Menggambarkan hubungan antara use case yang bersifat umum ke yang lebih spesifik |  |
| **Include** | Penambahan perilaku ke suatu use case dasar secara eksplisit mendeskripsikan penambahan tersebut |  |

Sumber Satzinger et al., (2010, p.242)




Sumber Satzinger et al., (2010, p.242)

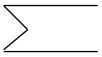
Gambar 2.2 *Usecase* Umum

2. *Activity* Diagram

*Activity diagram* menurut Martin Fowler (2005 : 163) adalah "teknik untuk menggambarkan logika prosedural, proses bisnis, dan jalur kerja". sedangkan Menurut Satzinger et al (2010:141), "*Activity* diagram merupakan sebuah tipe dari diagram *workflow* yang menggambarkan tentang aktivitas dari pengguna ketika melakukan setiap kegiatan dan aliran sekuensial". Dalam beberapahal, *activity diagram* memainkan peran mirip diagram alir, tetapi perbedaan prinsip antara notasi diagram alir adalah *activity diagram* mendukung *behaviorparalel*. *Node* pada sebuah *activity diagram* disebut sebagai *action*, sehingga diagram tersebut menampilkan sebuah *activity* yang tersusun dari *action*.

Tabel 2.3 Simbol *Activity* Diagram

| Simbol | Keterangan |
|---|---|
|  | Titik Awal |
|  | Titik Akhir |
|  | Activity |
|  | Pilihan untuk pengambilan keputusan |
|  | Fork; untuk menunjukkan kegiatan yang dilakukan secara paralel |
|  | Rake;menunjukkan adanya |

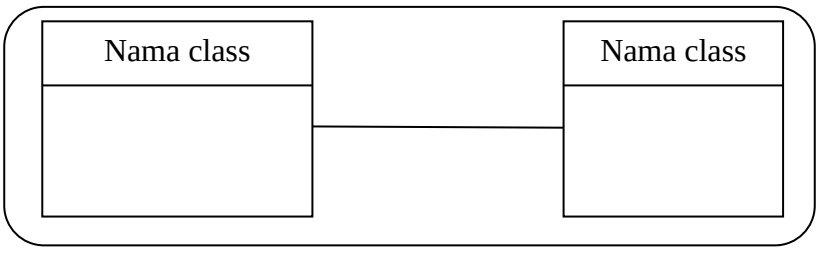
| | |
|---|---|
| | dekomposisi |
|  | Tanda Waktu |
|  | Tanda Penerimaan |
|  | Aliran Akhir (*Flow Final*) |

Sumber Satzinger et al (2010:141),

3. *Class* Diagram

*Class* diagram menurut Satzinger et al (2010:141),) "merupakan himpunan dari objek-objek yang sejenis. Sebuah objek memiliki keadaan sesaat (*state*) dan perilaku (*behavior*)". *State* sebuah objek adalah kondisi objek tersebut yang dinyatakan dalam *attribute/properties*. Sedangkan perilaku suatu objek mendefinisikan bagaimana sebuah objek bertindak/beraksi dan memberikan reaksi.Objek adalah niali tertentu dari setiap *attribute* kelas *entity*. Class memiliki tiga area pokok :

1. Nama (dan *stereotype*)
2. Atribut
3. Metoda




Sumber Satzinger et al (2010:141),

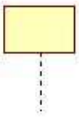
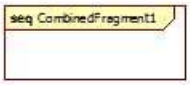
Gambar 2.3 *class* diagram

4. *Sequence* Diagram

*Sequence* diagram menurut Satzinger et al (2010:141) menggambarkan interaksi antar objek didalam dan disekitar sistem (termasuk pengguna,

display,dan sebagainya) berupa message yang digambarkan terhadap waktu. *Sequence* diagram terdiri atas dimensi vertical (waktu) dan dimensi horizontal (objek-objek yang terkait).

*Sequence* diagram biasa digunakan untuk menggambarkan scenario atau rangkaian langkah – langkah yang dilakukan sebagai respon dari sebuah *event* untuk menghasilkan *output* tertentu. Adapun komponen – komponen yang digunakan dalam *sequence* diagram sebagai berikut:

| Komponen | Keterangan |
|---|---|
|  | *Actor* |
|  | *Object* |
|  | *Combined Fragment* |
|  | *Frame* |
|  | *Stimulus* |
|  | *SelfStimulus* |

Sumber Satzinger et al (2010:141),

Gambar 2.4 komponen *sequence* diagram