BAB IV

METODE PENELITIAN

4.1. Lokasi dan Waktu Penelitian

4.1.1. Lokasi Penelitian

Penelitian ini dilakukan di PT. Dharma Mulia Buana Abadi yang beralamat di Jalan Rudus No. 39 Kel. 20 Ilir, Palembang.

4.1.2. Waktu Penelitian

Penulis melakukan penelitian Pada PT. Dharma Mulia Buana Abadi Palembang selama 1 (satu) bulan, yakni dimulai tanggal 1 November 2017 hingga 30 November 2017.

4.2. Jenis Data

Sumber data adalah segala sesuatu yang dapat memberikan informasi mengenai data. Berdasarkan sumbernya, data dibedakan menjadi dua, yaitu data primer dan data sekunder.

1. Data primer yaitu data yang dibuat oleh peneliti untuk maksud khusus menyelesaikan permasalahan yang sedang ditanganinya. Data dikumpulkan sendiri oleh peneliti langsung dari sumber pertama atau tempat objek

penelitian dilakukan. Data yang dikumpulkan berupa data harga sewa alat-berat, data company profile, dan data kontrak pada *client*.

2. Data sekunder yaitu data yang telah dikumpulkan untuk maksud selain menyelesaikan masalah yang sedang dihadapi. Data ini dapat ditemukan dengan cepat. Dalam penelitian ini yang menjadi sumber data sekunder adalah literatur, artikel, jurnal serta situs di internet yang berkenaan dengan penelitian yang dilakukan.

4.3. Teknik Pengumpulan Data

Teknik pengumpulan data pada penelitian ini menggunakan teknik *observasi,* wawancara dan studi pustaka, berikut penjelasannya:

1. *Observasi* (Pengamatan)

Berdasarkan hasil observasi yang dilakukan, penulis melakukan pengamatan terhadap tim marketing dalam penyampaian informasi *company profile* dan penawaran jasa pada PT Dharma Mulia Buana Abadi. Dalam pemesanan jasa konstruksi dimulai dari permintaan harga dari *client*, lalu *purchasing* perusahan ini memberikan penawaran harga. Dari hasil negosiasi antara *client* dan *purchasing* ini, kemudian *client* membuat *purchase order* (PO). Apabila *purchasing* perusahaan *meeting* dengan *client*, lalu ada *client* yang lain meminta penawaran harga, maka client yang meminta penawaran harga harus menunggunya.

2. *Interview* (Wawancara)

Pada tahap wawancara ini, Penulis melakukan wawancara dengan Bapak Safriyanto selaku Direktur PT Dharma Mulia Buana Abadi mengenai *profile* perusahan dan proses pemesanan jasa konstruksi untuk mendapatkan informasi dalam penyampaian *company profile* dan penawaran jasa pada PT Dharma Mulia Buana Abadi.

3. Studi Pustaka

Dalam menggunakan teknik studi pustaka, penulis mengumpulkan data dengan cara mengambil data-data dari jurnal serta buku-buku yang ada kaitannya dengan aplikasi web berbasis objek.

4.4. Alat dan Teknik Pengembangan Sistem

4.4.1. Alat Pengembangan Sistem

UML *(Unified Modeling Language)* adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek. (Rosa A.S dan M. Shalahudin, 2014:133).

1. *Diagram UML*

Rosa A.S dan M. Shalahudin (2014:140), pada *UML* terdiri dari 13 macam diagram yang dikelompokkan dalam 3 kategori. Berikut ini penjelasan singkat dari pembagian kategori tersebut.

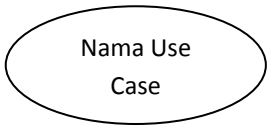
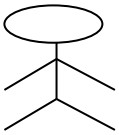
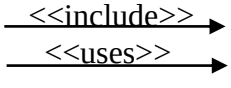
1. *Structure diagram*, yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan. *Structure diagram* terdiri dari *class diagram, object diagram, component diagram, composite structure diagram, package diagram dan deployment diagram.*
2. *Behavior diagram* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem. Behavior diagram terdiri dari *Use case diagram, Aktivity diagram, State Machine System.*
3. *Interaction diagram* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem. *Interaction diagram* terdiri dari *Sequence Diagram, Communication Diagram, Timing Diagram, Interaction Overview Diagram.*

2. *Use Case* Diagram

Rosa dan M. Shalahudin (2014:155), *use case* atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih actor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui

fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Tabel 4.1. berisi simbol-simbol yang ada pada diagram *use case*:

Tabel 4.1. *Diagram Use Case*

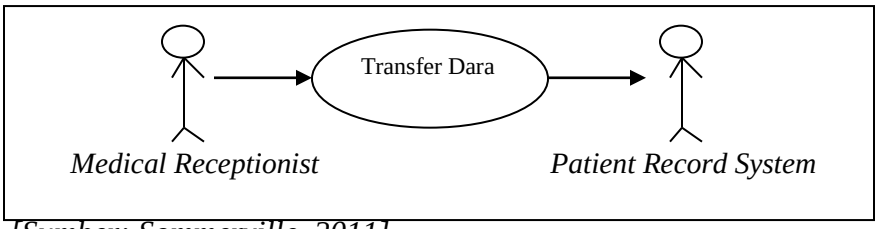
Simbol	Deskripsi
Use Case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja diawal frase nama *use case*.
Aktor/*actor* 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi actor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
Assosiasi/*association*	Komunikasi antara aktor dan *use case* yang berpartisilpasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
Exstensi/*extend* 	Relasi *use case* tambahan kesebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu, mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek, biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan.
Generalisasi/ *generalization*	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua *buah use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya,
Menggunakan / *include* / *uses* 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini

(Sumber : Rosa A.S dan M. Shalahudin (2014:156))

Pemodelan *use case* pada awalnya dikembangkan oleh Jacobson pada 1990-an dan dimasukan dalam rilis pertama UML. *Use case* secara luas digunakan untuk mendukung persyaratan dari *elicitation*. *Use case* dapat diambil sebagai skenario sederhana yang menggambarkan apa yang pengguna harapkan dari suatu sistem.

Setiap *use case* merepresentasikan tugas terpisah yang melibatkan interaksi eksternal dengan sistem. *Use case* mengindentifikasi interaksi individu antara sistem dan pengguna atau sistem lain. Setiap *use case* harus didokumentasikan dengan deksripsi tekstual, yang kemudian dapat dikaitkan dengan model lain dalam UML yang akan membangun skenario dengan lebih terperinci. (Sommerville, 2011:107).

Use case adalah abstraksi dari interaksi antara *system* dan *actor*. *Use case* bekerja dengan cara mendeskripsikan tipe interaksi antara *user* sebuah *system* dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah *system* dipakai. *Use case* merupakan konstruksi untuk mendeskripsikan bagaimana *system* akan terlihat di mata *user*. Sedangkan *use case diagram* memfasilitasi komunikasi diantara analis dan pengguna serta antara analis dan *client*.



[Sumber: Sommerville, 2011]

Gambar 4.1. *Use case* Informal

Use case terdiri dari:

1. *Actor*: pemakai sistem atau sesuatu yang berinteraksi dengan sistem merepresentasikan pesan, bukan pemakai individu.
2. *Use Case*: cara spesifikasi penggunan sistem oleh *actor*.

Tujuan utama memodelkan *use case*:

1. Memutuskan dan mendeskripsikan kebutuhan fungsional sistem.
2. Memberikan deskripsi yang jelas dan konsisten dari apa yang harus dilakukan.
3. Melakukan basis yang menyediakan pengujian sistem yang menverifikasi sistem.
4. Menyediakan kemampuan melacak fungsi analistis menjadi clas-clas, operasi-operasi, dan aktual sistem.

Ciri-ciri *use case*:

1. Terdapat pola perilaku yang harus dipenuhi oleh sistem.
2. Terdapat sekuen transaksi terhubung yang dilakukan oleh aktor dan sistem.
3. Memberikan informasi yang berharga bagi *user*.

Kegunaan *use case*:

1. Menangkap kebutuhan sistem.
2. Berkomunikasi dengan pemakai akhir dan pakar domain masalah.
3. Pengkajian sistem.

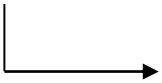
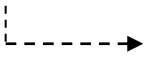
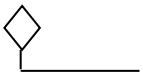
3. Class Diagram

Rosa dan M. Shalahudin (2014:141), diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan *method* atau operasi. Berikut penjelasan atribut dan *method* :

1. Atribut merupakan variable-variabel yang dimiliki oleh suatu kelas.
2. Operasi atau *method* adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Tabel 4.2. berisi simbol-simbol yang ada pada diagram kelas:

Tabel 4.2. *Class Diagram*

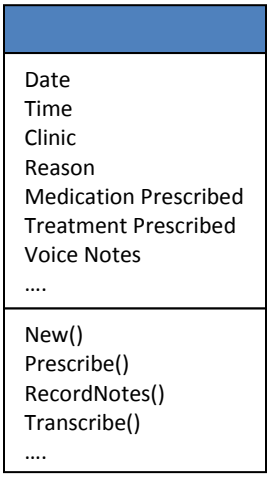
Simbol	Deskripsi
Kelas **ClassName** **- memberName** **- memberName**	Kelas pada struktur sistem
Antarmuka/*interface* 	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi/*association* 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*
Asosiasi berarah/ *Directed association* 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertaidengan *multiplicity*
Generalisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umumkhusus)
Kebergantungan/ *dependensi* 	Relasi antar kelas dengan makna kebergantungan antar kelas
Agrgasi/*aggregation* 	Relasi antar kelas dengan makna semua-bagian (*whole-part*)

(Sumber : Rosa A.S dan M. Shalahudin (2014:146))

Menurut Sommerville (2011:120), *class diagram* menunjukkan kelas objek dalam sistem dan asosiasi antara kelas-kelas. Diagram kelas digunakan ketika mengembangkan model sistem berorientasi objek untuk menunjukan kelas-kelas dalam sistem dan hubungan antara kelas-kelas tersebut. Sebuah kelas objek dapat dianggap

sebagai definisi umum dari sebuah objek sistem. Sebuah asosiasi adalah *link* antara kelas yang mengindekasikan bahwa ada relasi antara kelas-kelas tersebut. Akibatnya, masing-masing kelas mungkin harus memiliki pengetahuan mengenai kelas yang berkaitan.

Class adalah deskripsi kelompok objek-objek dengan property, perilaku (operasi) dan relasi yang sama. Sehingga dengan adanya Class Diagram dapat memberikan pandangan global atas sebuah system. Hal tersebut tercermin dari class-class yang ada dan relasinya satu dengan yang lainnya. Sebuah sistem biasanya mempunyai beberapa Class Diagram. Class Diagram sangat membantu dalam visualisasi struktur kelas dari suatu system.



[Sumber: Sommerville, 2011]

Gambar 4.2. *The Consultation Class*

Pada gambar 4.2. dapat dirumuskan bahwa:

1. Nama dari *class object* berada di bagian atas.
2. Atribut *class* berada di bagian tengah. Hal tersebut harus juga menyertakan nama atribut, dan pilihan tipenya.
3. *The operation* (dipanggil dalam methods pada Java dan OOP (*Object Oriented Programming*) lainnya), asosiasi dengan kelas objek yang letaknya dibawah persegi panjang tersebut.

Atribut dan metode dapat memiliki salah satu sifat berikut ini:

1. *Private*, tidak dapat dipanggil dari luar *class* yang bersangkutan.
2. *Protected,* hanya dipanggil oleh *class* yang bersangkutan dan anak-anak turunannya.
3. *Public*, dapat dipanggil oleh siapa saja.

Hubungan antar class:

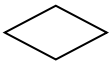
1. Asosiasi yaitu hubungan statis antar *class*. Umumnya *class* yang menggambarkan Atribut berupa *class* lain, atau *class* yang harus mengetahui eksistensi *class* lain. Panah menunjukan arah *query* antar *class.*
2. Agregasi yaitu hubungan yang menyatakan bagian.
3. Pewarisan yaitu hubungan hirarki antar *class. Class* dapat diturunkan dari *class* lain dan dapat mewarisi atribut dan metode *class* asalnya dan dapat menambahkan fungsionalitas baru, sehingga disebut anak dari *class* yang diwarisinya. Kebalikan dari pewarisan adalah generalisasi.

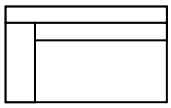
4. Hubungan dinamis yaitu rangkaian pesan yang di-*passing* antara satu *class* kepada *class* lainnya. Hubungan dinamis dapat digambarkan dengan menggunakan diagram *sequence*.

4. *Activity Diagram*

Rosa dan M. Shalahudin (2014:161), diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah 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. Tabel 4.3. berisi simbol-simbol yang ada pada diagram aktivitas:

Tabel 4.3. *Activity Diagram*

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
Percabangan 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
Penggabungan 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
Status akhir 	Status akhir yang dilakukan oleh sistem, sebuah diagram aktivitas memiliki sebuah status akhir.

Simbol	Deskripsi
Swimlane 	Memisahkan organisasi bisnis yang bertanggunga jawab terhadap aktivitas yang terjadi.

(Sumber : Rosa A.S dan M. Shalahudin (2014:162))

Menurut Sommerville (2011,120), *activity* diagram menunjukan aktivitas yang terlibat di dalam proses atau dalam pengolahan data.

Diagram activity seperti *diagram state*, merupakan *diagram* yang digunakan untuk memahami alur kerja dari objek atau komponen yang dilakukan. *Diagram activity* dapat digunakan untuk mevisualisasikan interelasi dan interaksi antara *use case* yang berbeda, serta sering digunakan untuk mengasosiasikan dengan *class* yang berbeda. Kekuatan *diagram activity* adalah merepresentasikan *concurrent activity*.

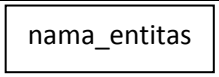
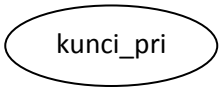
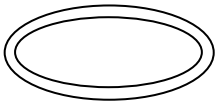
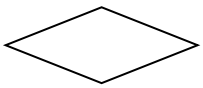
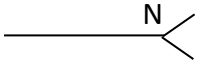
Menggambarkan rangkaian aliran dari aktivitas, digunakan untuk mendeskripsikan aktivitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktivitas lainnya seperti *use case* atau interaksi.

5. *Entity Relationship Diagram*

Menurut Sukamto dan Shalahuddin (2013:50), ERD (*Entity Relationship Diagram)* adalah pemodelan awal basis data yang paling banyak digunakan.

ERD memiliki beberapa aliran-aliran notasi seperti, notasi *Chen*, notasi *Barker*, notasi *Crow's Foot*, dan lainnya. Namun yang paling banyak digunakan adalah ERD dengan notasi *Chen*. Tabel 4.4. memperlihatkan simbol-simbol pada *Entity Relationship Diagram.*

Tabel 4.4. Simbol *Entity Relationship Diagram*

Simbol	Keterangan
	**Entitas,** data inti yang akan disimpan; benda yang memiliki data dan harus disimpan datanya agar dapat diakses oleh aplikasi komputer.
	**Atribut**, field atau kolom data yang butuh disimpan dalam suatu entitas.
	**Atribut Kunci Primer**, kolom data yang butuh disimpan dalam suatu entitas dan digunakan sebagai kunci akses *record* yang diinginkan.
	**Atribut Multinilai**, field atau kolom data yang butuh disimpan dalam suatu entitas yang dapat memiliki nilai lebih dari satu.
	**Relasi**, melambangkan antar entitas; biasanya diawali dengan kata kerja.
	**Asosiasi**, penghubung antar relasi dan entitas di mana di kedua ujungnya memiliki *multiplicity* kemungkinan jumlah pemakaian.

Sumber : Sukamto dan Shalahuddin (2013:50)

4.4.2. Metode Pengembangan Sistem

Metode pengembangan sistem penelitian menggunakan metode p*rototype.* Pengembangan pada metode ini dilakukan berdasarkan tujuh tahap, yaitu pengumpulan bahan, membangun *prototyping*, Evaluasi *prototyping*, mengkodekan *system*, menguji *system*, evaluasi *system*, dan menggunakan *system*. Berikut penjelasannya :

1. Pengumpulan kebutuhan

   Pada tahap ini penulis mengumpulkan data serta membahas solusi permasalahan yang ada pada PT. Dharma Mulia Buana Abadi.

2. Membangun *prototyping*

   Pada tahap ini penulis merancang dan membangun sebuah sistem sementara yang dibutuhkan oleh PT. Dharma Mulia Buana Abadi, yang meliputi data *company profile* dan jasa konstruksi.

3. Evaluasi *protoptyping*

   Pada tahap ini penulis melakukan evaluasi atau pengecekan terhadap sistem sementara yang telah dibangun oleh penulis, pengecekan sistem dilakukan oleh pihak PT. Dharma Mulia Buana Abadi..

4. Mengkodekan sistem

   Pada tahap ini penulis melakukan pengkodean terhadap sistem atau program yang telah disepakati oleh pihak PT. Dharma

Mulia Buana Abadi, pengkodean sistem berfungsi untuk menjalankan program.

5. Menguji Sistem

   Pada tahap ini penulis melakukan pengujian terhadap suatu program yang telah di bangun dan telah berhasil dijalankan. Pengujian dilakukan pada *localhost* dengan menggunakan metode *blackbox testing*. Pengujian dilakukan berguna untuk menguji sebuah sistem sebelum digunakan.

6. Evaluasi Sistem

   Pada tahap ini penulis melakukan pengecekan program kembali kepada pihak PT. Dharma Mulia Buana Abadi terhadap program yang telah dirancang dan dibangun.

7. Menggunakan sistem

   Pada tahap ini, rancang bangun sistem informasi jasa konstruksi yang telah dibangun telah dapat diserahkan kepada pihak PT. Dharma Mulia Buana Abadi.