BAB III

OBJEK DAN METODE PENELITIAN

3.1 Objek Penelitian

Objek penelitian merupakan suatu permasalahan yang dijadikan sebagai topik penulisan dalam rangka menyusun suatu laporan penelitian ini dilakukan untuk mendapatkan data-data yang berkaitan dengan objek penelitian tersebut.

3.1.1 Sejarah Singkat Perusahaan

Three Cores didirikan pada Febuari 2011, berawal dari berkumpulnya tiga orang pemuda yang memiliki kegemaran bersama dalam dunia Informatika. Karena didukung dengan kemampuan dan pengalaman dalam pengembangan sistem informasi, maka 3 orang pemuda bersahabat merencanakan untuk membangun suatu lembaga Consultant IT yang mengkombinasikan suatu objek, serta mampu mengkolaborasikan berbagai aplikasi untuk mendapatkan suatu sistem yang bagus.

Saat ini Three Cores telah berdiri di kota Bandung dan telah banyak membuat berbagai sistem informasi yang telah diterapkan di berbagai perusahaan. Tenaga IT yang tergabung di Three Cores memiliki kemampuan dan pengalaman dalam bidang informatika yang memungkinkan dapat memberikan karya yang terbaik di dunia informatika.

3.1.2 Visi dan Misi Perusahaan

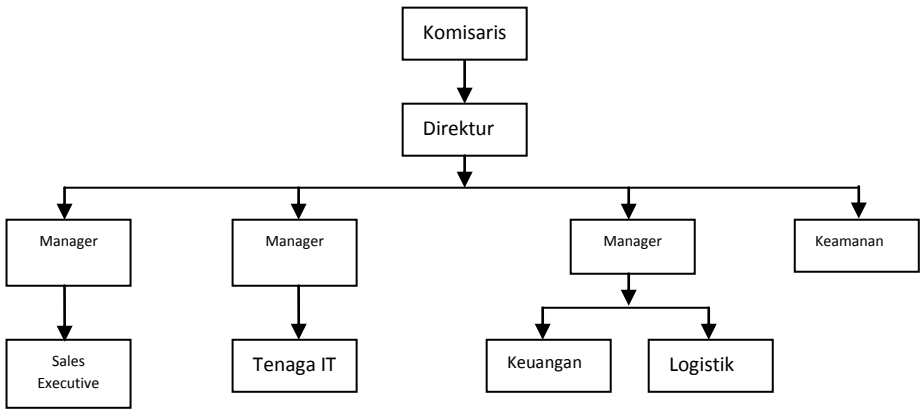
Visi

Menjadi Perusahaan konsultant terbaik dan berdedikasi tinggi serta mampu berkompetensi

Misi

1. Menyelenggarakan pelatihan praktis di bidang IT dengan teknologi terkini sehingga menciptakan tenaga – tenaga ahli di bidang informatika.
2. Melakukan penelitian dan kerja sama dengan industri, perguruan tinggi maupun lembaga-lembaga lainnya untuk menghasilkan Sistem Informasi yang baik.

3.1.3 Struktur Organisasi Perusahaan



Gambar 3.1 Struktur Organisasi

3.1.4 Deskripsi Tugas

1. Komisaris

   Tugas dan kewenangan:

   a. Melakukan pengawasan atas jalannya usaha dan memberikan nasihat kepada direktur

   b. Dalam melakukan tugas, dewan direksi berdasarkan kepada kepentingan dan sesuai dengan maksud dan tujuan perusahaan.

   c. Kewenangan khusus dewan komisaris, bahwa dewan komisaris dapat diamanatkan dalam anggaran dasar untuk melaksanakan tugas-tugas tertentu direktur, apabila direktur berhalangan atau dalam keadaan tertentu

   Kewajiban

   a. Membuat risalah rapat dewan komisaris dan menyimpan salinan rapat

   b. Melaporkan kepada perusahan mengenai kepemilikan saham dan/atau keluarga atas saham dan saham di perusahaan lainnya

   c. Memberikan laporan tentang tugas pengawasan yang telah dilakukan

2. Direktur

   a. Memimpin perusahaan dengan menerbitkan kebijakan-kebijakan perusahaan

b. Memilih, menetapkan, mengawasi tugas dari karyawan dan kepala bagian (manajer)
c. Menyetujui anggaran tahunan perusahaan
d. menyampaikan laporan kepada pemegang saham atas kinerja perusahaan

3. Manager Sales

Tugas manager sales adalah sebagai berikut :

a. Bertanggung jawab terhadap pelaksanaan marketing plan
b. Bertanggung jawab terhadap hasil peningkatan jumlah siswa
c. Bertanggung jawab terhadap sales management
d. Membina dan membimbing sales team
e. Membuat sales plan dan sales report

4. Manager Administrasi

Tugas seorang Manajer Administrasi adalah untuk koordinasi dan membantu operasional umum organisasi.

Untuk mengelola suatu proyek/acara organisasi guna menjamin audit dan dokumentasi pekerjaan tersimpan dengan baik sesuai prosedur

5. Manager HRD

Tugas dan tanggung jawab (secara umum) : mengelola, mengkoordinasi, dan menyelaraskan seluruh aktivitas yang berhubungan dengan SDM, termasuk administras.

3.2 Metode Penelitian

Suatu penelitian tidak akan berjalan dengan baik apabila tidak dilakukan dalam suatu proses yang teratur dan terarah. Oleh karena itu, diperlukan suatu metode pendekatan/penyelesaian untuk melaksanakan suatu penelitian.Metode yang digunakan oleh penulis pada penelitian ini adalah metode dengan merancang dan mendesain program berdasarkan analisis sistem yang diusulkan sesuai dengan masalah yang ada di dunia aktual (lapangan) dan sesuai dengan kebutuhan system untuk di implementasi nantinya.

3.2.1 Desain Penelitian

Desain penelitian adalah proses perencanaan yang dilakukan sebelum melaksanakan penelitian. Penulis memilih jenis desain penelitian dengan data primer/sekunder, karena jika peneliti membutuhkan data primer maka peneliti dapat langsung observasi ke tempat penelitian sedangkan jika membutuhkan data sekunder maka peneliti dapat mengevaluasi kembali data-data primer yang didapat dari tempat penelitian tersebut.

3.2.2 Jenis dan Metode Pengumpulan Data

Untuk memperoleh data dan keterangan yang diperlukan, dalam hal ini penulis menggunakan cara sebagai berikut :

3.2.2.1 Sumber Data Primer

Data primer yaitu data yang diperoleh langsung dari sumber yang akan diteliti, dalam hal ini adalah manager pada Three Cores. Untuk memperoleh data primer dapat dilakukan hal sebagai berikut :

Studi Lapangan yaitu penelitian dengan cara :

a. Observasi, yaitu teknik pengumpulan data dan informasi dengan cara pengamatan secara langsung terhadap operasional perusahaan.
b. Wawancara, yaitu teknik pengumpulan data dengan cara tanya jawab secara langsung dengan para karyawan yang berhubungan dengan penelitian ini. Wawancara dilakukan dilakukan dengan manager yang dapat memberikan data yang relevan dengan masalah yang diteliti oleh penulis.

3.2.2.2 Sumber Data Sekunder

Data sekunder merupakan cara pengumpulan data dengan cara mempelajari data yang telah tersedia atau dikumpulkan terlebih dahulu oleh pihak lain seperti buku-buku.

Cara yang digunakan untuk mengumpulkan data sekunder adalah metode Dokumentasi, yaitu dengan mengumpulkan data dan informasi yang diperlukan dari sumber-sumber kebanyakan dari materi sejenis dokumen yang berkenaan dengan masalah yang diteliti. Metode ini digunakan untuk pengumpulan data yang berhubungan dengan sejarah, visi dan misi, kegiatan dan struktur organisasi.

3.2.3 Metode Pendekatan dan Pengembangan Sistem

Adapun metode pendekatan dan pengembangan sistem yang digunakan oleh penulis adalah sebagai berikut :

3.2.3.1 Metode Pendekatan Sistem

Metode pendekatan yang digunakan oleh penulis adalah dengan menggunakan pendekatan sistem berorientasi objek .Adapun alat-alat yang digunakan untuk melakukan pendeketan dalam analisis dan pemograman berorientasi objek yaitu dengan notasi UML dengan membuat tujuah diagram yaitu, *Use case diagram, Activity Diagram, Sequence diagram, Collaboration diagram, Class diagram, Component diagram, Deployment diagram.*

1. *Use Case*

   Use Case adalah untuk mendeskripsikan bagaiman sistem akan terlihat dimata user. *Use case* terdiri dari sekumpulan *scenario* yang di lakukan oleh *actor*, sedangkan *use case diagram* memfasilitasi komunikasi diantara analis dan pengguna. Didalam *use case* terdapat <*include*>yang berfungsi untuk menyatakan sebuah *use case* adalah bagian dari *use case* lain. <*extend*>digunakan untuk membuat *use case* baru dengan menambahkan langkah-langkah pada *use case* yang sudah ada.

2. *Sequence Diagram*

   Sequence diagram menggambarkan perilaku pada sebuah *scenario*. Program ini menunjukka sejumlah objek dan *message* (pesan) yang diletakkan diantara objek-objek didalam *use casesequence diagram* terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang

terkait). *Sequence diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan *output* tertentu.

3. *Activity Diagram*

   Activity diagram adalah sebuah teknik untuk mendeskripsikan sebuah logika procedural proses bisnis dan aliran kerja dalam banyak kasus. *Activity diagram* mempunyai peran seperti halnya *flowchart*akan tetapi perbedaan dengan *flowchart* adalah *activity diagram* bisa mendukung perilaku parallel, sedangkan *flowchart* tidak bisa.*Activity diagram* bisa digunakan untuk menunjukkan siapa yang mengerjakan apa dengan teknik partision.

4. *Collaboration Diagram*

   Collaboration diagram adalah bentuk lain dari *sequence diagram*. Bila *sequence* diorganisir menurut waktu, maka *collaboration* diorganisir menurut ruang/*spase. Collaboration diagram* merupakan asosiasi diantara objek parallel didekat gasis asosiasi yang menunjukkan *message,* sedangkan konten *message* ditunjukkan dengan label. *Collaboration diagram* memungkinkan untuk memodelkan pengiriman sebuah *message* kebanyak objek pada *class* yang sama.Demikian juga halnya untuk menunjukkan adanya objek aktif yang mengendalikan aliran *message.*

5. *Class Diagram*

   Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain

berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi).*Class diagram* menggambarkan struktur dan deskripsi *class, package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.

6. *Component Diaram*

   Component diagram merepresentasikan dunia *real* item yaitu *component software*.*Component diagram* menetap dikomputer bukan dibenak analis.*Component* bisa diakses melalui *interfacenya* yaitu operasi-operasi relasi antar *component* dan *interfacenya* disebut *realization.* Suatu komponen mengakses s*ervices-service* yang ada di *component* dengan cara *import interface.* Sedangkan *component* yang menyediakan *service* menggunakan *export interface.*

7. *Deployment Diagram*

   Deployment/physical diagram menggambarkan detail bagaimana komponen di-*deploy* dalam infrastruktur sistem, di mana komponen akan terletak (pada mesin, server atau piranti keras apa), bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi server, dan hal-hal lain yang bersifat fisikal. Sebuah *node* adalah server, *workstation*, atau piranti keras lain yang digunakan untuk men-*deploy* komponen dalam lingkungan sebenarnya. Hubungan antar *node* (misalnya TCP/IP) dan *requirement* dapat juga didefinisikan dalam diagram ini.

Menurut Bambang Hariyanto (2004 : 86) keunggulan pendekatan berorientasi objek adalah sebagai berikut :

1. Bekerja yang mendekati kondisi manusia.
2. Menghasilkan sistem yang dibangun di atas bentuk-bentuk antara yang stabil dan dengan demikian lebih mampu untuk mengikuti perubahan.
3. Dapat digunakan tidak hanya pada perancangan perangkat lunak tapi juga seluruh proses pengembangan perangkat lunak.
4. Mereduksi resiko pengembangan sistem-sistem kompleks terutama karena pemaduan terjadi menyebar pada siklus kehidupan tidak terjadi dalam sekejap sebagaimana kejadian ledakan besar (*big bang*).
5. Pendekatan ini membantu mengeksploitasi keampuhan bahasa pemrograman berbasis objek.

Pendekatan berorientasi objek adalah salah satu faktor untuk menuntun menuju ke solusi perangkat lunak, untuk itu kadang harus menyeimbangkan dengan pertimbangan-pertimbangan pendekatan yang lain. Menurut Booch, prinsip-prinsip yang mendasari pengembangan berorientasi objek adalah :

1. Abstraksi
2. Pengkapsulan
3. Modularitas
4. Hirarki
5. *Typing*
6. Kongkurensi
7. *Persistence*

Prinsip-prinsip ini bekerja/diterapkan secara sinergi (bersama-sama bekerja sama). Prinsip-prinsip tersebut merupakan prinsip-prinsip utama, dimana keempat prinsip pertama adalah wajib sementara tiga prinsip berikutnya adalah *optional* (boleh digunakan ataupun tidak).

3.2.3.2 Metode Pengembangan Sistem

Metode yang digunakan dalam pengembangan untuk membangun sistem informasi ini yaitu metode *prototype* yang dimana *prototype* merupakan suatu metode dalam pengembangan sistem yang menggunakan pendekatan untuk membuat suatu program dengan cepat dan bertahap dan *prototype* juga membuat proses pengembangan sistem informasi menjadi lebih baik cepat dan lebih mudah. Dimana tahapan-tahapan yang harus dilaksanakan adalah sebagai berikut:

1. Identifikasi kebutuhan sistem

Pada tahap ini merupakan tahap awal dalam membangun sebuah sistem informasi, dimana antara pemakai sistem (*user*) dan pengembang sistem bertemu. User dijelaskan tentang kebutuhan sistem yang akan dibangun oleh pengembang sistem.

2. Membuat *prototype*

Setelah menganalisa sistem yang akan dibangun, pengembangan sistem mulai membuat *prototype*.

3. Menguji *prototype*

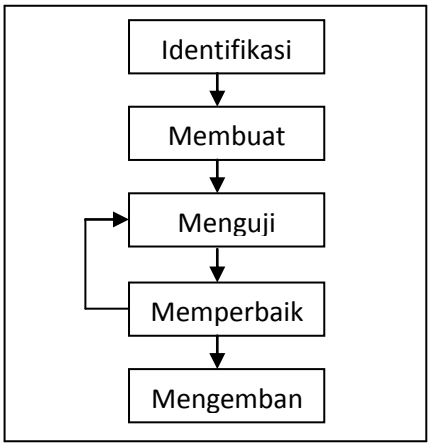
Setelah tahap ini pembuatan *prototype* selesai, kemudian pengembang sistem dan *user* melakukan pengujian program agar program dapat digunakan sesuai dengan kebutuhan, dan *user* memberikan saran atau masukan bila terdapat kekurangan pada program.

4. Memperbaiki *prototype*

Pada tahap ini pengembang sistem melakukan perbaikan dan modifikasi sesuai dengan masukan dan saran dari *user*.

5. Mengembangkan versi Produksi

Pada tahap ini pengembang sistem menyelesaikan sistem yang telah dibuatnya sesuai masukan atau saran terakhir dari pemakai sistem. Dibawah ini adalah tahapan pendekatan *prototype* yang ditujukkan pada gambar sebagai berikut:



Gambar 3.2 Tahapan Pendekatan *Prototype*

(Sumber : Susanto azhar. 2004. *Sistem Informasi Manajemen, konsep dan pengembangan.* lingga jaya. Bandung**)**

Penulis mengambil model proses secara *prototyping* karena :

a. Penyajian aspek-aspek perangkat lunak yang akan dibangun akan nampak bagi pemakai secara cepat (contohnya pendekatan input dan format output). Perancangan kilat membawa kepada konstruksi sebuah *prototype*.
b. *Prototype* dievaluasi oleh kedua belah pihak (*programmer* dan calon *user*) sehingga penyaringan kebutuhan pengembangan perangkat lunak dapat dengan cepat dilakukan sesuai dengan keinginan *user*.
c. *Programmer* akan memahami apa yang harus dilakukan agar kebutuhan *user* dapat dipenuhi.

3.2.3.3 Alat Bantu Analisis dan Perancangan

Model data berorientasi objek dikatakan dapat memberi fleksibilitas yang lebih, kemudahan mengubah program dan digunakan luas dalam teknik piranti lunak skala besar.Lebih jauh lagi pendukung OOP (*Object Oriented Programming*) mengklaim bahwa OOP lebih mudah dipelajari bagi pemula dibanding dengan pendekatan sebelumnya dan pendekatan OOP lebih mudah dikembangkan dan dirawat. Dalam UML (*Unified Modelling Language*) yang dapat dikelompokan berdasarkan sifatnya yaitu statis atau dinamis jenis diagramnya adalah sebagai berikut :

1. Diagram Kelas (*Class Diagram*) bersifat statis. Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-

kolaborasi serta relasi-relasi. Diagram ini umum dijumpai pada pemodelan berorientsi objek.

2. Diagram Objek (*ObjectDiagram*) bersifat statis. Diagram ini memperlihatkan objek-objek serta relasi-relasi antar objek. Diagram objek memperlihatkan instansiasi statis dari segala sesuatu yang dijumpai didalam diagram kelas.

1. Diagram *Use-Case* (*Use-CaseDiagram*) bersifat statis. Diagram ini memperlihatkan himpunan *use case* dan aktor-aktor (suatu jenis khusus dari kelas). Diagram ini sangat penting untuk mengorganisasi dan memodelkan perilaku dari suatu sistem yang dibutuhkan serta diharapkan pengguna.
2. Diagram Sekuen (*SequenceDiagram*) bersifat dinamis. Diagram urutan adalah diagram interaksi yang menekankan pada pengiriman pesan (message) dalam suatu waktu tertentu.
3. Diagram Kolaborasi (*CollaborationDiagram*) bersifat dinamis. Diagram berkolaborasi adalah menggambarkan interaksi antar objek seperti sequence diagram, tetapi lebih menekankan pada peran masing-masing objek dan bukan pada waktu penyampaian pesan *(message).*
6. Diagram*Statechart* (*Statechart diagram*) bersifat dinamis. Diagram *state* ini memperlihatkan state–state pada sistem, memuat sstate, transisi, *event*, serta aktifitas. Diagram ini penting untuk memperlihatkan sifat dinamis dari antar muka , kelas, kolaborasi, dan terutama penting pada pemodelan sistem–sistem yang reaktif.

7. Diagram aktivitas (*ActivityDiagram*) bersifat dinamis. Diagram ini adalah diagram tipe khusus dari digram state yang memperlihatkan aliran dari suatu aktifitas ke aktifitas yan lainnya dalam suatu sistem. Diagram ini penting dalam pemodelan fungsi-fungsi dalam satu sistem dan memberi tekanan pada aliran kendali antar objek.
8. Diagram Komponen (*ComponenDiagram*) bersifat statis. Diagram ini memperlihatkan organisasi serta ketergantungan system atauperangkat lunak pada komponen-komponen yang telah ada sebelumnya. Diagram ini berhubungan dengan diagram kelas dimana komponen secara tifikal dipetakan kedalam satu atau lebih kelas-kelas.
9. Diagram Penyebaran (*DeploymentDiagram*) bersifat statis. Diagram ini memperlihatkan konfigurasi saat aplikasi dijalankan. Diagram ini memeuat simpul-simpul beserta komponen-

   komponen yang ada didalamnya. *Deployment diagram* berhubungan denga n diagram komponen.dimana*deployment diagram* memuat satu atau lebih komponen-komponen.

3.2.4 Pengujian software

Pengujian perangkat lunak (*software*) menggunakan metode pengujian *Black Box*. Metode ujicoba *Black Box* memfokuskan pada keperluan fungsional dari *software*. Ujicoba *Black Box* berusaha untuk menemukan kesalahan dalam beberapa kategori, diantaranya :

1. Fungsi-fungsi yang salah atau hilang
2. Kesalahan *interface*
3. Kesalahan dalam struktur data atau akses database eksternal
4. Kesalahan performa
5. Kesalahan inisialisasi dan terminasi

Ujicoba didesain untuk dapat menjawab pertanyaan-pertanyaan berikut :

1. Bagaimana validitas fungsionalnya diuji?
2. Jenis input seperti apa yang akan menghasilkan kasus uji yang baik ?
3. Apakah sistem secara khusus sensitif terhadap nilai input tertentu ?
4. Bagaimana batasan-batasan kelas data diisolasi?
5. Berapa rasio data dan jumlah data yang dapat ditoleransi oleh sistem?
6. Apa akibat yang akan timbul dari kombinasi spesifik data pada operasi sistem?

Dengan mengaplikasikan ujicoba *Black Box*, diharapkan dapat menghasilkan sekumpulan kasus uji yang memenuhi kriteria berikut:

1. Kasus uji yang berkurang, jika jumlahnya lebih dari 1, maka jumlah dari ujikasus tambahan harus didesain untuk mencapai ujicoba yang cukup beralasan.
2. Kasus uji yang memberitahukan sesuatu tentang keberadaan atau tidaknya suatu jenis kesalahan, daripada kesalahan yang terhubung hanya dengan suatu ujicoba yang spesifik.