BAB III

OBJEK DAN METODE PENELITIAN

3.1 Objek Penelitian

Dalam penelitian ini, yang menjadi objek penelitian adalah CV. CITRA RETROFITA BANDUNG.

3.1.1 Sejarah Singkat Perusahaan

CV. CITRA RETROFITA BANDUNG sebagai perusahaan yang bergerak dalam bidang Perbaikan Beton, Penguatan Srtruktur Tanah dan Waterproofing berupaya untuk memberikan solusi dalam mengantisipasi kendala-kendala kontruksi, serta memberikan berbagai alternatif pelaksanaan kontruksi yang berorientasi pada peningkatan kualitas, penghematan waktu dan efesiensi biaya. Perusahaan ini berdiri pada hari senin tanggal 13 april 1998 yang terletak Jl. Mars Barat IV No. 4 Margahayu Raya – Bandung.

Perusahaan ini memusat kan pada kegiatan kerja pada:

a. Jasa konsultan
b. Supplier
c. Peningkatan Mutu Beton
d. Coring, Drilling & Anchoring
e. Perbaikan Beton
f. Perkuatan Struktur Tanah
g. Aplicator Product Chemical

Metoda dan produk yang disajikan CV. CITRA RETROFITA BANDUNG didasarkan atas hasil uji Laboratorium LITBANG dan berbagai pengalaman yang pernah dilaksanakan dilapangan, dengan dukungan dari tenaga ahli produk dan tenaga aplikasi yang berpengalaman.

Terciptanya hasil kerja yang memuaskan tidak terlepas dari sistem kerjasama yang baik antara Main Kontraktor dan Owner dengan CV. CITRA RETROFITA BANDUNG.

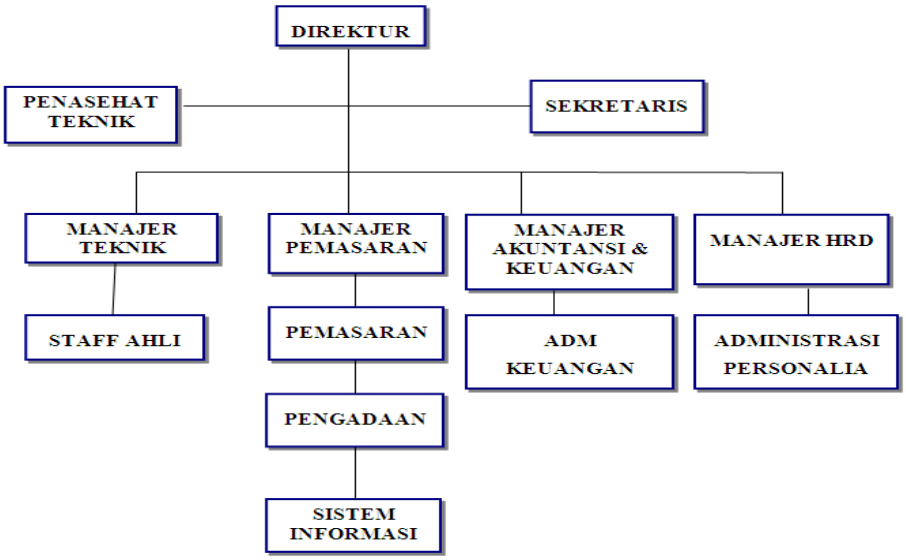
3.1.2 Visi dan Misi Perusahaan

Visi CV.CIRTA RETROFITA BANDUNG adalah sebagai pelaksana konstruksi bangunan yang Profesional, dan Amanah.

Adapun Misi CV.CIRTA RETROFITA BANDUNG adalah:

1. Membantu pemerintah mengatasi penganguran sekaligus membantu pembinaan di bidang agama.
2. Memberikan solusi dalam mengatasi kendala konstruksi.
3. Memberikan berbagai alternatif pelaksanaan kontruksi yang berorientasi pada peningkatan kualitas, penghematan waktu dan efesiensi biaya.

3.1.3 Struktur Organisasi Perusahaan



Gambar 3.1 Struktur Organisasi CV. CITRA RETROFITA BANDUNG

3.1.4 Deskripsi Tugas

Berikut adalah deskripsi tugas dari bagian-bagian yang terlibat didalam sistem sebagai berikut:

1. Direktur merumuskan dan menetapkan kebijakan-kebijakan perusahaan dengan mengkoordinasikan, mengarahkan dan mengendalikan kegiatan-kegiatan perusahaan untuk mencapai tujuan.
2. Penasehat Teknik melaksanakan dan mengendalikan, mengarahkan dan mengawasi kinerja manejer teknik.
3. Sekretaris mempunyai tugas membantu kerja direktur.
4. Manajer teknik menyusun dan merumuskan kebijakan pengolahan bahan baku, pengelolaan kegitan – kegiatan kerja di lapangan.

5. Manajer pemasaran Mengkoordinir kegiatan pemasaran secara keseluruhan sehingga menjadi meningkat.
6. Manajer akuntansi dan keuangan mempunyai tugas membuat laporan keuangan perusahaan.
7. Administrasi keungan membuat administrasi keuangan dan pengelompokkan biaya.
8. Manajer HRD (*Human Research Development)* mempunyai tugas membuat kebijakan – kebijakan yang menyangkut pegawai.
9. Administrasi personalia mempunyai tugas membuat struktur kerja pegawai.

3.2 Metode Penelitian

Metode penelitian yang akan dilakukan dalam penelitian ini adalah metode yang menghasilkan produk juga melibatkan individu sebagai objek penelitian.

3.2.1 Desain Penelitian

Desain penelitian ini termasuk kedalam penelitian yang bersifat deskriptif. Penelitian deskriptif adalah penelitian yang bertujuan untuk memperoleh ciri-ciri variabel, dimana dalam penelitian ini adalah untuk memperoleh gambaran tentang kinerja program dirancang dan diimplementasikan kepada pengguna (*user*) pada CV.CITRA RETROFITA BANDUNG.

3.2.2 Jenis dan Metode Pengumpulan Data

Dalam pengembangan sistem informasi tentunya diperlukan jenis data yang akurat sesuai dengan sistem yang akan dikembangkan, oleh karena itu

dibutuhkan suatu teknik pengumpulan data yang tepat. Jenis data yang digunakan penulis dalam tahap analisa dan tahap perancangan adalah sebagai berikut:

3.2.2.1 Sumber Data Primer (Wawancara, Observasi, Kuesioner)

Sumber data primer, yaitu data yang diperoleh langsung dari sumbernya, diamati dan dicatat untuk pertama kalinya. Data diperoleh secara langsung dalam kegiatan yang berlangsung pada cv.citra retrofita bandung.

Adapun metode penelitian yang dilakukan oleh penulis untuk pengumpulan data dan informasi berupa :

1. *Observasi*

Metode ini merupakan teknik pengumpulan data yang dilakukan dengan mengamati secara langsung dan ikut serta kedalam kegiatan yang berlangsung pada cv.citra retrofita bandung khususnya bagian HRD dan akuntansi dan keuangan guna memperoleh gambaran yang berhubungan dengan sistem informasi penggajian pegawai.

2. Wawancara (*Interview*)

Dalam hal ini penulis melakukan tanya jawab secara langsung mengenai permasalahan yang akan diteliti kepada pihak yang bersangkutan di bagian manager HRD dan manager akuntansi dan keuangan yaitu mengenai hal-hal yang menyangkut pendataan data pegawai tetap dan kontrak, pendataan jabatan, pendataan absensi , penghitungan potongan dari absensi, dan perhitungan gai pegawai tetap dan ontrak. Wawancara ini bertujuan untuk memperjelas dan meyakinkan atas fakta atau informasi yang diperoleh melalui observasi.

3.2.2.2 Sumber Data Sekunder (dokumentasi)

Sumber data sekunder, yaitu data yang diperoleh dari sumber lain seperti catatan perusahaan, arsip-arsip, dokumen-dokumen, file-file yang ada. Penulis mengumpulkan dokumen-dokumen yang dibutuhkan dalam proses analisis dan desain pengembangan sistem yang akan dibuat. Studi Dokumentasi, melakukan pengumpulan data yang berkaitan dengan masalah penggajian pegawai dari dokumen yang diberikan oleh perusahaan berupa dokumentasi form untuk data gaji pegawai, laporan pendataan pegawai, beserta absensinya.

3.2.3 Metode Pendekatan dan Pengembangan Sistem

Adapun metode pendekatan dan pengembangan sistem akan dijelaskan seperti dibawah ini antara lain yaitu:

3.2.3.1 Metode Pendekatan Sistem

Metode pendekatan yang digunakan adalah metode pendekatan berorientasi objek (*Object Oriented).* Menurut Adi Nugroho (2005:4) Pendekatan berorientasi objek merupakan cara berfikir baru serta berlogika dalam menghadapi masalah-masalah yang akan di atasi dengan bantuan komputer. OOP mencoba mencoba melihat permasalahan lewat pengamatan dunia nyata dimana setiap objek adalah entitas tunggal yang memiliki kombinasi struktur data dan fungsi tertentu. Pendekatan berorientasi objek terdiri dari analisis berorientasi objek *(OOA)* dan desain berorientasi objek *(OOD).*

Analisis berorirentasi objek *(OOA)* dimulai dengan menyatakan suatu masalah, analisis membuat suatu model situasi dari dunia nyata, menggambarkan sifat yang

penting. Sedangkan Desain berorientasi objek (*OOD)* merupakan tahap lanjutan setelah *(OOA),* dimana tujuan sistem diorganisasikan kedalam sub-sistem berdasarkan struktur analisis dan arsitektur yang dibutuhkan.

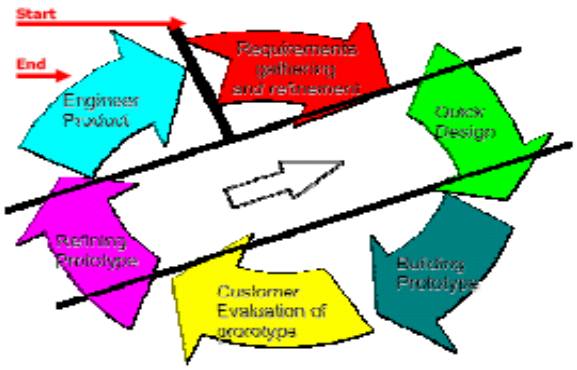
3.2.3.2 Metode Pengembangan Sistem

Adapun metode pengembangan yang digunakan adalah prototype, dalam buku Kendal (2003:221) prototyping sistem informasi yaitu suatu teknik yang sangat berguna untuk mengumpulkan informasi tertentu mengenai syarat-syarat informasi pengguna secara cepat. Prototyping yang efektif seharusnya dilakukan pada awal-awal siklus pengembangan sistem, yakni selama fase penetapan syarat-syarat. Prototyping adalah suatu teknik yang sangat rumit karena memerlukan pengetahuan mengenai siklus hidup pengembangan sistem secara keseluruhan.

Sistem prototype merupakan bagian operasional dari sistem secara keseluruhan yang akan dibangun. *Prototype* merupakan suatu cara yang baik untuk mendapatkan umpan balik mengenai sistem yang diajukan dan mengenai bagaimana sistem tersebut tersedia untuk memenuhi kebutuhan informasi pengguna.

Ada empat petunjuk yang harus diamati saat mengintegrasikan prototype kedalam fase penetapan siklus hidup pengembangan sistem, yakni:

1. Bekerja sesuai modul
2. Membangun *prototype* dengan cepat
3. Memodifikasi *prototype* dengan iterasi yang berurutan
4. Menekankan antar muka pengguna.



Gambar 3.2 Model Prototype

Sumber : blog.unsri.ac.id/userfiles/09071003013.doc

Tahapan-tahapan dalam *Prototype* adalah sebagai berikut:

1. Pengumpulan kebutuhan

Pelanggan dan pengembang bersama-sama mendefinisikan format seluruh perangkat lunak, mengidentifikasikan semua kebutuhan, dan garis besar sistem yang akan dibuat.

2. Membangun *prototyping*

Membangun *prototyping* dengan membuat perancangan sementara yang berfokus pada penyajian kepada pelanggan (misalnya dengan membuat input dan format output).

3. Evaluasi *protoptyping*

Evaluasi ini dilakukan oleh pelanggan apakah *prototyping* yang sudah dibangun sudah sesuai dengan keinginann pelanggan. Jika sudah sesuai maka langkah 4 akan diambil. Jika tidak *prototyping* direvisi dengan mengulangu langkah 1, 2 , dan 3.

4. Mengkodekan sistem

Dalam tahap ini *prototyping* yang sudah di sepakati diterjemahkan ke dalam bahasa pemrograman yang sesuai.

5. Menguji sistem

Setelah sistem sudah menjadi suatu perangkat lunak yang siap pakai, harus dites dahulu sebelum digunakan. Pengujian ini dilakukan dengan pengujian Black Box.

6. Evaluasi Sistem

Pelanggan mengevaluasi apakah sistem yang sudah jadi sudah sesuai dengan yang diharapkan . Jika ya, langkah 7 dilakukan; jika tidak, ulangi langkah 4 dan 5.

7. Menggunakan sistem

Perangkat lunak yang telah diuji dan diterima pelanggan siap untuk digunakan.

3.2.3.3 Alat Bantu Analisis dan Perancangan

Alat bantu yang digunakan analisis dan perancangan sisitem yaitu menggunakan *Unified Modeling Language* (UML). UML adalah bahasa spesifikasi standar untuk mendokumentasikan, menspesifikasikan, dan membangun sistem perangkat lunak. UML tidak berdasarkan pada bahasa pemrograman tertentu.

UML mendeskripsikan OOP (*Object Oriented Programming*) dengan beberapa diagram, diantaranya:

1. *Use case diagram*

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem.

2. *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.

3. *Statechart diagram*

Statechart diagram menggambarkan transisi dan perubahan keadaan (dari satu state ke state lainnya) suatu objek pada sistem sebagai akibat dari stimuli yang diterima. Pada umumnya statechart diagram menggambarkan *class* tertentu (satu *class* dapat memiliki lebih dari satu statechart diagram).

4. *Activity diagram*

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

Activity diagram merupakan state diagram khusus, di mana sebagian besar state adalah action dan sebagian besar transisi di-trigger oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity* diagram tidak menggambarkan behaviour internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

5. *Sequence diagram*

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence* 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. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara *internal* dan *output* apa yang dihasilkan.

6. *Collaboration diagram*

Collaboration diagram juga menggambarkan interaksi antar objek seperti *sequence diagram*, tetapi lebih menekankan pada peran masing-masing objek dan bukan pada waktu penyampaian *message*. Setiap *message* memiliki *sequence number*, di mana *message* dari *level* tertinggi memiliki nomor 1. *Messages* dari *level* yang sama memiliki *prefiks* yang sama.

7. *Component diagram*

Component diagram menggambarkan struktur dan hubungan antar komponen piranti lunak, termasuk ketergantungan (*dependency*) di antaranya.

Komponen piranti lunak adalah modul berisi code, baik berisi *source code* maupun *binary code*, baik library maupun *executable*, baik yang muncul pada compile time, link time, maupun run time. Umumnya komponen terbentuk dari beberapa *class* dan/atau *package*, tapi dapat juga dari komponen-komponen yang lebih kecil.

Komponen dapat juga berupa interface, yaitu kumpulan layanan yang disediakan sebuah komponen untuk komponen lain.

8. *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.

3.2.4 Pengujian Software

Pengujian *Software* adalah cara atau teknik untuk menguji perangkat lunak, mempunyai mekanisme untuk menentukan data uji yang dapat menguji perangkat lunak secara lengkap dan mempunyai kemungkinan tinggi untuk menemukan kesalahan.

Pengujian *Black-box* berfokus pada struktur tampilan kontrol program. Test case dilakukan untuk memastikan bahwa semua statement pada program telah dieksekusi paling tidak satu kali selama pengujian dan bahwa semua kondisi logis telah diuji.

Pengujian *Black-box* berfokus kepada persyaratan fungsional perangkat lunak. Pengujian *Black-box* memungkinkan perangkat lunak mendapatkan serangkaian kondisi input yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program. Pengujian *Black-box* bukan merupakan alternatif dari teknik *white-box*, tetapi merupakan pendekatan komplementer yang

kemungkinan besar mampu mengungkap kesalahan-kesalahan pada metode *white-box*.