BAB III

OBJEK DAN METODE PENELITIAN

3.1 Objek Penelitian

Objek penelitian dilaksanakan di Kecamatan Banjaran, Kabupaten Bandung, yang beralamat di Jl. Alun-alun Selatan No. 222 (022) 5940007, dengan alamat e-mail kec_banjaran@bandungkab.go.id.

3.1.1 Sejarah Singkat

Kecamatan Banjaran berada di sebelah timur Kota Kabupaten Bandung (Soreang) dengan jarak 6 KM dan berada di sebelah selatan Ibu Kota Provinsi Jawa Barat dengan jarak 20 KM.

Kecamatan Banjaran yang mempunyai luas wilayah 5,455,292 Ha merupakan salah satu dari 31 kecamatan yang ada di Kabupaten Bandung. Terdiri dari 11 Desa, 39 Dusun, 159 RW , 588 RT dan memiliki jumlah penduduk sebanyak 106.842 orang yang terdiri dari laki-laki 52,438 orang dan perempuan sebanyak 54,404 orang dengan jumlah KK 30,797 orang.

Kecamatan Banjaran telah berupaya memberikan pelayana kepada masyarakat dan meningkatkan pemberdayaan masyarakat melalui peningkatan kesehatan, pembinaan organisasi, *intesifikasi* dan *ekstentifikasi* sumber pendapatan daerah, sosialisai pelatihan aparatur pemerintahan desa dalam

manajemen pemerintahan desa, mengadakan pelatihan keterampilam manajemen badan usah milik desa, pembinaan kelompok masyarakat, pembangunan desa dan pelaksanaan siskamling di daerah.

3.1.2 Visi dan Misi Jangka Menengah Daerah

Rencana kerja merupakan suatu proses yang berorientasi pada hasil yang ingin dicapai dalam jangka waktu 1 tahun dengan perhitungan potensi dan peluang yang ada dan/atau mungkin timbuk dalam operasionalisasinya.

Untuk arah dan focus strategi pada tujuan yang ingin dicapai Kecamatan Banjaran mempunyai Visi :

“TERWUJUDNYA MASYARAKAT BANJARAN YANG DAMAI TERTIB DAN MAKMUR MELALUI AKSELERASI DUKUNGAN APARATUR BERKUALITAS DAN PROFESIONAL YANG BERORIENTASI PADA PENINGKATAN KINERJA PEMBANGUNAN DESA”

Guna mewujudkan dan meralisasikan Visi tersebut, Kecamatan Banjaran secara umum mempunyai Misi :

“TERWUJUDNYA PELAYANAN PRIMA PADA MASYARAKAT”

3.1.4 Deskripsi Pekerjaan

1. Camat

Mempunyai tugas pokok melaksanakan kewenangan pemerintahan yang dilimpahkan Bupati untuk menangani sebagian urusan otonomi daerah dan menyelengarakan tugas umum pemerintahan

2. Sekretasi Kecamatan

Mempunyai tugas pokok membantu camat dalam memimpin, merencanakan, melaksanakan, mengkordinasikan kegiatan pelayanan dan administrasi umum kepegawaian, keuangan, perlengkapan, kerumahtanggaan, informasi kehumasan dan ketatausahaan.

3. Sub Bagian Program

Mempunyai tugas pokok merencanakan, melaksanakan, mengevaluasi dan melaporkan pelaksanaaan tugas dan pelayanan penyusunan rencana program kerja Kecamatan.

4. Sub Bagian Umum Dan Kepegawaian

Mempunyai tugas pokok merencanakan, melaksanakan, mengevaluasi dan melaporkan pelaksanaan tugas pokok pelayanan administrasi umum, informasi kehumasan dan kepegawaian di lingkungan Kecamatan.

5. Sub Bagian Keuangan

Mempunyai tugas pokok merencanakan, melaksanakan, mengevaluasi dan melaporkan pelaksanaan tugas pengolahan administrasi dan prtanggungjawaban pengelolaan keuangan Kecamatan.

6. Seksi Pemerintahan

Mempunyai tugas pokok membantu camat dalam menyiapkan bahan rumusan kebijakan dan pelaksanaan penyelenggaraan pemerintahan yang meliputi pengembangan otonomi daerah, politik dalam negeri dan administasi publik kependudukan, hukum dan perundang-undangan.

7. Seksi Ketentraman dan Ketertiban Umum

Mempunyai tugas pokok membantu camat dalam menyelenggarakan bahan rumusan kebijakan dan pelaksanaan tugas camat dalam bidang pelayanan dan pengedalian ketentraman dan ketertiban umum.

8. Seksi Sosial Budaya

Mempunyai tugas pokok membantu camat dalam menyiapkan bahan rumusan kebijakan dan pelaksanaan tugas camat dalam bidang pelayanan keserjahteraan social dan kebudayaan.

9. Seksi Pemberdayaan Masyarakat

Mempunyai tugas pokok membantu camat dalam menyiapkan bahan rumusan kebijakan dan pelaksanaan tugas camat dalam bidang pemberdayaan masyarakat.

10. Seksi Pemeliharaan Prasarana Umum

Mempunyai tugas pokok membantu camat dalam menyiapkan bahan rumusan kebijakan dan pelaksanaan tugas camat dalam bidang pengelolaan dan pemeliharaan prasarana umum.

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 ke dalam penelitian yang bersifat tindakan (*Action Reserch*), dimana dalam penelitian ini adalah untuk memperoleh gambaran tentang kinerja program dirancang dan di implementasikan kepada pengguna (*user)* di dalam suatu intansi.

3.2.2 Jenis dan Metode Pengumpulan Data

Metode pengumpulan data digunakan penulis untuk mendapatkan data sebagai bahan kajian dalam penulisan skripsi dengan tujuan membuat suatu perancangan sistem. Dalam hal ini penulis menggunakan metode pengumpulan

data berupa sumber data primer (observasi, wawancara) dan sumber data sekunder (dokumentasi).

3.2.2.1 Sumber Data Primer

Data primer merupakan pengumpulan data secara langsung dari objek yang sedang diteliti, cara yang digunakan untuk mengumpulkan data primer adalah sebagai berikut :

a. Observasi

Observasi yaitu cara untuk mendapatkan data dengan mengadakan pengamatan dan penelitian secara langsung di lapangan, dengan melihat atau mengamati secara langsung pada instansi terkait. Tujuannya untuk mempermudah penulis dalam proses pengumpulan data sehingga data-data yang didapat dapat terjamin keaslian dan keakuratannya.

b. Wawancara

Wawancara merupakan metode pengumpulan data dengan mengajukan pertanyaan-pertanyaan / *interview* secara langsung kepada bagian yang terkait tentang objek.

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, literatur internet atau artikel-artikel ilmiah yang dapat dikaji sebagai bahan rujukan dan landasan teoritis dalam pemecahan masalah.

Cara yang digunakan untuk mengumpulkan data sekunder adalah metode dokumentasi, yaitu dengan mengumpulkan data dan informasi yang diperlukan dari sumber-sumber yang ada. Biasanya dokumen yang diamati berupa sejarah instansi terkait, visi misi, struktur organisasi, dan prosedur kerja (*job description*).

3.2.3 Metode Pendekatan dan Pengembangan Sistem

Metode pendekatan sistem yang digunakan untuk membangun sistem informasi yang baru menggunakan metode *prototype* dengan pengembangan sistem yang berorientasi pada objek (*object oriented*) dengan menggunakan notasi UML sebagai alat dokumentasi dan perancangan system

3.2.3.1 Metode Pendekatan Sistem Berorientasi Objek

Metode pendekatan yang digunakan adalah metode pendekatan yang berorientasi objek, yang mencakup analisis dan desain yang disebut *OOAD* (*Object Oriented Analysis and Design*). *Object-oriented* mencakup bidang aplikasi yang sangat luas, para pengguna sistem komputer dan sistem lain yang didasarkan atas teknologi komputer merasakan efek *object-oriented* dalam bentuk meningkatnya aplikasi *software* yang mudah digunakan dan servis yang lebih *fleksibel,* yang muncul dalam berbagai bidang industri, seperti dalam perbankan, telekomunikasi, dan sebagainya. Sedangkan bagi *software enginer*, *object-*

oriented berpengaruh dalam bahasa pemrograman, metodologi rekayasa, manajemen proyek, *hardware*, dan sebagainya.

Object-oriented analysis adalah metode analisis yang memeriksa *requirements* (syarat/keperluan yang harus dipenuhi suatu sistem) dari sudut pandang kelas-kelas dan objek-objek yang ditemui dalam ruang lingkup permasalahan. metode ini mengarahkan arsitektur *software* yang didasarkan pada manipulasi objek-objek sistem atau subsistem.

3.2.3.2 Metode Pengembangan Sistem

Metode pengembangan sistem yang digunakan adalah menggunakan model *prototype*, karena dengan metode *prototype* ini pengembang dan pelanggan dapat saling berinteraksi selama proses pembuatan system, juga memungkinkan adanya komunikasi yang baik antara pengembang dan pelanggan**,** sehingga memberikan kemudahan dalam penyimpanan informasi,

Untuk mengatasi ketidakserasian antara pelanggan dan pengembang , maka harus dibutuhakan kerjasama yanga baik diantara keduanya sehingga pengembang akan mengetahui dengan benar apa yang diinginkan pelanggan dengan tidak mengesampingkan segi-segi teknis dan pelanggan akan mengetahui proses-proses dalm menyelasaikan system yang diinginkan. Sehingga implementasi mudah karena pemakai mengetahui dari awal apa yang akan diperolehnya

Menurut Gordon dan Bieman (Sommerville, 2001:164), terdapat beberapa keuntungan dalam menggunakan Metode pengembangan sistem *prototype* diantaranya :

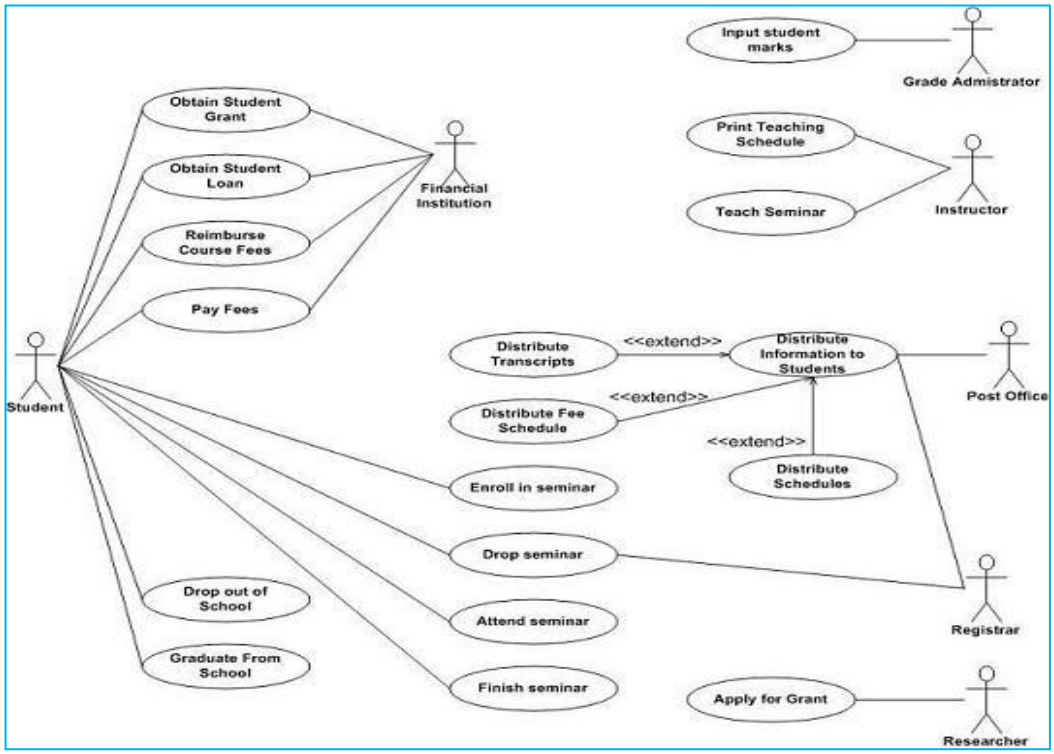
1. Kesalahpahaman antara pengembang perangkat lunak dan *user* dapat diidentifikasi ketika fungsi sistem didemonstrasikan sehingga terjadi kesesuaian sistem yang lebih dekat dengan kebutuhan *user*.
2. Kegunaan sistem dan kualitas desain yang lebih baik
3. Pemeliharaan sistem yang lebih baik
4. Usaha pengembangan yang lebih ringan
5. *Prototype* bisa langsung didemonstrasikan

3.2.3.3 Alat Bantu Analisis dan Perancangan

Adapun alat bantu yang digunakan dalam pemodelan sistem penelitian ini adalah *Unified Modeling Language* (UML). UML adalah sebuah bahasa yang berdasarkan grafik/gambar untuk memvisualisasi, menspesifikasikan, membangun, dan pendokumentasian dari sebuah sistem pengembangan software berbasis OO (Object-Oriented). UML sendiri juga memberikan standar penulisan sebuah system blue print, yang meliputi konsep bisnis proses, penulisan kelas-kelas dalam bahasa program yang spesifik, skema database, dan komponen-komponen yang diperlukan dalam sistem software

1. *Use Case Diagram*

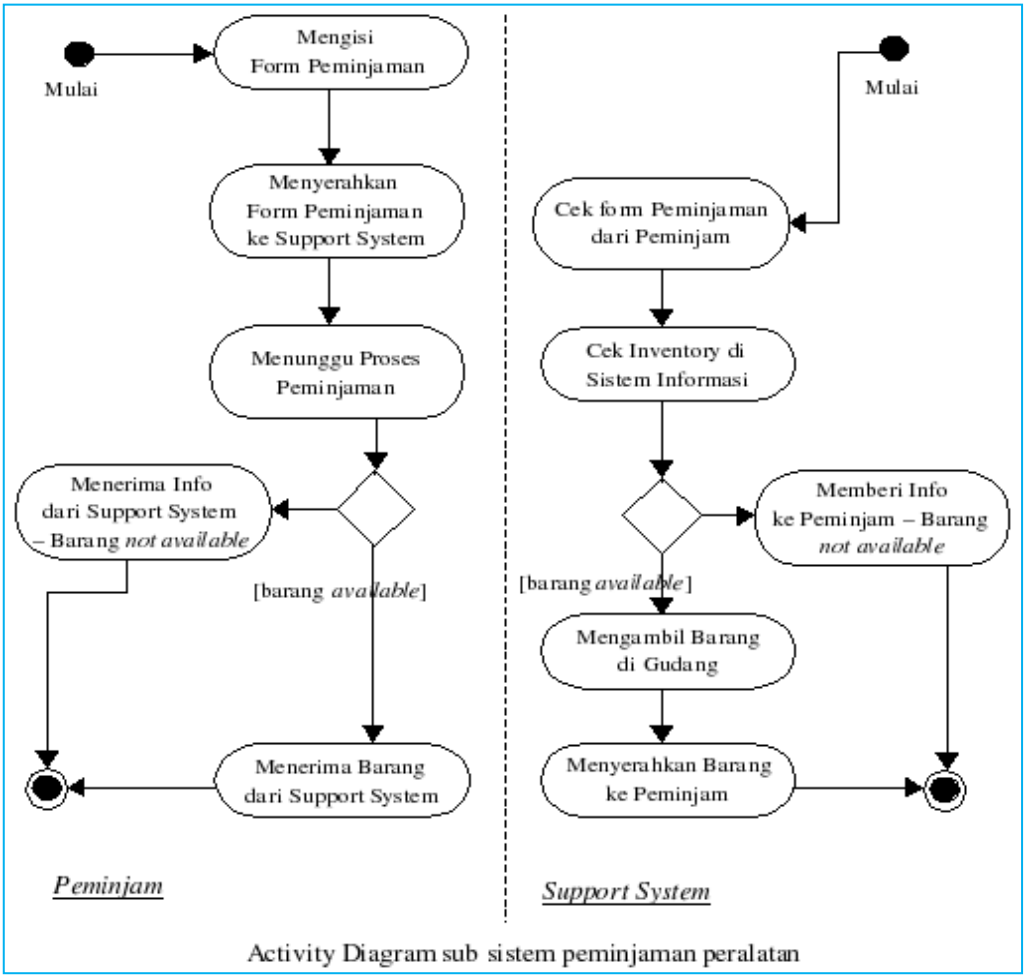
Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem.



Gambar 3.2 Contoh *Use Case Diagram*

2. *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.

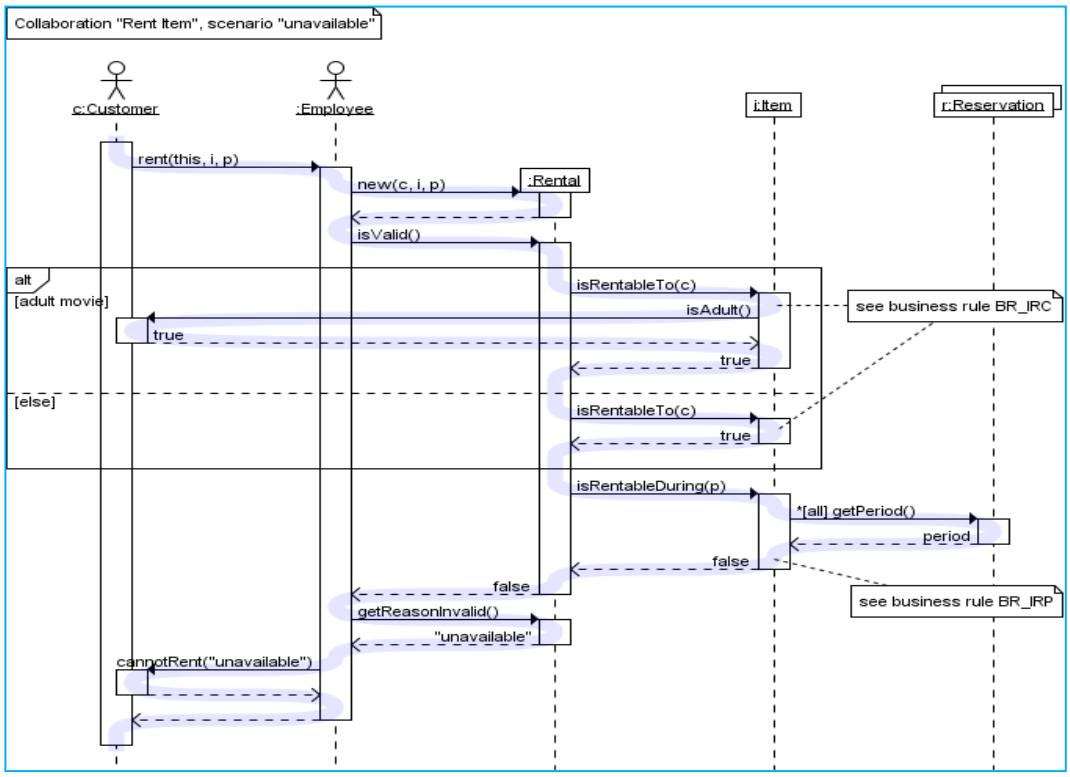


Gambar 3.3 Contoh Aktivity Diagram

3. 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



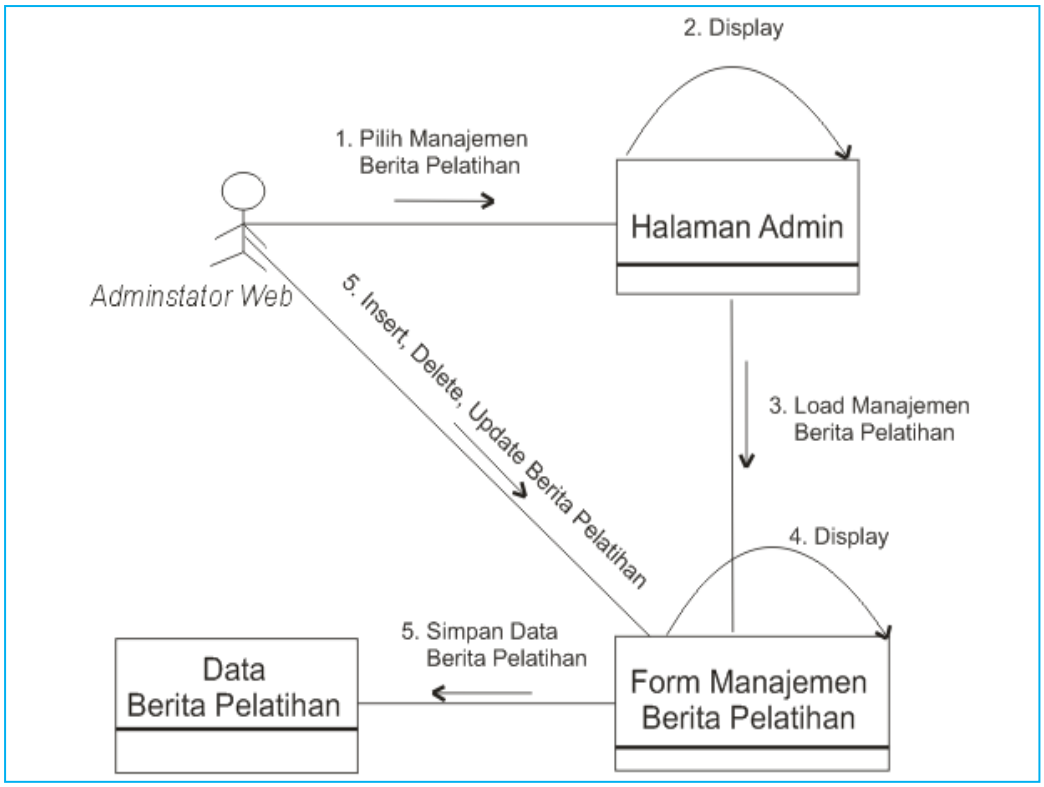
Gambar 3.4

Contoh *Sequence Diagram*

4. 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.

Contoh *Collaboration Diagram* :



Gambar 3.5

Contoh *Collaboration Diagram*

5. Class Diagram

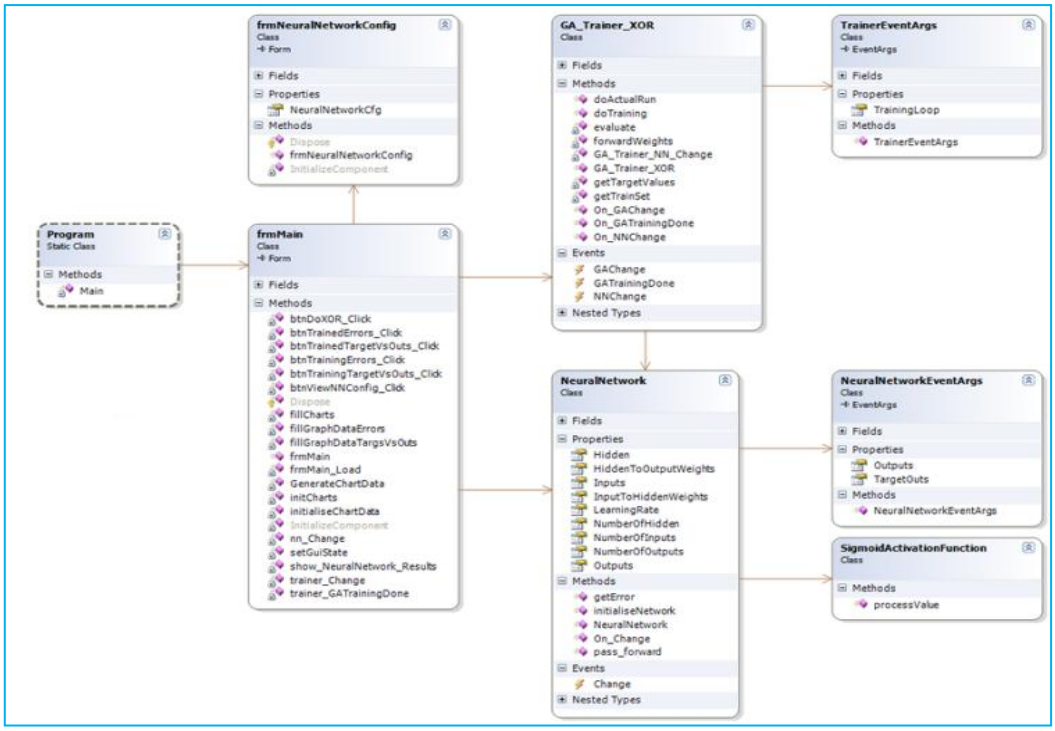
Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class diagram* menggambarkan struktur dan deskripsi class, package dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.

Class memiliki tiga area pokok :

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

Atribut dan metoda dapat memiliki salah satu sifat berikut :

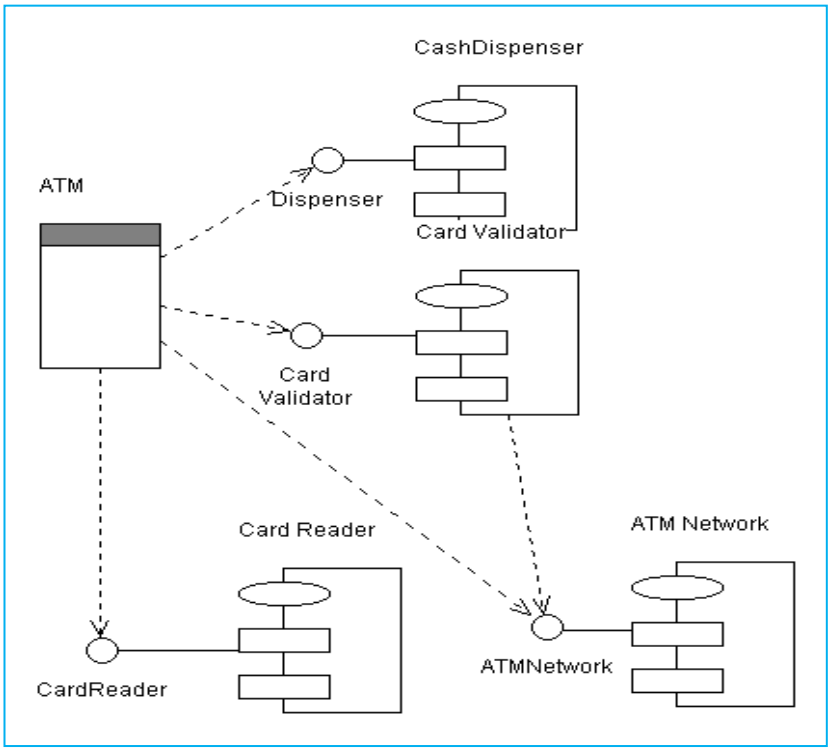
1. *Private*, tidak dapat dipanggil dari luar class yang bersangkutan
2. *Protected*, hanya dapat dipanggil oleh class yang bersangkutan dan anak-anak yang mewarisinya
3. *Public*, dapat dipanggil oleh siapa saja



Gambar 3.6 Contoh *Class Diagram*

6. 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.

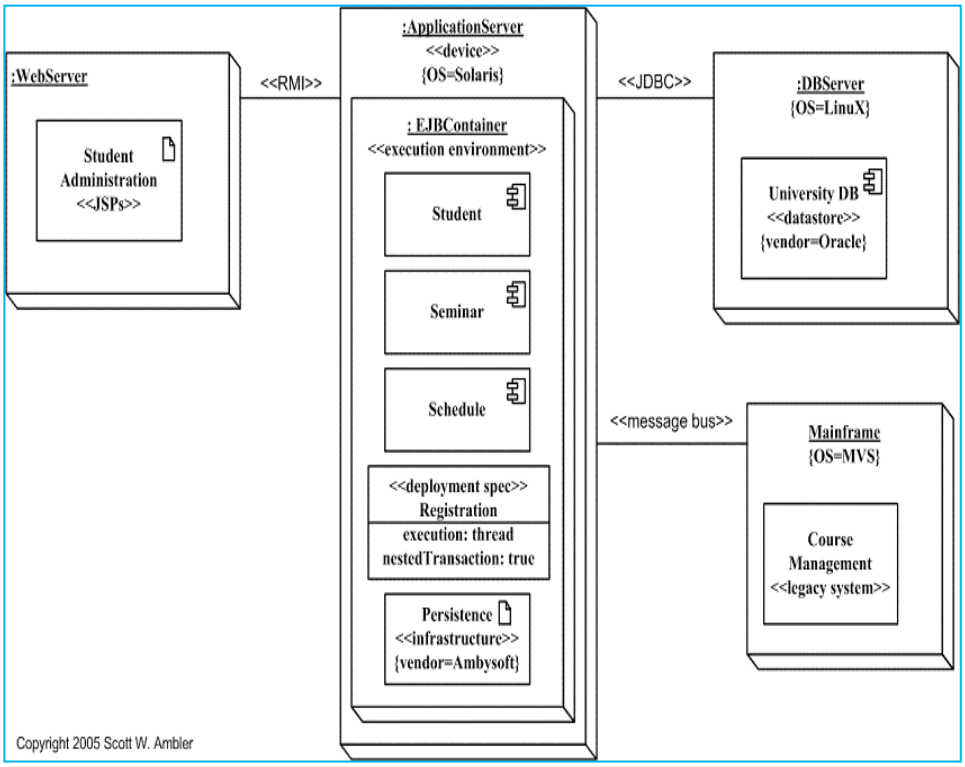


Gambar 3.8

Contoh *Component Diagram*

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.



Gambar 3.8

Contoh *Deployment Diagram*

3.2.4 Pengujian Software

Pengujian *black box* adalah pengujian aspek fundamental sistem tanpa memperhatikan struktur logika internal perangkat lunak. Metode ini digunakan untuk mengetahui apakah perangkat lunak berfungsi dengan benar. Pengujian *black box* merupakan metode peracangan data uji yang didasarkan pada spesifikasi perangkat lunak.

Faktor Pengujian *Black Box*

1. *Authorization*

Menjamin data diproses sesuai dengan ketentuan manajemen. Authorisasi menyangkut proses transaksi secara umum dan khusus.

2. *Continuity of processing*

Menekankan kemampuan untuk meneruskan proses, ketika terjadi suatu permasalahan, dengan menetapkan prosedur yang diperlukan dan *back-up* informasi untuk melindungi operasi yang mungkin hilang karena masalah tersebut..

3. *Access control*

Menekankan sumberdaya sistem harus dilindungi dari kemungkinan modifikasi, pengrusakan, penyalahgunaan dan Prosedur keamanan harus dijalankan secara penuh untuk menjamin integritas data dan program aplikasi.

4. *Correctness*

Menjamin pada data yang dimasukkan, proses dan *output* yang dihasilkan dari aplikasi harus akurat dan lengkap. Kelengkapan dan akurasi akan dicapai melalui kontrol transaksi dan elemen data

5. *Ease of operations*

Sejumlah usaha yang diminta untuk mengintegrasikan sistem ke dalam lingkungan operasi dan lingkungan sistem aplikasi, berupa prosedur manual dan otomatisasi.