BAB 2
TINJAUAN PUSTAKA

2.1 Profil Perusahaan

Pada tahun 1986 di Jalan Gunungpuntang Km 28 Cimaung didirian sebuah perusahaan sepatu yang didirikan oleh Bapak Wahyu berbekal pengetahuan, keterampilan, dan pengalaman selama menjadi tukan sepatu, perusahaan tersebut di berinama GP SHOES. Sebelumnya perusahaan ini hanyalan sebuah perusahaan industry kecil dengan modal terbatas yang hanya untuk memenuhi kebutuhan masyarakat sekitar.

Awal mulanya bapak wahyu mengawali usahanya dengan bekal pengalaman dan keterampilan yang dimilikinya dalam memproduksi sepatu untuk pria dan wanita. Dengan bertambahnya produksi sepatu, bapak wahyu meminta bantuan rekanya untuk menyelesaikan produksi tersebut agar proses produksi dapat terselesaikan. Hasil produksinya dikirim ke toko sepatu di daerah cibaduyut.

Perkembangan usaha Bapak wahyu dari tahun ke tahun menunjukan peningkatan dengan banyaknya pesanan untuk membuatkan sepatu sehingga untuk melayani pesanan yang begitu banyak maka Bapak wahyu menambah karyawan untuk membantu mengerjakan pesanan tersebut. Selain itu bapak Wahyu menambah jumlah peralatan sehingga usaha ini pada tahun 1995 merupakan usaha *home industry* yang terkenal di daerah Kebontunggul khususnya. Dalam perkembangan selanjutnya, perusahaan terus menambah kapasitas produksinya dan memperluas jangkauan pemasaran produksinya. Pada tahun 2009 pemasaran produksinya sudah mengekspor produknya keluar negeri.

2.1.1 Visi dan Misi Perusahaan

Visi :

“Menjadi perusahaan sepatu kualitas terbaik”

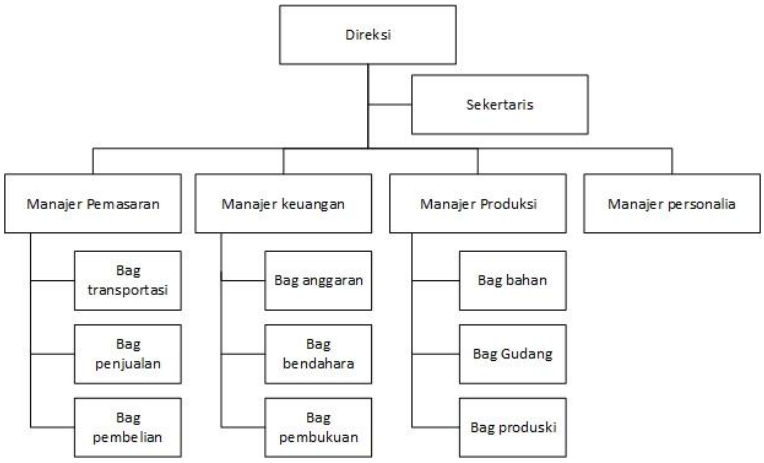
Misi :

1. Menjaga kepercayaan para pelanggan.
2. Menghasilkan produk yang berkualitas tinggi

3. Mempertahankan harga yang kompetitif dan menciptakan produk yang berbeda dari perusahaan lain.

2.1.2 Struktur Organisasi

Struktur organisasi adalah pola hubungan antara bagian-bagian dari instansi atau menggambarkan dengan jelas pemisah kegiatan pekerjaan antar bagian dalam suatu instansi. Berikut struktur organisasi divisi produksi di GP SHOES dapat dilihat pada Gambar 2.1.



Gambar 2.1 Struktur Organisasi GP SHOES

2.2 Landasan Teori

Pada sub bab ini berisi teori- teori yang digunakan dalam proses analisis dan perancangan serta implementasi dalam tugas akhir ini

2.2.1 Data Warehouse

Data warehouse merupakan kumpulan data yang berorientasi subjek, terintegrasi, tidak dapat di *update,* memiliki dimensi waktu, yang digunakan untuk mendukung proses manajemen pengambilan keputusan dan kecerdasan bisnis [4]. Data *warehouse* memiliki karakteristik, sebagai berikut :

1. Berorientasi subjek (*Subject Oriented)*
   Data warehouse berorientasi subjek artinya *data warehouse* didesain untuk menganalisa data berdasarkan subjek-subjek tertentu dalam organisasi, bukan pada proses atau fungsi aplikasi tertentu. *Data warehouse* diorganisasikan disekitar subjek-subjek utama dari perusahaan

(*customers*, *product* dan *sales*), hal ini dikarenakan kebutuhan dari *data warehouse* untuk menyimpan data-data yang bersifat penunjang suatu keputusan.

2. Terintegrasi (*Integrated)*
   Data Warehouse dapat menyimpan data-data yang berasal dari sumber-sumber yang terpisah kedalam suatu format yang konsisten dan saling terintegrasi satu dengan lainnya.
3. Rentang Waktu (*Time-variant*)
   Seluruh data pada data warehouse dapat dikatakan akurat atau valid pada rentang waktu tertentu. Untuk melihat interval waktu yang digunakan dalam mengukur keakuratan suatu data warehouse, kita dapat menggunakan cara antara lain :
   - Cara yang paling sederhana adalah menyajikan data warehouse pada rentang waktu tertentu, misalnya antara 5 sampai 10 tahun ke depan.
   - Cara yang kedua, dengan menggunakan variasi/perbedaan waktu yang disajikan dalam data warehouse baik implicit maupun explicit secara explicit dengan unsur waktu dalam hari, minggu, bulan dsb. Secara implicit misalnya pada saat data tersebut diduplikasi pada setiap akhir bulan, atau per tiga bulan. Unsur waktu akan tetap ada secara implisit didalam data tersebut.
   - Cara yang ketiga,variasi waktu yang disajikan data warehouse melalui serangkaian snapshot yang panjang. Snapshot merupakan tampilan dari sebagian data tertentu sesuai keinginan pemakai dari keseluruhan data yang ada bersifat read-only.
4. Non-Volatile

Data warehouse tidak dapat di-*update* secara *real time* tetapi di-*referesh* dari sistem operasional secara reguler. Data yang baru selalu ditambahkan kedalam *database* itu sendiri. *Database* tersebut secara terus-menerus menerima dan menyimpan data baru, kemudian disatukan dengan data sebelumnya.

2.2.2 Data Mart

Data Mart adalah bagian pada data warehouse yang mendukung pembuatan laporan dan analisa data pada suatu unit, bagian atau operasional pada suatu persahaan. Data mart sering digunakan untuk memberikan informasi kepada segmen fungsional organisasi. [1].

Ada empat tugas yang dapat dilakukan dengan adanya data mart, empat tugas tersebut adalah sebagai berikut :

1. Pembuatan laporan

   Pembuatan laporan merupakan salah satu kegunaan *data mart* yang paling umum dilakukan. Dengan menggunakan *query* sederhana didapatkan laporan per hari, per bulan, per tahun, atau jangka waktu kapanpun yang diinginkan.
2. *On-Line Analytical Processing* (OLAP)

   Dengan adanya *data mart*, semua informasi baik detail maupun hasil *summary* yang dibutuhkan dalam proses analisa mudah didapat. OLAP merupakan konsep data multidimensi dan memungkinkan pada pemakai menganalisa data sampai mendetail, tanpa mengetikkan satupun perintah SQL. Fasilitas lain adalah *roll-up* dan *drill-down*. *Drill-down* adalah kemampuan untuk melihat detail dari suatu informasi dan *roll-up* adalah kebalikannya.
3. *Data Mining*

   Data mining merupakan proses untuk menggali (*mining*) pengetahuan dan informasi baru dari data yang berjumlah banyak pada *data mart*.
4. Proses informasi *executive*

Data mart dapat membuat ringkasan informais yang penting dengan tujuan membuat keputusan bisnis, tanpa harus menjelajahi keseluruhan data. Dengan menggunakan *data mart* segala laporan telah diringkas dan dapat pula mengetehaui segala rinciannya secara lengkap, sehingga mempermudah proses pengambilan keputusan.

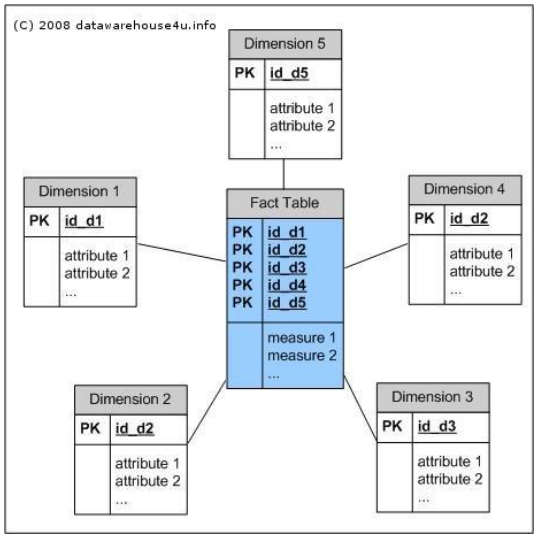
2.2.2.1 Model Dimensional Data Mart

Model dimensional pada *data mart* terdiri dari tabel fakta dan tabel dimensi. Tabel fakta merupakan tabel yang berisi kumpulan *foreign key* dari *primary* key yang terdapat pada masing-masing tabel dimensi, sedangkan tabel dimensi merupakan tabel yang berisi data detail yang menjelaskan *foreign key* yang terdapat pada tabel fakta.

Ada beberapa model skema yang terdapat pada pemodelan *data mart*, yaitu skema *star*, skema *snowflake*, dan skema *constellation*. Penjelasan dari masing-masing model skema adalah sebagai berikut :

1. *Star Schema*

   Skema ini mengikuti bentuk bintang, dimana terdapat satu tabel fakta di pusat bintang dengan beberapa tabel dimensi yang mengelilinginya. Semua tabel dimensi berhubungan dengan ke tabel fakta. Tabel fakta memiliki beberapa *primary key* dalam tabel dimensi. Berikut adalah contoh skema bintang dapat dilihat pada Gambar 2.2

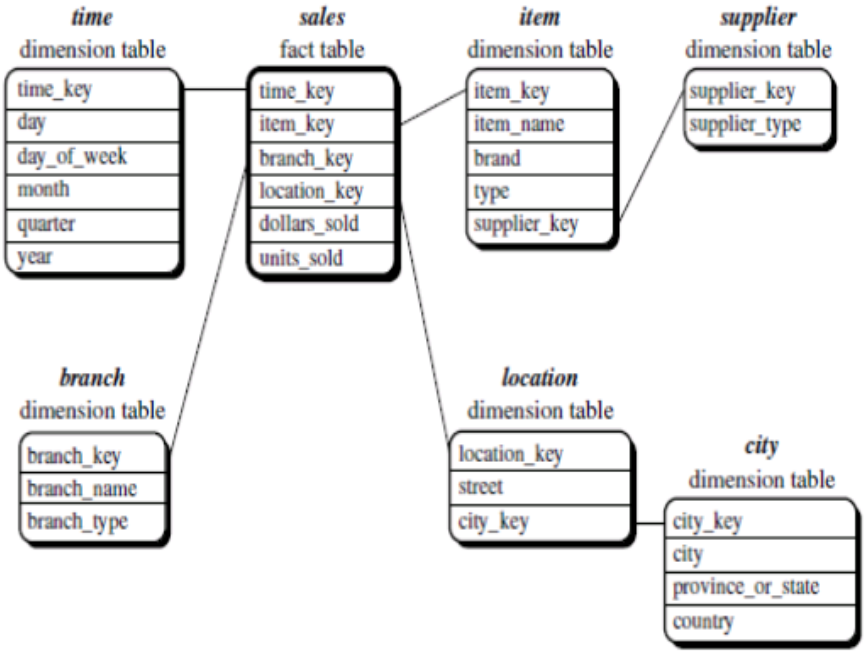


Gambar 2.2 *Star Schema*

2. *Snowflake Schema*

   Menurut *Connolly dan Begg* [1], *Snowflake Schema* merupakan perluasan dari skema bintang dengan tambahan beberapa tabel dimensi yang tidak berhubungan secara langsung dengan tabel fakta. Tabel dimensi tersebut

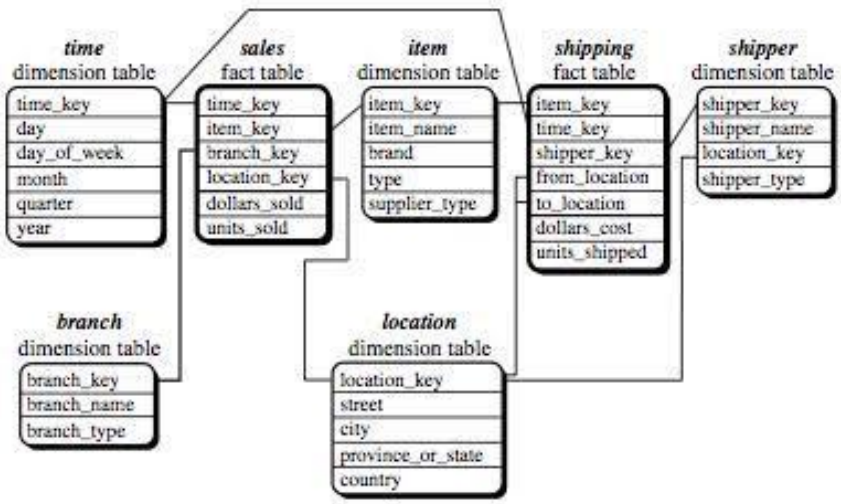
berhubungan dengan tabel dimensi yang lain. Berikut adalah contoh skema bola salju dapat dilihat pada Gambar 2.3



Gambar 2.3 *Snowflake schema*

3. Skema *Constellation*

Skema Constellation adalah skema muti dimensional yang berisikan lebih dari satu table fakta yang saling berbagi table dimensi. Berikut adalah contoh skema constellation dapat dilihat pada Gambar 2.4



Gambar 2.4 Skema *Constellation*

2.2.2.2 ETL (*Extract, Transform, Loading*)

Proses ETL (*Extract, Transform, Loading*) merupakan proses yang harus dilalui dalam pembentukan *data mart* [5]. ETL merupakan fase pemrosesan data dari sumber data kemudian masuk ke dalam *data mart*. Tujuan ETL adalah mengumpulkan, menyaring, mengolah dan menggabungkan data dari berbagai sumber untuk disimpan ke dalam *data mart*. Berikut adalah penjelasan dari tiap proses ETL :

1. Ekstraksi Data (*Extract*)

   Ekstraksi data adalah proses pengambilan data dari berbagai sistem operasional, baik menggunakan *query* atau aplikasi ETL. Terdapat beberapa fungsi ekstraksi data, yaitu :

   a. Ekstraksi data secara otomatis dari aplikasi sumber.
   b. Penyaringan atau seleksi data hasil ekstraksi.
   c. Pengiriman data dari berbagai *platform* aplikasi ke sumber data.
   d. Perubahan format *layout* data dari format aslinya.
   e. Penyimpanan dalam *file* sementara untuk penggabungan dengan hasil ekstraksi dari sumber lain.

2. Transformasi Data (*Transformation*)

   Transformasi adalah proses dimana data hasil ekstraksi disaring dan diubah sesuai dengan aturan bisnis yang berlaku. Langkah-langkah dalam transformasi data adalah sebagai berikut :

   a. Memetakan data input dari skema data aslinya ke skema *data mart*.
   b. Melakukan konversi tipe data.
   c. Membersihkan dan membuang data yang sama (duplikasi).
   d. Memeriksa referensi data.
   e. Mengisi nilai-nilai yang kosong dengan nilai *default*.
   f. Menggabungkan data.

3. Pengisian Data (*Loading*)

Pengisian data adalah proses memasukkan data yang didapatkan dari hasil transformasi ke dalam *data mart*. Cara untuk memasukkan data adalah dengan menjalankan *SQL script* secara periodik.

2.2.2.3 OLAP (*On-Line Analytical Processing*)

OLAP (*On-Line Analytical Processing*) adalah teknologi yang memproses data ke dalam struktur multidimensi, menyediakan jawaban yang cepat untuk *query* analisis yang kompleks dengan tujuan untuk mengorganisir sejumlah data yang besar, agar bisa dianalisa dan dievaluasi dengan cepat serta menyediakan kecepatan dan fleksibelitas untuk mendukung analisa dalam waktu nyata [6].

Ada beberapa karakteristik OLAP, yaitu :

1. Mengijinkan pelaku bisnis melihat data dari sudut pandang *logical* dan multidimensional pada *data warehouse*.
2. Memfasilitasi *query* yang kompleks dan analisa bagi pengguna.
3. Mengijinkan pengguna melakukan *drill-down* untuk menampilkan data yang lebih detail atau *roll-up* untuk agregasi dari suatu dimensi atau beberapa dimensi.
4. Menyediakan proses kalkulasi dan perbandingan data.
5. Menampilkan hasil dalam bentuk tabel atau grafik.

Keuntungan dari OLAP, yaitu :

1. Mengingkatkan produktivitas pemakai akhir bisnis, pengembang IT, dan keseluruhan organisasi.
2. Pengawasan yang lebih dan akses tepat waktu terhadap informasi strategis dapat membuat pengambilan keputusan lebih cepat.
3. Mengurangi pengembangan aplikasi bagi staf IT dengan membuat pemakaian akhir dapat mengubah skema dan membuat model sendiri.
4. Penyimpanan pengawasan organisasi melalui integritas data koorporasi sebagai aplikasi OLAP tergantung pada *data warehouse* dan sistem OLTP untuk memperbaharui sumber tingkatan data.

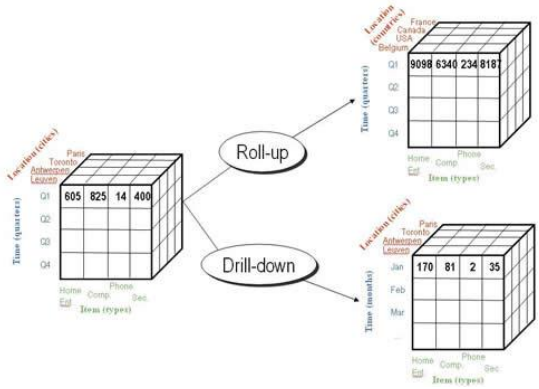
OLAP dapat digunakan untuk melakukan seperti:

1. Konsolidasi (*roll-up*)

   Konsolidasi melibatkan pengelompokan data.

2. *Drill-down*

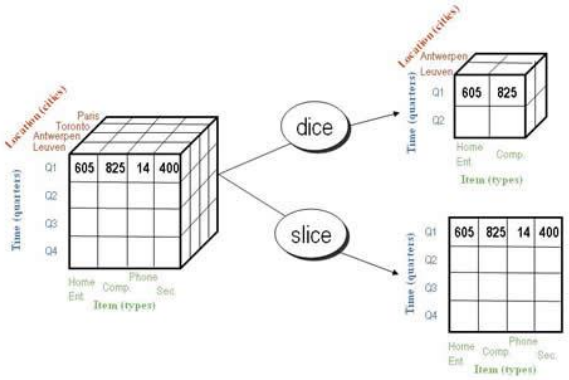
   Suatu bentuk yang merupakan kebalikan dari konsolidasi untuk menjabarkan data yang ringkas menjadi data yang lebih detail. Gambaran untuk *roll-up* dan *drill-down* dapat dilihat pada Gambar 2.5



Gambar 2.5 *Roll-up* dan *Drill-down*

3. *Slicing* dan *dicing*

   Menjabarkan pada kemampuan untuk melihat data dari sudut pandang. Gambaran untuk *slicing* dan *dicing* dapat dilihat pada Gambar 2.6



Gambar 2.6 *Slicing* dan *Dicing*

2.2.2.4 Pengertian Tabel Fakta dan Dimensi

Menurut Inmon [4], table fakta adalah tabel pusat dari skema bintang dimana data sering muncul akan ditempatkan di tabel tersebut. Tabel fakta disebut juga tabel utama(major table), merupakan inti dari skema bintang dan berisi data actual yang akan dianalisis(data kuantitatif dan transaksi). Tabel fakta adalah tabel yang umumnya mengandung angka dan data historis dimana key(kunci) yang dihasilkan sangat unik karena merupakan kumpulan foreign key dan primary key yang ada pada masing-masing tabel dimensi yang berhubungan atau merupakan tabel terpusat dari skema bintang. Tabel fakta menyimpan tipetipe measure yang berbeda, seperti measure yang secara langsung berhubungan dengan tabel dimensi dan measure yang tidak berhubungan dengan tabel dimensi

`Tabel dimensi adalah tempat dimana data-data yang tidak berhubungan yang berelasi dengan tabel fakta ditempatkan di dalam tabel dimensional. Tabel dimensi juga disebut tabel kecil (minor table), karena memegang data deskriptif yang mencerminkan dimensi suatu bisnis. Tabel dimensi merupakan tabel yang berisi kategori dengan ringakasan data detail yang dapat dilaporkan, seperti laporan keuntungan pada tabel fakta dapat dilaporkan sebagai dimensi waktu(berupa per bulan atau per tahun).

2.2.3 Microsoft SQL Server

Microsoft SQL Server adalah sebuah sistem manajemen basis data relasional (RDBMS) produk *Microsoft*. Bahasa *query* utamanya adalah *Transact-*

SQL yang merupakan implementasi dari *SQL* standar ANSI/ISO yang digunakan oleh *Microsoft* dan *Sybase*. Umumnya *SQL Server* digunakan di sunia bisnis yang memiliki basis data yang berskala kecil sampai dengan menengah, tetapi kemudian berkembang dengan digunakannya *SQL Server* pada basis data besar [7]. Pengguna yang menggunakan *SQL Server* dapat menyimpan banyak data dan dapat untuk diimplementasi untuk kepentingan bisnis dan perusahaan.

2.2.4 SSIS (*SQL Server Integration Service*)

SSIS (*SQL Server Integration Services*) adalah suatu platform untuk membangun sistem yang handal untuk integrasi data, *extraction*, *transformation*, dan *loading* yang digunakan pada *data warehousing*. SSIS menawarkan solusi dalam menghadapi permasalahan data integrasi. Selain itu, *tools* ini membatu untuk menigkatkan efisiensi waktu pembuatan.

Arsitektur *SQL Server Integration Services* secara umum berisi berbagai macam komponen, diantaranya yaitu :

1. *SSIS Deginer*. Merupakan *tools* yang digunakan untuk membuat dan mengatur paket *integration service*. Pada *SQL Server 2012*, *tools* ini sudah terintegrasi dengan *Visual Studio 2010* yang merupakan bagian *project Bussiness Intelegence*.
2. *Runtime Engine*. Komponen ini berguna untuk menjalankan semua paket SSIS yang sudah dibuat.
3. *Task* dan *executable binary*.
4. *Data Flow Engine* dan *Data Flow*. Komponen *data flow* merupakan enkapsulasi *data flow engine* yang menyediakan *buffer* didalam memori dan bertugas memindahkan data dari sumber data ke tujuan data. Sedangkan *data flow* merupakan sumber data, tujuan data, dan transformasi.
5. *Integration Services service*. Memungkinkan *SQL Server Management Studio* dapat digunakan untuk melakukan monitoring paket SSIS dan mengatur SSIS *storage* yang digunakan.

SQL Server Import dan *Export Wizard*. *Tools* ini digunakan untuk melakukan *copy* data dari sumber ke tujuan data.

2.2.5 UML (*Unified Modeling Language*)

UML (*Unified Modelling Language*) adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi objek. Hal ini disebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembang untuk membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah untuk dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancangan mereka yang lain [8].

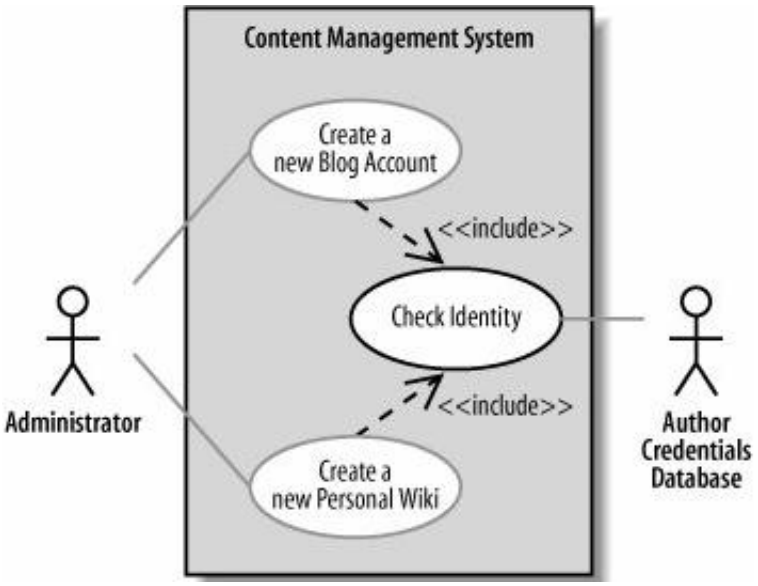
UML merupakan kesatuan dari bahasa pemodelan yang dikembangkan oleh Booch, *Object Modelling Technique* (OMT) dan *Object Oriented Software Engineering* (OOSE) [8]. Metode Booch dari Grady Booch sangat terkenal dengan nama metode *Design Object Oriented*. Metode ini menjadikan proses analisis dan design ke dalam empat tahapan iteratif, yaitu identifikasi kelas-kelas dan objek-objek, identifikasi semantik dari hubungan objek dan kelas tersebut, perincian interface dan implementasi. Keunggulan metode Booch adalah pada detail kayanya dengan notasi dan elemen. Pemodelan OMT yang dikembangkan oleh Rumbaugh didasarkan pada analisis terstruktur dan pemodelan *entity-relationship*. Tahapan utama dalam metodologi ini adalah analisis, design sistem, design objek dan implementasi. Keunggulan metode ini adalah dalam penotasian yang mendukung semua konsep OO. Metode OOSE dari Jacobson lebih memberi penekanan pada *use case*. OOSE memiliki tiga tahapan yaitu membuat model *requirement* dan analisis, design dan implementasi, dan model pengujian (*test model*). Keunggulan metode ini adalah mudah dipelajari karena memiliki notasi yang sederhana namun mencakup seluruh tahapan dalam rekayasa perangkat lunak.

Dengan UML, metode Booch, OMT dan OOSE digabungkan dengan membuang elemen-elemen dari metode lain yang lebih efektif dan elemen-elemen baru yang belum ada pada metode terdahulu sehingga UML lebih ekspresif dan seragam daripada metode lainnya.

2.2.5.1 Use Case Diagram

Use Case adalah deskripsi fungsi dari sebuah sistem dari perspektif pengguna. *Use case* bekerja dengan cara mendeskripsikan tipikal interaksi antara *user* (pengguna) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. Urutan langkah-langkah yang menerangkan antara pengguna dan sistem disebut skenario. Setiap skenario mendeskripsikan urutan kejadian. Setiap urutan diinisialisasi oleh orang, sistem lain, perangkat keras atau urutan waktu. Dengan demikian secara singkat biasa dikatakan *use case* adalah serangkaian skenario yang digabungkan bersama-sama oleh tujuan umum pengguna [8].

Berikut ini adalah contoh *Use Case Diagram*:



Gambar 2.7 *Use Case Diagram*

2.2.5.2 Use Case Scenario

Sebuah diagram yang menunjukkan kasus penggunaan dan aktor dapat menjadi titik awal yang bagus, tetapi tidak memberikan cukup detail untuk desainer sistem untuk benar-benar memahami persis bagaimana kekhawatiran sistem akan dipenuhi. Use Case Skenario adalah cara terbaik untuk mengungkapkan informasi penting ini dalam bentuk penggunaan description setiap kasus berbasis teks harus disertai dengan satu [5].

Berikut ini adalah contoh *Use Case Scenario*:

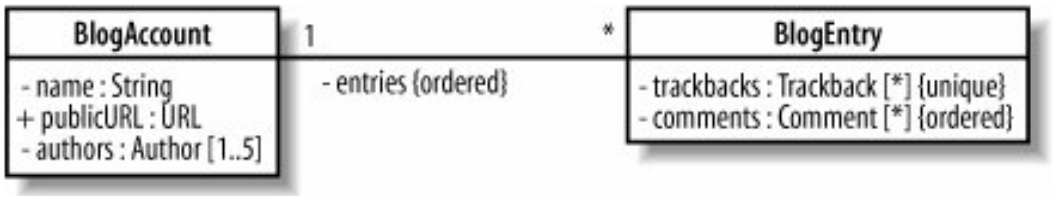
Use case name	**Create a new Blog Account**	
Related Requirements	Requirement A.1.	
Goal In Context	A new or existing author requests a new blog account from the Administrator.	
Preconditions	The system is limited to recognized authors and so the author needs to have appropriate proof of identity.	
Successful End Condition	A new blog account is created for the author.	
Failed End Condition	The application for a new blog account is rejected.	
Primary Actors	Administrator.	
	Secondary Actors	Author Credentials Database.
	Trigger	The Administrator asks the CMS to create a new blog account.
Main Flow	Step	Action
	1	The Administrator asks the system to create a new blog account.
	2	The Administrator selects an account type.
	3	The Administrator enters the author's details.
	4	The author's details are verified using the Author Credentials Database.
	5	The new blog account is created.
	6	A summary of the new blog account's details are emailed to the author.
Extensions	Step	Branching Action
	4.1	The Author Credentials Database does not verify the author's details.
	4.2	The author's new blog account application is rejected.

Gambar 2.8 *Use Case Scenario*

2.2.5.3 Class Diagram

Class Diagram adalah diagram yang digunakan untuk menampilkan beberapa kelas serta paket-paket yang ada dalam sistem/perangkat lunak yang sedang kita kembangkan. Diagram kelas memberikan gambaran/diagram statis tentang sistem/perangkat lunak dan relasi-relasi yang ada di dalamnya [9].

Berikut ini adalah contoh *Class Diagram*:



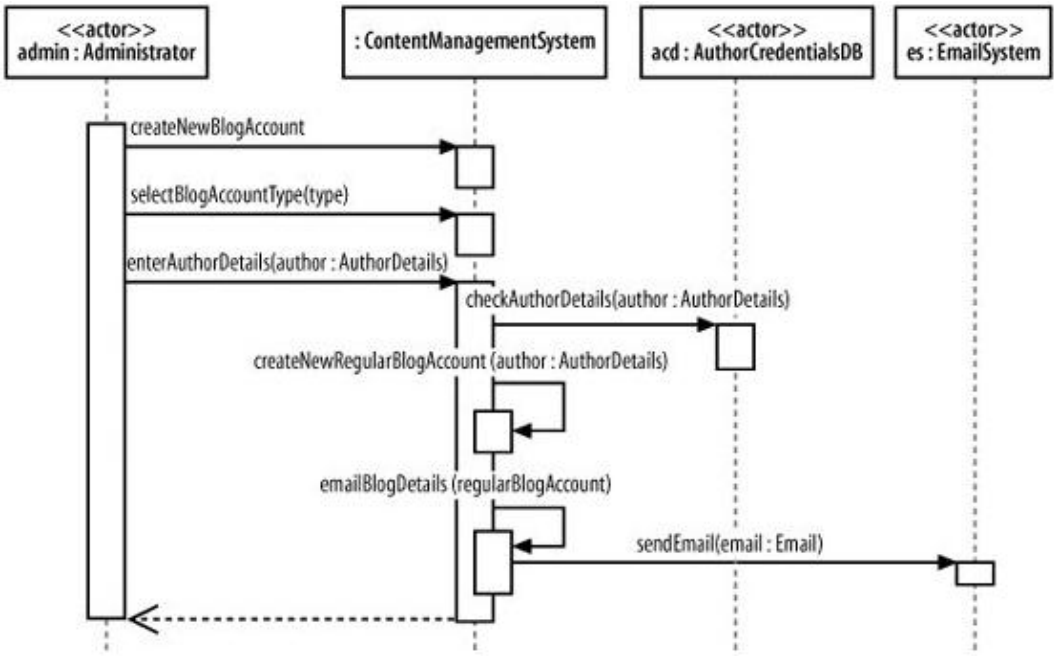
Gambar Error! No text of specified style in document.**..1** *Class Diagram*

2.2.5.4 Sequence Diagram

Sequence Diagram digunakan untuk menggambarkan perilaku pada sebuah skenario. Diagram ini menunjukkan sejumlah contoh objek dan *message* (pesan) yang diletakkan diantara objek-objek ini dalam *use case*.

Komponen utama *sequence diagram* terdiri atas objek yang dituliskan dengan kotak segiempat bernama. *Message* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan *progress* vertical [8].

Berikut ini adalah contoh *Sequence Diagram*:

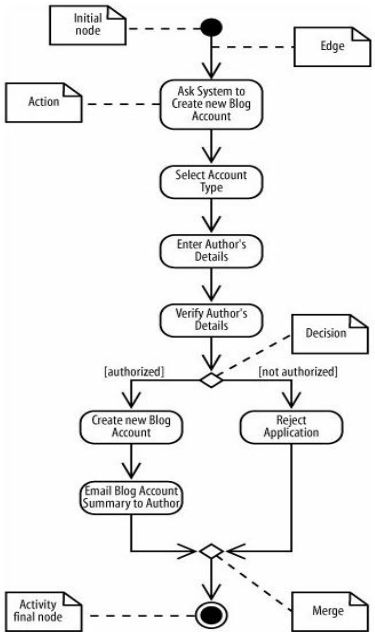


Gambar 2.10 *Sequence Diagram*

2.2.5.5 Activity Diagram

Activity Diagram adalah teknik untuk mendeskripsikan logika prosedural, proses bisnis dan aliran kerja dalam banyak kasus. *Activity Diagram* mempunyai peran seperti halnya *flowchart*, akan tetapi perbedaannya dengan *flowchart* adalah activity diagram mendukung perilaku paralel sedangkan *flowchart* tidak bias [8].

Berikut ini adalah contoh *Activity Diagram*:



Gambar 2.11 *Activity Diagram*