BAB II

TINJAUAN PUSTAKA

II.1. Konsep Dasar Sistem

II.1.1. Pengertian Sistem

Terdapat dua kelompok pendekatan di dalam pendefenisian sistem, yaitu kelompok yang menekankan pada prosedur dan kelompok yang menekankan pada elemen atau komponennya. Pendekatan yang menekankan pada prosedur mendefinisikan sistem sebagai suatu jaringan kerja prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran tertentu. Sedangkan pendekatan sistem yang lebih menekankan pada elemen atau komponen mendefinisikan sistem sebagai kumpulan elemen yang berinteraksi untuk mencapai tujuan tertentu. Kedua kelompok definisi ini adalah benar dan tidak bertentangan. Yang berbeda adalah cara pendekatannya (Tata Sutabri; 2012 : 2).

Sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, terkumpul bersama-sama untuk melakukan suatu kegiatan atau tujuan tertentu (Yakub; 2012 : 1).

Sistem merupakan sekumpulan elemen yang saling terkait atau terpadu yang dimaksud untuk mencapai suatu tujuan. Dengan demikian di dalam suatu sistem, komponen-komponen ini tidak dapat berdiri sendiri, tetapi sebaliknya saling berhubungan hingga membentuk satu kesatuan sehingga tujuan sistem itu dapat tercapai (Kusrini, S. Kom & Andri Kaniyo; 2007 : 5).

II.1.2. Elemen-elemen Sistem

Ada beberapa elemen yang membentuk sebuah sistem yaitu tujuan, masukan, proses, keluaran, batas, mekanisme pengendalian dan umpan balik serta lingkungan.

a. Tujuan

   Tujuan ini menjadi motivasi yang mengarahkan pada sistem, karena tanpa tujuan yang jelas sistem menjadi tak terarah dan tak terkendali.

b. Masukan *(input)* sistem

   Masukan *(input)* sistem adalah segala sesuatu yang masuk kedalam sistem dan selanjutnya menjadi bahan untuk diproses. Masukan dapat berupa hal-hal berwujud maupun yang tidak berwujud. Masukan berwujud adalah bahan mentah, sedangkan yang tidak berwujud adalah informasi.

c. Proses

   Proses merupakan bagian yang melakukan perubahan atau transformasi dari masukan menjadi keluaran yang berguna dan lebih bernilai.

d. Keluaran *(output)*

   Keluaran (*output)* merupakan hasil dari pemrosesan sistem dan keluaran dapat menjadi masukan untuk subsistem lain.

e. Batas *(boundary)* sistem

   Batas *(boundary)* sistem adalah pemisah antara sistem dan daerah diluar sistem. Batas sistem menentukan konfigurasi, ruang lingkup atau kemampuan system.

f. Mekanisme pengendalian dan umpan balik

   Mekanisme pengendalian diwujudkan dengan menggunakan umpan balik *(feedback)*, sedangkan umpan balik ini digunakan untuk mengendalikan masukan maupun proses. Tujuannya untuk mengatur agar sistem berjalan sesuai dengan tujuan.

g. Lingkungan

   Lingkungan adalah segala sesuatu yang berada diluar sistem.

II.1.3. Klasifikasi Sistem

Sistem dapat diklasifikasikan dari beberapa sudut pandang. Klasifikasi sistem tersebut diantaranya yaitu sistem abstrak *(abstract system)*, sistem fisik *(physical system)*, sistem tertentu *(deterministic system)*, sistem tak tentu *(probabilistic system)*, sistem tertutup *(close system)* dan sistem terbuka *(open system)*.

a. Sistem tak tentu *(probabilistic system)*, adalah suatu sistem yang kondisi masa depannya tidak dapat diprediksi karena mengandung unsure probabilitas. Sistem arisan merupakan contoh *probabilistic system* karena sistem arisan tidak dapat diprediksi dengan pasti.
b. Sistem abstrak *(abstract system)*, adalah sistem yang berupa pemikiran atau ide-ide yang tidak tampak secara fisik. Sistem teologia yang berisi gagasan tentang hubungan manusia dengan Tuhan merupakan contoh *abstract system*.

c. Sistem fisik *(physical system),* adalah sistem yang ada secara fisik. Sistem komputer, sistem akuntansi, sistem produksi, sistem sekolah dan sistem transportasi merupakan contoh *physical system.*
d. Sistem tertentu *(determinic system),* adalah sistem yang beroperasi dengan tingkah laku yang dapat diprediksi, interaksi antara bagian dapat dideteksi dengan pasti sehingga keluarannya dapat diramalkan. Sistem komputer sudah diprogramkan merupakan contoh *determinic system* karena program komputer dapa diprediksi dengan pasti.
e. Sistem tertutup *(close system),* sistem yang tidak bertukar materi , informasi atau energi dengan lingkungan. Sistem ini tidak berinteraksi dan tidak dipengaruhi oleh lingkungan, misalnya : reaksi kimia dalam tabung yang terisolasi.
f. Sistem terbuka *(open system),* adalah sistem yang berhubungan dengan lingkungan dan dipengaruhi oleh lingkungan. Sistem perdagangan merupakan contoh *open system* karena dapat dipengaruhi oleh lingkungan.

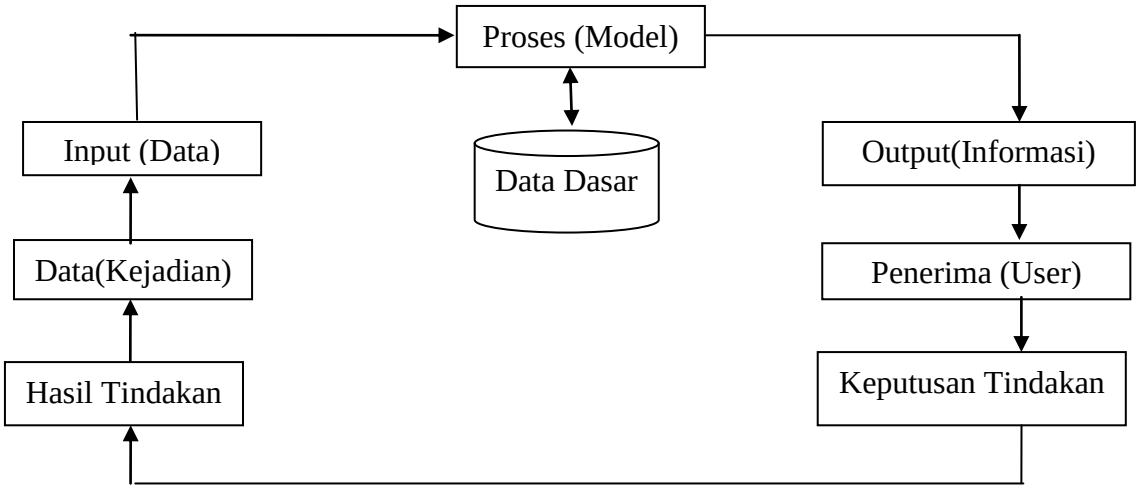
II.2. Konsep Dasar Informasi

Informasi adalah data yang diolah menjadi bentuk lebih berguna dan lebih berarti bagi yang menerimanya. Informasi juga disebut data yang diproses atau data yang memiliki arti. Informasi merupakan data yang telah diproses sedemikian rupa sehingga meningkatkan pengetahuan seseorang yang menggunakan. Informasi dapat berupa data mentah, data tersusun, kapasitas sebuah saluran informasi, dan sebagainya (Yakub; 2012 : 8).

II.3. Konsep Dasar Sistem Informasi

Sistem Informasi merupakan kombinasi teratur dari orang-orang, perangkat keras, perangkat lunak, jaringan komunikasi, dan sumber daya data yang mengumpulkan, mengubah, dan menyebarkan informasi dalam sebuah organisasi. Sistem informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan data transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategi dari suatu organisasi serta menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan. Sistem informasi juga dapat didefinisikan sebagai suatu sitem yang dibuat oleh manusia yang terdiri dari komponen-komponen dalam organisasi untuk menyajikan informasi. Sistem informasi merupakan sistem pembangkit informasi, kemudian dengan integrasi yang dimiliki antar subsistem, maka sistem informasi akan mampu menyediakan informasi yang berkualitas, tepat, cepat dan akurat sesuai dengan manajemen yang membutuhkannya. Sistem informasi juga merupakan suatu kumpulan dari komponen-komponen dalam organisasi yang berhubungan denga proses penciptaan dan aliran informasi. Pada lingkungan berbasis komputer, sistem informasi menggunakan perangkat keras dan lunak komputer, jaringan telekomunikasi, manajemen basis data, dan berbagai bentuk teknologi informasi yang lain dengan tujuan untuk mengubah sumber data menjadi berbagai macam informasi yang dibutuhkan oleh pemakai (Yakub; 2012 : 17-18).

Adapun contoh proses sistem informasi dapat dilihat pada gambar II.1



Gambar II.1. Siklus Informasi

(Sumber: Yakub ; 2012:12)

II.4. Sistem Informasi Akuntansi

Sistem Informasi Akuntansi (SIA) adalah sistem yang bertujuan untuk mengumpulkan data serta melaporkan informasi yang berkaitan dengan transaksi keuangan (Anastasia Diana dan Lilis Setiawati ; 2008 : 4 - 5).

Manfaat atau tujuan sistem informasi akuntansi tersebut sebagai berikut.

1. Mengamankan harta/ kekayaan perusahaan.
2. Menghasilkan beragam informasi untuk pengambilan keputusan.
3. Menghasilkan informasi untuk pihak eksternal.
4. Menghasilkan informasi untuk penilaian kinerja karyawan atau divisi.
5. Menyediakan data masa lalu untuk kepentingan audit (pemeriksaan).
6. Menghasilkan informasi untuk penyusunan dan evaluasi anggaran perusahaan.
7. Menghasilkan informasi yang diperlukan dalam kegiatan perencanaan dan pengendalian.

Sistem Informasi Akuntansi (SIA) merupakan sebuah sistem informasi yang mengubah data transaksi bisnis menjadi informasi keuangan yang berguna bagi pemakainya (Kusrini dan Andi Koniyo ; 2007 : 10).

II.5. Akuntansi

Akuntansi merupakan proses mengidentifikasi, mengukur, mencatat dan mengkomunikasikan peristiwa-peristiwa ekonomi dari suatu organisasi (bisnis maupun nonbisnis) kepada pihak-pihak yang berkepentingan dengan informasi bisnis tersebut (Anastasia Diana dan Lilis Setiawati ; 2011 : 14)

Akuntansi adalah sebuah sistem informasi yang menghasilkan informasi keuangan kepada pihak-pihak yang berkepentingan mengenai aktivitas ekonomi dan kondisi suatu perusahaan (rudianto ; 2009 : 4).

Menurut Kusrini dan Andi Koniyo (2007 : 15-16) menjelaskan kata akuntansi berasal dari kata, *to account,* yang berarti memperhitungkan atau mempertanggungjawabkan. Kata akuntansi sebenarnya diserap dari kata *accountancy* yang berarti hal-hal yang bersangkutan dengan *accountant* (akuntan) atau bersangkutan dengan hal-hal yang dikerjakan oleh akuntan dalam menjalankan profesinya. Beberapa pengertian lain mengenai akuntansi, yaitu :

a. Akuntansi adalah suatu sistem yang mengukur aktifitas-aktifitas bisnis, memproses informasi tersebut kedalam bentuk laporan dan mengkomunikasikannya kepada para pengambil keputusan.

b. Akuntansi adalah suatu proses pencatatan, penggolongan, peringkasan dan pelaporan atas transaksi keungan perusahaan serta implementasinya.

II.6. Penegertian Akuntansi Biaya Produksi

Pengertian Akuntansi Biaya Adalah proses pencatatan, penggolongan, peringkasan dan penyajian biaya pembuatan dan penjualan produk atau jasa, dengan cara-cara tertentu serta penafsiran terhadapnya. Biaya dalam arti luas adalah pengorbanan sumber ekonomis, yang diukur dalam satuan uang, yang telah terjadi atau kemungkinan akan terjadi untuk mencapai tujuan tertentu. Sedangkan biaya dalam arti sempit biaya merupakan bagian daripada harga pokok yang dikorbankan di dalam usaha untuk memperoleh penghasilan. Tujuan Akuntansi Biaya adalah untuk menyediakan informasi biaya bagi kepentingan manajemen guna membantu mereka di dalam mengelola perusahaan atau bagiannya. Biaya produksi terdiri dari biaya bahan baku, biaya tenaga kerja langsung dan biaya overhead pabrik. (Mulyadi; 2008 : 143).

II.7. Basis Data

Basis dapat diartikan sebagai markas atau gudang, tempat bersarang atau berkumpul. Sedangkan data merupakan representasi fakta dunia nyata yang mewakili suatu objek seperti manusia (pegawai, siswa, pembeli, pelanggan), barang, hewan peristiwa, konsep, keadaan, dan sebagainya yang direkam dalam bentuk angka, huruf, simbol, teks, gambar, bunyi, atau kombinasinya. Basis data (*database*) merupakan kumpulan data yang saling berhubungan (punya relasi). Relasi biasanya ditunjukkan dengan kunci (*key*) dari tiap *file* yang ada. Dalam satu

file terdapat *record-record* sejenis, sama besar, sama bentuk, yang merupakan satu kumpulan entitas yang seragam. Satu *record* terdiri dari *field-field* yang saling berhubungan dan menunjukkan dalam satu pengertian yang lengkap dalam satu *record*. Berdasarkan pengertian di atas dapat disimpulkan bahwa basis data mempunyai beberapa kriteria penting, yaitu : bersifat *data oriented* dan bukan *program oriented*, dapat digunakan oleh beberapa program aplikasi tanpa perlu mengubah basis datanya. Hal ini juga dapat dikembangkan dengan mudah baik *volume* maupun strukturnya sehingga dapat memenuhi kebutuhan sistem-sistem baru secara mudah. Prinsip utama basis data adalah pengaturan data dengan tujuan utama fleksibilitas dan kecepatan dalam pengambilan data kembali. Adapun tujuan basis data diantaranya sebagai efisiensi yang meliputi *speed, space* & *accuracy*, menangani data dalam jumlah besar, kebersamaan pemakaian (*sharebility*), dan meniadakan duplikasi dan inkonsistensi data (Yakub; 2012 : 51-53).

II.8. *Data Base Management System* (DBMS)

Data Base Management System (DBMS) merupakan kumpulan program aplikasi yang digunakan untuk membuat dan mengelola basis data. DBMS berisi suatu koleksi data dan satu set program untuk mengakses data. DBMS merupakan perangkat lunak (*software*) yang menentukan bagaimana data tersebut diorganisasi, disimpan, diubah, dan diambil kembali. Perangkat lunak ini juga menerapkan mekanisme pengamanan data, pemakaian data bersama, dan konsistensi data (Yakub; 2012 : 55).

Perangkat lunak yang termasuk DBMS adalah sebagai berikut:

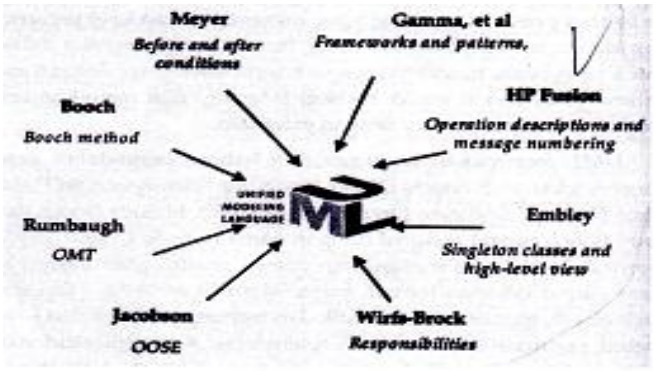
Tabel II.1. Nama DBMS

Nama DBMS	Nama Perusahaan
Access	Microsoft Corporation
DB2	IBM
Informix	IBM
Ingres	Computer Associate
MySQL	The MySQL AB Company
Oracle	Oracle Corporation
PostgreSQL	www.postgresql.com
Sybase	Sybase Inc

(Sumber : Yakub; 2012 : 55)

II.9. UML (*Unified Modeling Language*)

Unified Modeling Language (UML) adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan *class* dan *operation* dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasa-bahasa berorientasi objek seperti C++, Java, C# atau VB.NET (Yuni Sugiarti ; 2013 : 34).



Gambar II.2. Metodologi Pemodelan Berorientasi Objek

Sumber : **Yuni Sugiarti (2013 : 35)**

Pada perkembangan teknologi perangkat lunak, diperlukan adanya bahasa yang digunakan untuk memodelkan perangkat lunak yang akan dibuat dan perlu adanya standarisasi agar orang diberbagai negara dapat mengerti pemodelan perangkat lunak. Pada perkembangan teknik pemrograman berorientasi objek , muncul lah sebuah standarisasi bahasa pemodelan yang membangun perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek, yaitu *Unified Modeling Language* (UML). UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasi, menggambarkan, dan dokumentasi dari sistem perangkat lunak. UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung.

UML hanya berfungsi untuk melakukan pemodelan. Jadi pengguna UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataanya UML paling banyak digunakan pada metodologi berorientasi objek (Rosa A.S dan M.Shalahuddin ; 2011 : 117-118).

UML singkatan dari *Unified Modeling Language* yang berarti bahasa pemodelan standar. Ketika kita membuat model mengunakan konsep UML ada aturan-aturan yang harus diikuti. Bagaimana elemen pada model-model yang kita buat berhubungan satu dengan yang lainnya harus mengikuti standar yang ada. UML bukan hanya sekedar diagram, tetapi juga menceritakan konteksnya.

UML diaplikasikan untuk maksud tertentu, biasanya antar lain untuk :

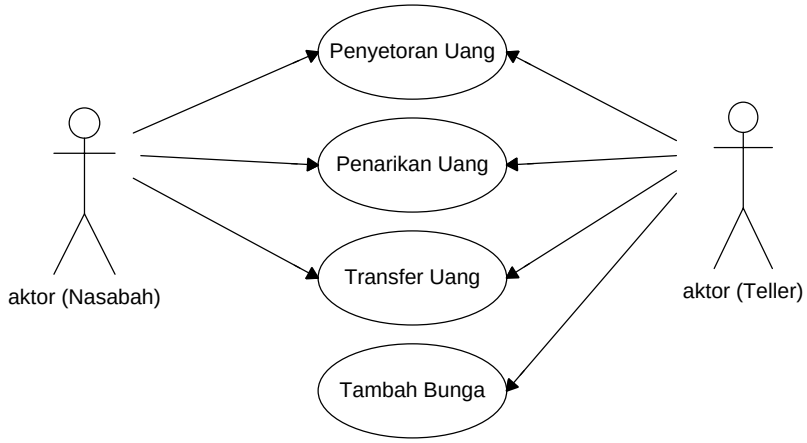
1. Merancang perangkat lunak
2. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem
3. Mendokumentasi sistem yang ada, proses-proses dan organisasinya.

UML telah diaplikasikan dalam bidang investasi perbankan, lembaga kesehatan, departemen pertahanan, sistem terdistribusi, sistem pendukung alat kerja, retail, sales dan supplier. Blok pembangun utama *UML* adalah diagram. Beberapa diagram ada yang rinci (jenis *timing diagram*) dan lainnya ada yang bersifat umum (misalnya diagram kelas). Para pengembang sistem berorientasi objek menggunakan bahasa model untuk menggambarkan, membangun, dan mendokumentasikan sistem yang mereka rancang. *UML* memungkinkan para anggota team untuk bekerja sama dengan bahasa model yang sama dalam mengaplikasikan beragam sistem. Intinya, *UML* merupakan alat komunikasi yang konsisten dalam mensuport para pengembang sistem saat ini (Prabowo Pudjo Widodo dan Herawati ; 2011 : 6-7)

II.9.1. Use Case Diagram

Diagram *Use case* bersifat statis. Diagram ini memperlihatkan himpunan use-case dan aktor-aktor (suatu jenis khusus dari kelas). Diagram ini terutama sangat penting untuk mengorganisasi dan memodelkan perilaku suatu sistem yang dibutuhkan serta diharapkan pengguna (Prabowo Pudjo Widodo dan Herlawati : 2011 : 10).

Gambar di bawah ini merupakan salah satu contoh bentuk diagram *use case* :



Gambar II.3. Contoh Diagram *Use Case*

Sumber : Prabowo Pudjo Widodo dan Herlawati : 2011 : 17

Diagram *use case* bersama dengan narasi *use case* dan skenario mendefinisikan tujuan suatu sistem atau pengklasifi lain seperti *enterprise*, subsistem atau komponen. Konsep ini diperkenalkan oleh Ivar Jacobson bersama organisasinya dalam bentuk metodologi yang mereka namakan *Object-Oriented Software Engineering* (*OOSE*). Tujuan dibentuknya metode ini adalah agar dihasilkan fokus yang baik pada pengembangan dan tujuan utama tanpa terpengaruh oleh implementasi praktis.

Elemen *use case* terdiri dari :

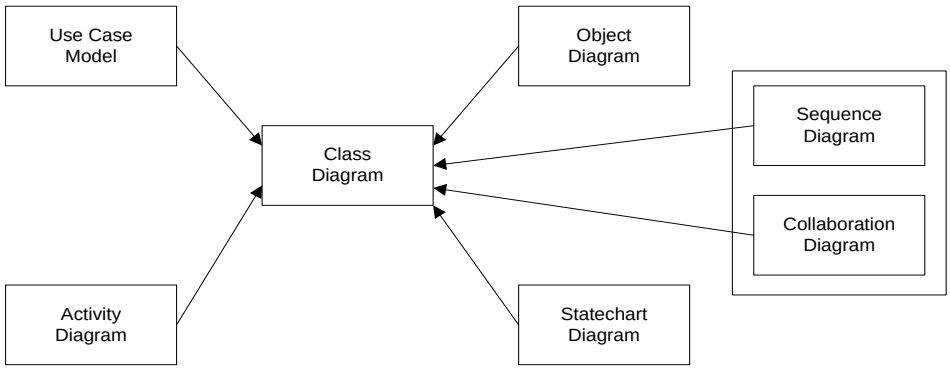
1. Diagram *use case*, disertai dengan narasi dan skenario
2. Aktor (*actor*), mendefinisikan entitas di luar sistem yang memakai sistem
3. Asosiasi (*assosition*), mengindikasikan aktor mana yang berinteraksi dengan *use case* dalam suatu sistem
4. <<*include*>> dan <<*extend*>>, merupakan indikator yang menggambarkan jenis relasi dan interaksi antar *use case*
5. Generalisasi (*generalization*), menggambarkan hubungan turunan antara *use case* atau antar aktor.

Use case mengekspresikan apa yang *user* harapkan terhadap sistem. Narasi *use case* menjelaskan secara detail bagaimana *user* berinteraksi dengan sistem saat mengakses *use case*. Skenario memecah penjelasan narasi untuk menyediakan penjelasan detail terhadap segala kemungkinan yang terjadi pada *use case*, apa yang terjadi dan apa respon sistem (Prabowo Pudjo Widodo dan Herlawati : 2011 : 34-36).

II.9.2. Class Diagram

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 sistem berorientasi objek. Meskipun bersifat statis, sering pula diagram kelas memuat kelas-kelas aktif (Prabowo Pudjo Widodo dan Herlawati : 2011 : 10).

Kemampuan menghasilkan kode program yang dimiliki diagram kelas menyebabkan diagram ini memiliki hubungan yang khas dengan diagram UML lainnya. Pada gambar II.2 terlihat bahwa diagram yang lain memberi masukan kepada diagram kelas. Diagram kelas yang baik menghasilkan suatu rancangan sistem atau program yang mendekati kenyataan.



Gambar II.4. Hubungan diagram kelas dengan diagram UML Lainnya

Sumber : Prabowo Pudjo Widodo dan Herlawati : 2011 : 38

Kelas digambarkan dengan kotak yang terdiri dari sekat-sekat berturut-turut dari atas ke bawah untuk nama, atribut dan operasi. Dalam satu paket, nama kelas harus unik (tidak boleh ada kesamaan nama). Bila namanya sama tapi beda paket dengan kelas lain, harus disebutkan dalam format nama paket :: nama kelas (Prabowo Pudjo Widodo dan Herlawati : 2011 : 76).

II.9.3. Activity Diagram

Diagram Aktivitas (*Activity diagram*) bersifat dinamis. Diagram aktivitas adalah tipe khusus dari diagram status yang memperlihatkan aliran dari suatu aktivitas ke aktivitas lainnya dalam suatu sistem. Diagram ini terutama penting

dalam pemodelan fungsi-fungsi suatu sistem dan memberi tekanan pada aliran kendali antar objek (Prabowo Pudjo Widodo dan Herlawati : 2011 : 11).

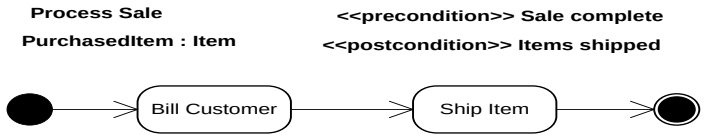
Diagram aktivitas lebih memfokuskan diri pada eksekusi dan alur sistem dari pada bagaimana sistem itu dirakit. Diagram ini tidak hanya memodelkan software melainkan memodelkan model bisnis juga. Diagram aktivitas menunjukkan aktivitas sistem dalam bentuk kumpulan aksi-aksi. Ketika digunakan dalam pemodelan *software*, diagram aktivitas merepresentasikan pemanggilan suatu fungsi tertentu misalnya *call*. Sedangkan bila digunakan dalam pemodelan bisnis, diagram ini menggambarkan aktivitas yang dipicu oleh kejadian-kejadian di luar seperti pemesanan atau kejadian-kejadian internal misalnya proses penggajian tiap jumat sore.

Aktivitas merupakan kumpulan aksi-aksi. Aksi-aksi melakukan langkah sekali saja tidak boleh dipecah menjadi beberapa langkah lagi. Contoh aksi yaitu :

a. Fungsi matematika
b. Pemanggilan perilaku
c. Pemrosesan data

Ketika kita menggunakan diagram aktivitas untuk memodelkan perilaku suatu *classifier*, *classifier* dikatakan kontek dari aktivitas. Aktivitas dapat mengakses atribut dan operasi *classifier*, tiap objek yang terhubung dan parameter-parameter jika aktivitas memiliki hubungan dengan perilaku. Ketika digunakan untuk model proses bisnis, informasi itu biasanya disebut *process-relevant* data. Aktivitas diharapkan dapat digunakan ulang dalam suatu aplikasi,

sedangkan aksi biasanya *spesific* dan digunakan hanya untuk aktivitas tertentu (Prabowo Pudjo Widodo dan Herlawati : 2011 : 143-145).



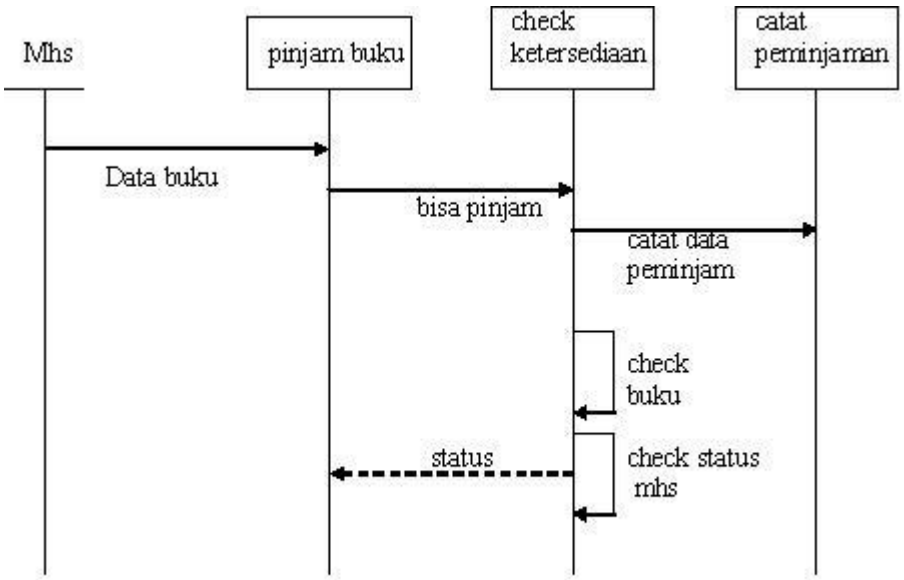
Gambar II.5. Diagram aktivitas dengan kondisi mula dan akhir

Sumber : Prabowo Pudjo Widodo dan Herlawati : 2011 : 147

II.9.4. *Secuence Diagram*

Diagram urutan (*Squence Diagram*) adalah diagram interaksi yang menekankan pada pengiriman pesan dalam suatu waktu tertentu (Prabowo Pudjo Widodo dan Herlawati : 2011 : 11).

Sequence Diagram digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai sebuah respon dari suatu kejadian untuk menghasilkan *output* tertentu. *Sequence Diagram* diawali dari apa yang memicu aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara *internal* dan *output* apa yang dihasilkan.



Gambar II.6. Diagram Urutan (*Sequence Diagram*) Untuk Menggambarkan Langkah Sebagai Respon

Sumber : Prabowo Pudjo Widodo dan Herlawati : 2011 : 11

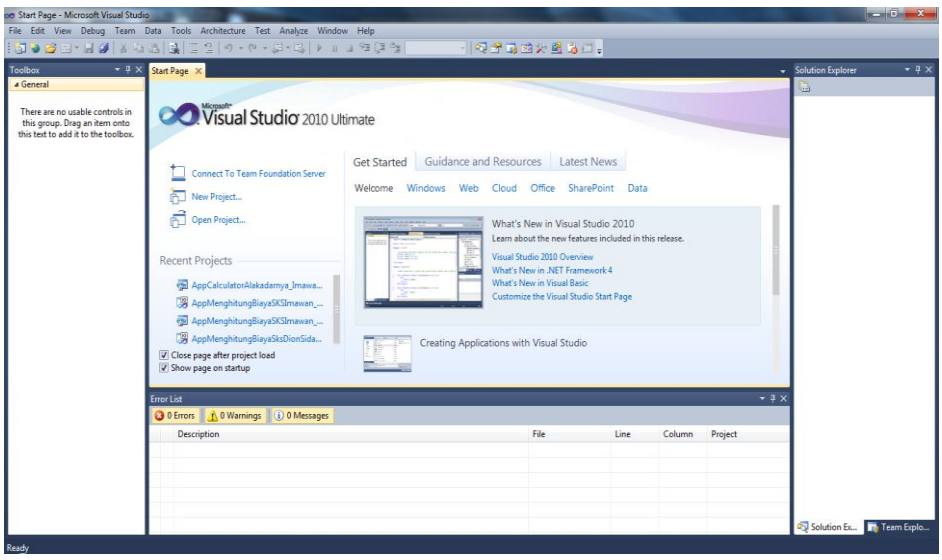
II.10. Visual Basic 2010

Visual Basic 2010 merupakan salah satu bagian dari produk pemrograman terbaru yang dikeluarkan oleh *Microsoft*, yaitu *Microsoft Viasual Basic* 2010. *Visual Studio* merupakan produk pemrograman andalan dari *Microsoft Corporation*, dimana di dalamnya berisi beberapa jenis IDE pemrograman seperti *Visual Basic, Visual C++, Visual Web Developer,Visual C #*, dan *Visual F#.*

Semua IDE pemrograman tersebut sudah mendukung penuh implementasi *.Net Framework 4.0* yang merupakan pengembangan dari *.Net Framework 3.5*. Adapun database standar yang disertaka adalah *Microsoft SQL Server 2008 expres*.

Visual Basic 2010 merupakan versi perbaikan dan pengembangan dari versi pendahulunya, yaitu *Visual Basic 2008.* Beberapa pengembangan yang terdapat di dalamnya antara lain dukungan terhadap *library* terbaru *Microsoft* , yaitu *.Net Framework 4.0*, dukungan terhadap pengembangan aplikasi menggunakan *Microsoft SilverLight*, dukungan terhadap aplikasi berbasis *Cloud Computing*, serta perluasan dukungan terhadap *database-database*, baik *standalone* maupun *database server*.

Bahasa *Visual Basic 2010* sendiri awalnya berasal dari bahasa pemrograman yang sangat popular di kalangan *programmer computer*, yaitu bahasa *Basic*, yang oleh *Microsoft* diadaptasi dalam program *Microsoft Qiuck Basic*, seiring dengan perkembangan teknologi komputasi dan desain, *Microsoft* mengeluarkan produk yang dinamakan *Microsoft Visual Studio*, yang didalamnya termasuk Visual Basic 2010 (Wahana Komputer ; 2010 : 2-3)



Gambar II.7. *Microsoft Visual Basic 2010*

Sumber : Wahana Komputer ; 2011 : 3

II.11. SQL Server 2008 R2

Microsoft SQL Server 2008 R2 adalah data *platform* yang paling canggih, terpercaya, dan terukur yang pernah dirilis hingga saat ini. Dibangun pada keberhasilan rilis *SQL Server 2008* yang asli, *SQL Server 2008 R2* telah memberi dampak pada organisasi di seluruh dunia dengan kemampuan inovatifnya, memberdayakan pengguna akhir melalui *intelijen bisnis self-service* (BI), memperkuat efisiensi dan kolaborasi antara *database administrator* (DBA) dan pengembang aplikasi, dan skala untuk mengakomodasi beban kerja data yang paling menuntut.

Sekarang, lebih dari sebelumnya, organisasi memerlukan database *platform* yang dapat dipercaya, hemat biaya, dan terukur yang menawarkan efisiensi dan dikelola secara *self-service* BI. Wajah organisasi-organisasi selalu berubah sesuai kondisi bisnis dalam ekonomi global, kendala anggaran IT, dan kebutuhan untuk tetap kompetitif dengan mendapatkan dan memanfaatkan informasi yang tepat pada waktu yang tepat. Dengan *SQL Server 2008 R2*, mereka dapat mengatasi tekanan untuk mencapai ini. Rilis ini memberikan sebuah platform database kelas *enterprise* pemenang penghargaan dengan kemampuan yang kuat yang meningkatkan efisiensi melalui pemanfaatan sumber daya yang lebih baik, pemberdayaan pengguna akhir, dan *scaling* keluar dengan biaya lebih rendah. Perangkat tambahan untuk skalabilitas dan kinerja, ketersediaan tinggi, keamanan perusahaan, pengelolaan perusahaan, data *warehouse*, pelaporan, *self-service* BI, kolaborasi, dan integrasi ketat dengan *Microsoft Visual Studio 2010*, *Microsoft SharePoint 2010*, dan *SQL Server PowerPivot* untuk *SharePoint*

membuat *platform database* terbaik yang tersedia. *SQL Server 2008 R2* dianggap versi minor *upgrade* dari *SQL Server 2008*. Namun, untuk *upgrade* kecil menawarkan sejumlah terobosan besar terbaru, kemampuan yang DBA dapat dimanfaatkan.

Microsoft telah membuat investasi besar dalam produk *SQL Server* secara keseluruhan, namun, fitur-fitur baru dan kemampuan terobosan yang harus paling menarik DBA adalah kemajuan dalam aplikasi dan administrasi *multi-server*. Bagian ini memperkenalkan beberapa fitur-fitur baru dan kemampuan kelompok produk *SQL Server* telah melakukan investasi yang cukup besar dalam meningkatkan aplikasi dan kemampuan manajemen *multi-server*. Beberapa aplikasi utama dan perangkat tambahan administrasi *multi-server* yang memungkinkan organisasi untuk lebih baik dalam mengelola lingkungan *SQL Server* mereka masukkan, yakni :

a. SQL Server Utilitas

   Ini adalah fitur baru yang digunakan untuk memonitor pengelolaan terpusat dan mengelola aplikasi database dan contoh *SQL Server* dari satu antarmuka manajemen dikenal sebagai *Control Point Utility* (UCP). Contoh dari SQL Server,aplikasi *data-tier*, *file database*, dan *volume* dikelola dan dilihat dalam *SQL Server Utility*.

b. Utility Control Point (UCP)

   Sebagai titik penalaran pusat untuk *SQL Server Utility*, *Control Point Utilitas* mengumpulkan konfigurasi dan kinerja informasi dari kasus dikelolanya *SQL Server* setiap 15 menit. Setelah data telah dikumpulkan

dari kasus yang dikelola, *SQL Server Utilitas dashboard* dan sudut pandang dalam *SQL Server Management Studio (SSMS)* menyediakan DBA dengan ringkasan kesehatan SQL Server sumber daya melalui evaluasi kebijakan dan analisis sejarah.

c. Aplikasi data-tier

Sebuah aplikasi *data-tier* (DAC) adalah satu unit penyebaran yang berisi semua skema database, dan persyaratan penyebaran digunakan oleh aplikasi. DAC A dapat digunakan dalam salah satu dari dua cara: bisa ditulis dengan menggunakan Server proyek aplikasi *data-tier* SQL dalam *Visual Studio 2010,* atau dapat dibuat dengan mengekstraksi definisi DAC dari *database* yang sudah ada dengan *Ekstrak Wisaya Application Data-Tier* di SSMS. Melalui penggunaan DAC, penyebaran aplikasi data dan kolaborasi antara pengembang data tier dan DBA meningkat secara signifikan.

d. Utilitas Explorer dashboard

Dashboard dalam *SQL Server* Utilitas menawarkan DBA wawasan yang luar biasa dalam pemanfaatan sumber daya dan negara kesehatan untuk kasus dikelola yang *SQL Server* dan disebarkan aplikasi data-tier seluruh perusahaan. Sebelum pengenalan dari *SQL Server Utility,* DBA tidak memiliki alat yang ampuh disertakan dengan *SQL Server* untuk membantu mereka dalam memantau pemanfaatan sumber daya.

Tambahan aplikasi dan administrasi *multi-server SQL Server 2008 R2*

.

DBA harus menyadari kemampuan baru berikut:

a. Paralel Data Warehouse

Paralel Data Warehouse adalah alat yang sangat *scalable* untuk data perusahaan pergudangan. Ini terdiri dari kedua perangkat lunak dan perangkat keras yang dirancang untuk memenuhi kebutuhan data warehouse terbesar. Solusi ini memiliki kemampuan untuk skala ratusan terabyte dengan penggunaan teknologi baru secara besar-besaran, disebut sebagai besar-besaran pemrosesan paralel (MPP), dan melalui perangkat keras murah dikonfigurasi dalam sebuah hub dan *PowerPivot* akses data dalam pertanian. Pendekatan baru ini menjanjikan integrasi yang lebih baik dengan *SharePoint* sementara juga meningkatkan dukungan *SharePoint* dari *workbook PowerPivot* dipublikasikan ke *SharePoint*.

b. Edisi Premium SQL Server 2008 R2 memperkenalkan dua edisi premium baru memenuhi kebutuhan data *center* berskala besar dan gudang data

c. *Unicode* Kompresi *SQL Server 2008 R2* mendukung kompresi untuk tipe data *Unicode*. Tipe data kompresi adalah kompresi dukungan *unicode* dan tetap-panjang *nchar* (n) dan *nvarchar* (n) tipe data. Sayangnya, nilai yang disimpan *off* baris atau di *nvarchar* (max) kolom tidak dikompresi. Tingkat kompresi hingga 50 persen dalam ruang penyimpanan dapat dicapai (Ross Mistry, Stacia Misner ; 2010 : 3-11).