PEMBUATAN SISTEM INFORMASI DENGAN ANALISIS DAN PERANCANGAN BERORIENTASI OBJEK

Dewan Pelawi

Information Systems Department, School of Information Systems, Binus University
Jl. K.H. Syahdan No. 9, Palmerah, Jakarta Barat 11480
dewanpelawi2318@binus.ac.id; dewanpelawi@yahoo.com

ABSTRACT

Analysis of business processes by observing the existing system is the main issue to produce information systems according to the needs of the company. Any search of problem solutions when doing business process is performed by a system approach. Object Oriented Analysis and Design (OOAD) is one of the methods currently used to build information systems. The method includes the building of activity diagrams, event table, class diagram, usecase diagrams, usecase description, state chart diagrams, deployment and software architecture, first-cut design class diagram, simple sequence diagram (SSD), sequence diagram (first-cut, view layer, data access layer), communication diagram, updated design class diagram, package diagram, persistent object, and user interface. The results obtained after performing all phases of the method is an information system based on the needs of companies that is expected to improve the performance and effectiveness of the company's business processes and support the business goals.

***Keywords**: analysis, design, information system, OOAD*

ABSTRAK

Untuk menghasilkan sistem informasi sesuai dengan kebutuhan perusahaan, hal utama yang dilakukan adalah analisis proses bisnis dengan mengamati sistem berjalan yang ada. Pencarian solusi dari masalah ketika melakukan proses bisnis dilakukan dengan pendekatan sistem. Metode analisis dan perancangan berorientasi objek atau Object Oriented Analysis & Design (OOAD) adalah salah satu metode yang digunakan saat ini untuk membangun sistem informasi. Pembangunan sistem dengan metodologi ini meliputi pembuatan activity diagrams, event table, class diagram, usecase diagrams, usecase description, state chart diagrams, deployment and software architecture, first-cut design class diagram, simple sequence diagram (SSD), sequence diagram (first- cut, view layer, data access layer), communication diagram, updated design class diagram, package diagram, persistent object, dan user interface. Hasil yang diperoleh setelah melakukan seluruh tahapan ini adalah sebuah sistem informasi yang sesuai dengan kebutuhan perusahaan sehingga diharapkan dapat meningkatkan kinerja dan efektifitas dalam proses bisnis perusahaan dan juga menunjang tujuan bisnis.

***Kata kunci:** analisis, perancangan, sistem informasi, OOAD*

PENDAHULUAN

Perkembangan teknologi dan sistem informasi membawa perubahan pada dunia usaha. Kemampuan menyerap teknologi yang dimanfaatkan untuk menunjang operasional dan proses bisnis banyak dilakukan, trend penggunaan sistem informasi pada perusahaan terjadi. Sistem informasi yang digunakan terdiri dari yang sudah jadi dan di jual bebas di pasaran atau dibangun sendiri. Kelemahan dari sistem yang dibeli bebas dipasaran adalah kurang flexibel dan kurang sesuai dengan kebutuhan bisnis yang terjadi sedangkan keuntungannya adalah implementasi yang cepat dan dapat langsung diterapkan. Sementara jika membangun sistem sendiri maka sudah pasti disesuaikan dengan kebutuhan proses bisnis dan sesuai dengan permasalahan yang terjadi dan memang kelemahannya dibutuhkan waktu untuk pembangunannya.

Menurut McLeod and Schell (2004, p.9), sistem didefinisikan sebagai sekelompok elemen yang terintegrasi dengan maksud yang sama untuk mencapai suatu tujuan. Menurut Laoudon (2003, p.7), secara teknis system informasi didefinisikan sebagai kumpulan-kumpulan komponen yang saling berhubungan, yang mengumpulkan (atau mengambil), memproses, menyimpan, dan mendistribusikan informasi untuk membantu dalam pengambilan keputusan, pengkoordinasian, pengendalian analisis dan menampikannya dalam suatu organisasi. Informasi adalah hasil pengolahan data menjadi sesuatu yang berguna bagi *user* yang menggunakannya. Menurut O'Brien(2005, p7), sistem informasi adalah kombinasi dari orang-orang, perangkat keras, piranti lunak, jaringan komunikasi dan sumber data yang terorganisasi yang mengumpulkan, mengubah dan menyebarkan informasi dalam sebuah organisasi. Ada pun komponen dari sistem informasi adalah: (1) sumber daya manusia termasuk *end-user* dan IS specialist. *End-user* adalah orang yang menggunakan system informasi atau informasi yang dihasilkan dan orang yang mengembangkan dan menjalankan system informasi adalah IS *specialist*; (2) sumber daya perangkat keras merupakan komponen-komponen fisik seperti komputer, dan bagian-bagiannya yang digunakan untuk mendukung pemrosesan informasi; (3) sumber daya piranti lunak seperti *software*, aplikasi *software*, dan prosedur, merupakan semua kumpulan perintah pemrosesan informasi; (4) sumber daya data yang terdiri dari fakta dan gambaran dari sesuatu yang secara umum tidak dapat digunakan oleh *user*; (5) sumber daya jaringan merupakan sumber pokok dari system informasi. Sumber daya jaringan seperti media komunikasi dan jaringan pendukung.

Salah satu kunci sukses mendasar dalam mengembangkan sistem adalah daur hidup pengembangan system (SDLC). Organisasi dan perusahaan menggunakan informasi untuk mendukung berbagai macam proses yang dibutuhkan oleh perusahaan untuk menjalankan fungsinya. Terdapat berbagai jenis informasi yang memiliki kegunaan tersendiri dalam mendukung proses bisnis. Setiap sistem informasi memiliki daur hidup system sendiri. Selama daur hidup tersebut system informasi menerima ide-ide, dirancang, dibangun dan dikembangkan selama dalam pengembangan proyek dan akhirnya diproduksi untuk digunakan mendukung bisnis. Walaupun dalam penggunaan system masih menerima perubahan untuk perbaikan, update atau dimodifikasi. Proses untuk membangun, menyebarkan, menggunakan, dan merubah system informasi dinamakan daur hidup pengembangan system (SDLC).

METODE

Analisis dan perancangan system informasi berbasis objek adalah salah satu metodologi pengembangan system informasi yang digunakan untuk membangun system informasi perusahaan. Metodologi ini dibagi menjadi dua bagian yaitu analisis berorientasi objek (OOA) dan perancangan berorientasi objek (OOD) meliputi pembuatan *activity diagram, event table, class diagram, usecase diagram, usecase description, state chart diagram, deployment and software architecture, first-cut*

design class diagram, simple sequence diagram (SSD), sequence diagram (first-cut, view layer, data access layer), communication diagram, updated design class diagram, package diagram, persistent object, user interface. Langkah terakhir adalah melakukan coding berdasarkan langkah-langkah yang sudah disusun sebelumnya.

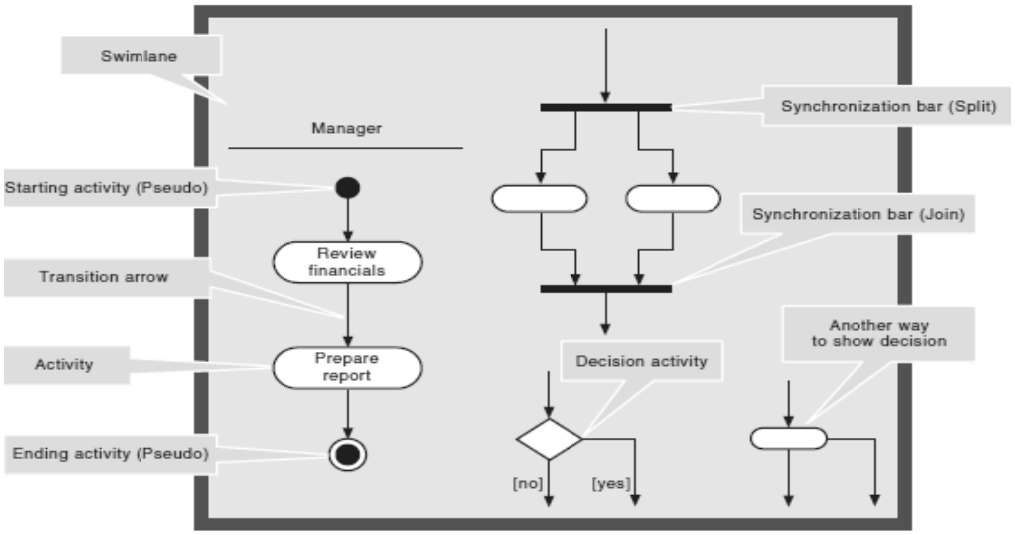
HASIL DAN PEMBAHASAN

Analisis berorientasi objek mendefenisikan seluruh tipe-tipe objek yang digunakan pada system dan menunjukkan kepada *user* kebutuhan yang diperlukan berinteraksi dengan system untuk menyelesaikan pekerjaan yang dilakukan. Teknik ini bertujuan untuk mempelajari objek yang ada dan mempertimbangkan apakah objek tersebut masih dapat digunakan lagi atau diambil lagi untuk penggunaan yang baru, juga digunakan untuk mendefenisikan objek-objek yang baru atau objek-objek yang sudah dimodifikasi yang akan digabungkan dengan objek yang sudah ada menjadi aplikasi komputasi yang berguna bagi bisnis. Objek adalah segala sesuatu yang memiliki *attribute* dan *behaviors*. Sebuah system computer dapat memiliki banyak tipe objek seperti objek *user* interface (UI) yang membuat *user* interface, *system domain* objek dan *problem domain* objek. *User interface* objek adalah objek-objek yang berinteraksi dengan *user* ketika menggunakan system seperti tombol-tombol (*button*), *menu item, text box* atau label. *Attribute* adalah karakteristik objek yang memiliki nilai seperti ukuran, bentuk, warna, lokasi, dan nama dari button atau label atau nama, alamat dan nomor telepon dari pelanggan. *Methods* adalah *behavior* atau operasi yang menggambarkan apa yang mampu dilakukan oleh objek. *Problem domain object* adalah objek yang khusus bagi aplikasi bisnis sebagai contoh objek pelanggan, objek order, objek produk dan sebagainya. *Class* adalah klasifikasi atau tipe objek yang memiliki *attribute* yang sama. Instan adalah sama dengan objek. *Messages* atau pesan adalah komunikasi antara satu objek dengan objek lainnya untuk menggunakan satu dari *method*nya.

Satzinger (2005, p.60) mendefinisikan perancangan berorientasi objek adalah mendefinisikan seluruh tipe objek-objek yang penting untuk berkomunikasi dengan manusia dan peralatan dalam system dan menunjukkan bagaimana objek-objek saling beriteraksi untuk menyelesaikan pekerjaan tertentu dan memperbaiki defenisi masing-masing tipe objek sehingga dapat diimplementasikan dengan bahasa khusus atau lingkungan khusus. Langkah – langkah OOA&D adalah membuat *activity diagram, event table, class diagram, usecase diagram, usecase description, state chart diagram, deployment and software architecture, first-cut design class diagram, simple sequence diagram (SSD), sequence diagram (first-cut, view layer, data access layer), communication diagram, updated design class diagram, package diagram, persistent object, user interface.*

Activity Diagram

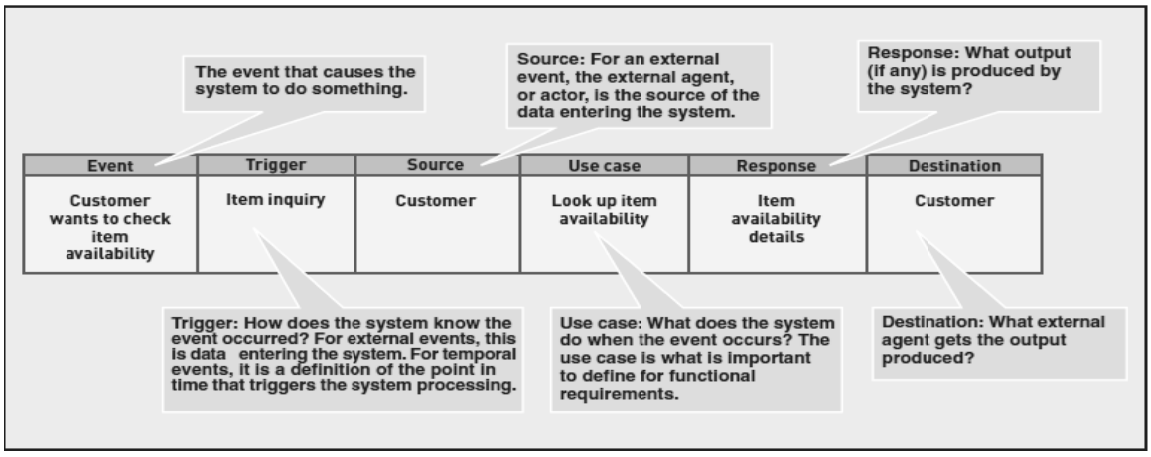
Activity diagram (Gambar 1) adalah sebuah tipe dari *workflow diagram* yang mendeskripsikan aktifitas *user* dan tahapan-tahapan pengerjaannya secara sekuensial. Diagram ini dimulai digambarkan dengan sebuah bulatan hitam kecil yang disebut sebagai awal kegiatan kemudian dilanjutkan dengan kegiatan/*activity* yang dilakukan yang digambarkan dengan suatu elips dan dihubungkan dengan konektor berupa tanda panah yang menunjukkan arah dari kegiatan. Kegiatan diakhiri dengan bulatan hitam kecil yang dilingkari yang merupakan akhir dari kegiatan.



Gambar 1. Activity diagram (Satzinger et al., 2005).

Event Table

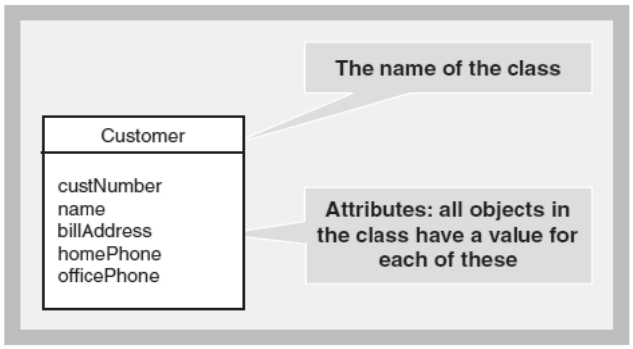
Event tabe terdiri dari *Event, Trigger , Source , Response, Use Case*, dan *Destination* (Gambar 2). *Event* adalah suatu kejadian di sebuah waktu tertentu dan tempat yang dapat digambarkan dan bernilai mengingat. *Trigger* adalah sinyal yang memberitahu sistem bahwa suatu peristiwa telah terjadi, baik kedatangan data membutuhkan pemrosesan atau suatu titik dalam waktu. *Source* adalah Agen eksternal atau aktor yang memasok data ke sistem. *Use Case* adalah apa yang sistem lakukan ketika peristiwa itu terjadi. *Response* adalah *output* dari sistem. Ketika sistem menghasilkan ringkasan laporan transaksi, laporan-laporan adalah output. Satu kasus digunakan dapat menghasilkan beberapa tanggapan. *Destination* adalah agen eksternal atau aktor yang menerima data dari sistem.



Gambar 2. Event table (Satzinger et al., 2005).

Class Diagram

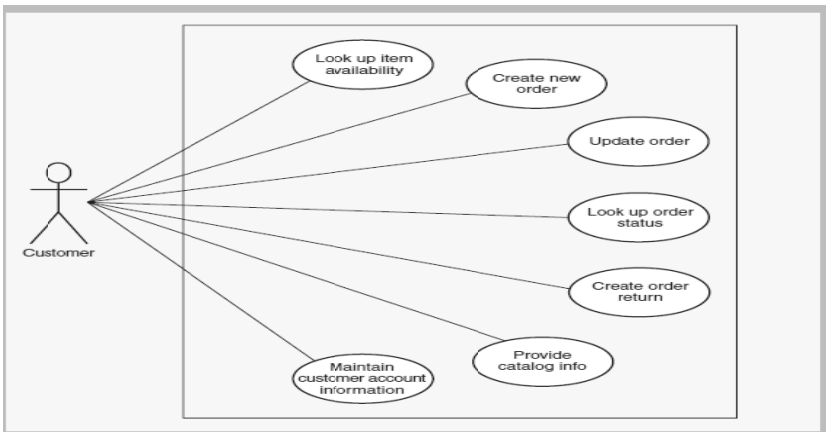
Class Diagram (Gambar 3) adalah sebuah model grafikal yang digunakan di dalam pendekatan object oriented untuk menunjukkan class-class yang ada di dalam sistem. Class terdiri dari nama *class*, *attribute* dari *class* dan digambarkan dalam bentuk persegi panjang yang dibagi menjadi dua bagian yaitu bagian judul dan atribut.



Gambar 3. Class Diagram (Satzinger et al., 2005).

Use Case Diagram

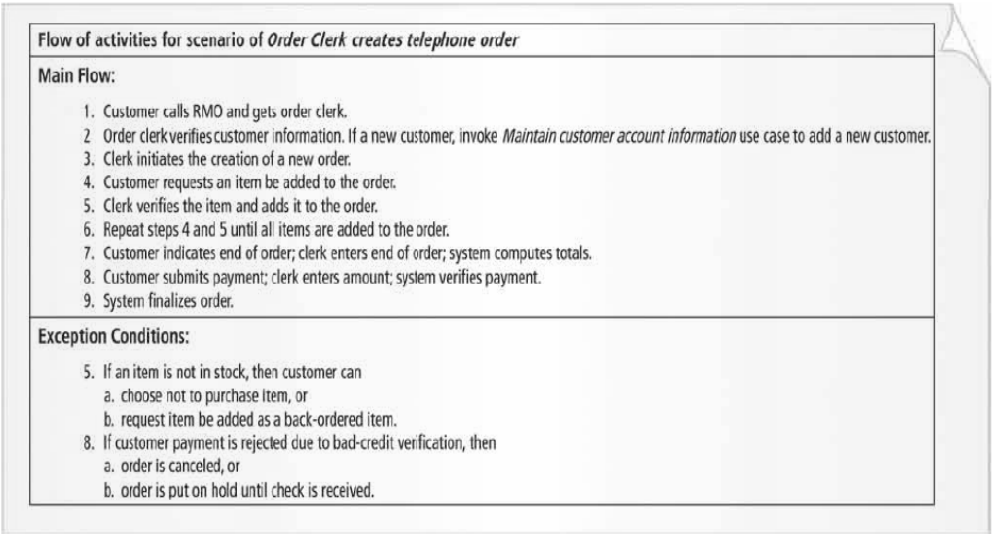
Use case diagram (Gambar 4) adalah sebuah diagram yang digunakan untuk menunjukkan beragam peran dari *user* dan bagaimana mereka menggunakan sistem yang ada. Komponen dari use *case diagram* adalah elips untuk menunjukkan *use case*, *actor* untuk menunjukkan *user* dan garis untuk menghubungkan *actor* dengan *use case*-nya.



Gambar 4. Use case diagram (Satzinger et al., 2005).

Use Case Description

Use case description adalah deskripsi yang berisi daftar rincian pengolahan untuk kasus penggunaan. Dalam diagram UML, sebuah orang yang menggunakan sistem.



Flow of activities for scenario of *Order Clerk creates telephone order*

Main Flow:

1. Customer calls RMO and gets order clerk.
2. Order clerk verifies customer information. If a new customer, invoke *Maintain customer account information* use case to add a new customer.
3. Clerk initiates the creation of a new order.
4. Customer requests an item be added to the order.
5. Clerk verifies the item and adds it to the order.
6. Repeat steps 4 and 5 until all items are added to the order.
7. Customer indicates end of order; clerk enters end of order; system computes totals.
8. Customer submits payment; clerk enters amount; system verifies payment.
9. System finalizes order.

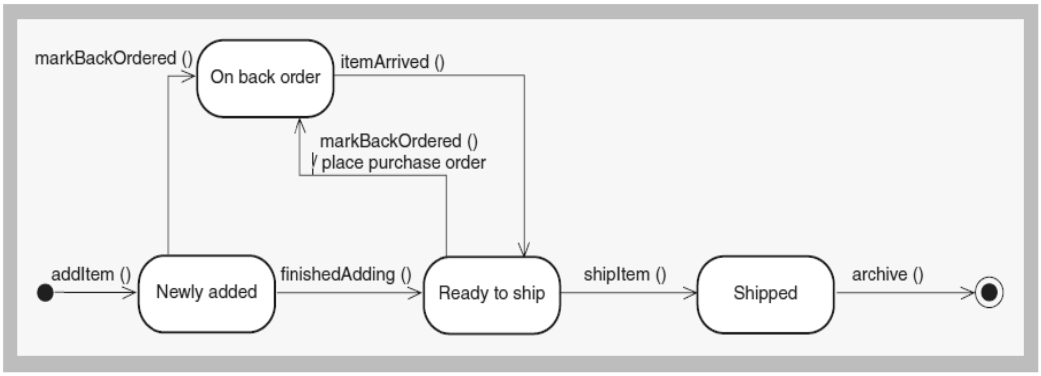
Exception Conditions:

5. If an item is not in stock, then customer can
   a. choose not to purchase item, or
   b. request item be added as a back-ordered item.
8. If customer payment is rejected due to bad-credit verification, then
   a. order is canceled, or
   b. order is put on hold until check is received.

Gambar 5. Use case description (Satzinger et al., 2005).

State Chart

Menurut Satzinger, *Statechart diagram* (Gambar 6) merupakan sebuah diagram yang menunjukan siklus hidup dari sebuah objek dalam *state* dan *transition*. *State chart diagram* membutuhkan kombinasi dari *states* dan *transistions* di antara *states* untuk menyediakan mekanisme yang analyst dapat gunakan untuk menangkap *business rules*. *State chart diagram* di bentuk dalam oval untuk menunjukkan *states* dari objek dan panah menunjukkan transisi. Kemudian dibentuk titik awal sebagai untuk permulaan yang disebut *Pseudostate*. Transisi dimulai sampai selesai dengan mengambil objek untuk *state* ke *state* baru.



Gambar 6. State chart (Satzinger et al., 2005).

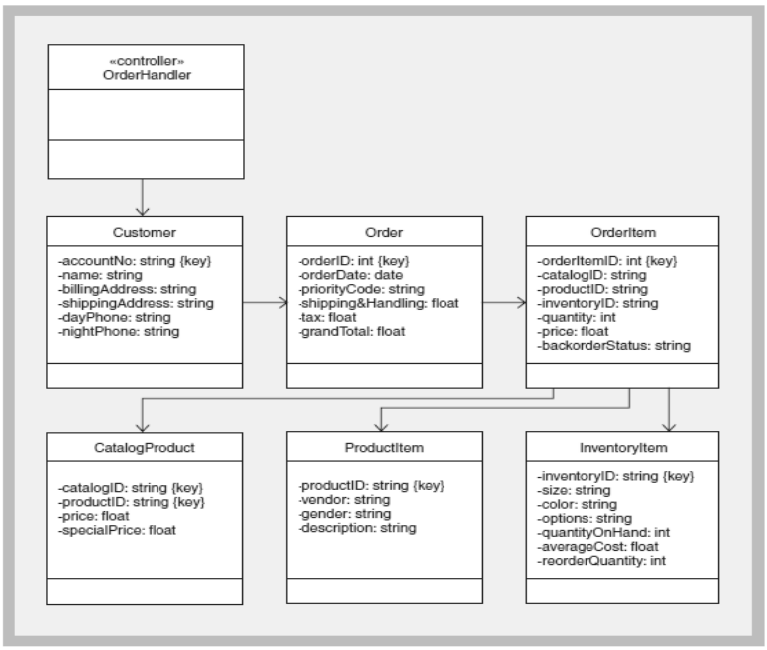
Deployment and Software Architecture

Deployment enviroment terdiri dari *Single Tier* yaitu Arsitektur yang memperkerjakan satu sistem computer yang mengeksekusi semua aplikasi perangkat lunak. *Multitier* adalah arsitektur yang mempekerjakan banyak sistem computer untuk mengeksekusi aplikasi perangkat lunak yang dibagi dua jenis, yaitu *arcitecture* dan *multi computer*. *Centralized architecture* adalah arsitektur yang menempatkan sumber daya komputasi semua di satu lokasi pusat. *Distributed architecture* adalah

Arsitektur yang menyebarkan sumber daya komputasi di beberapa lokasi yang terhubung dengan komputer jaringan.

First-Cut Design Class Diagram

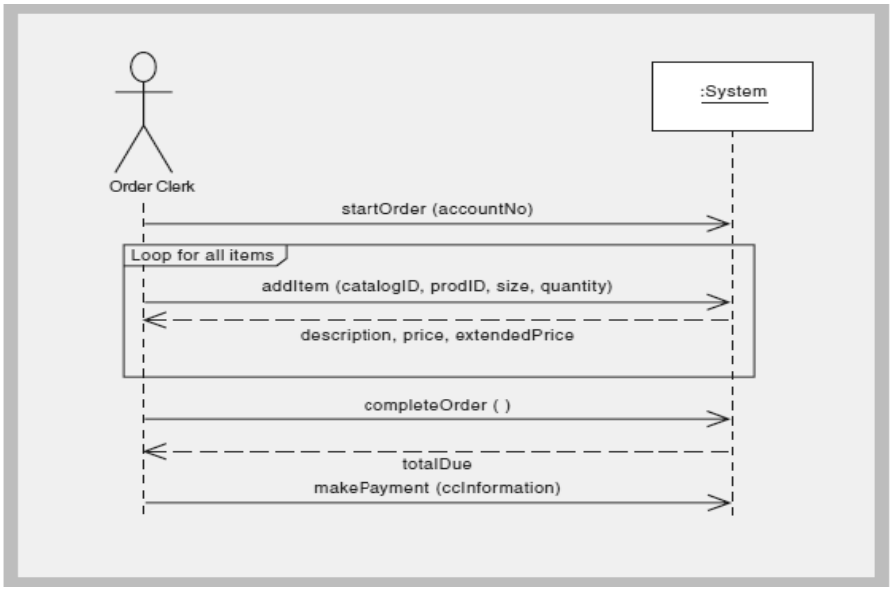
First-cut design class diagram (Gambar 7) dikembangkan dengan memperluas diagram domain model kelas. Hal ini membutuhkan dua langkah, yaitu menguraikan atribut dengan tipe dan informasi nilai awal, dan menambah visibilitas panah navigasi. Sebagaimana ditunjukkan sebelumnya, berorientasi objek. Berikut adalah beberapa pedoman umum: (1) satu-ke-banyak hubungan yang menunjukkan hubungan atasan / bawahan biasanya navigasikan dari atasan ke bawahan, misalnya, dari Order untuk OrderItem. Kadang-kadang hubungan bentuk hirarki rantai navigasi, misalnya, dari katalog untuk *product item* untuk *inventory item*; (2) wajib hubungan, di mana objek dalam satu kelas tidak bisa ada tanpa objek kelas lain, biasanya navigasikan dari kelas yang lebih independen kepada kelas yang tergantung pada kelas lainnya, misalnya, dari nasabah untuk order; (3) ketika sebuah objek mebutuhkan informasi dari obyek yang lain, sebuah panah navigasi mungkin diperlukan, baik untuk menunjuk obyek itu sendiri atau untuk *parent* dalam suatu hirarki; (4) navigasi panah juga dapat dua arah.



Gambar 7. First-cut design class diagram (Satzinger et al., 2005).

Simple Sequence Diagram

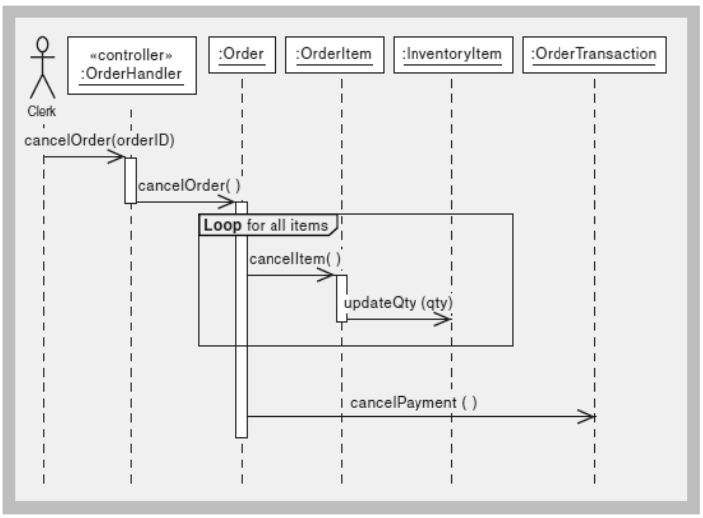
Simple sequence diagram (Gamabr 8) adalah sebuah diagram yang menunjukkan urutan pesan antara eksternal aktor dan sistem selama kasus penggunaan atau skenario.



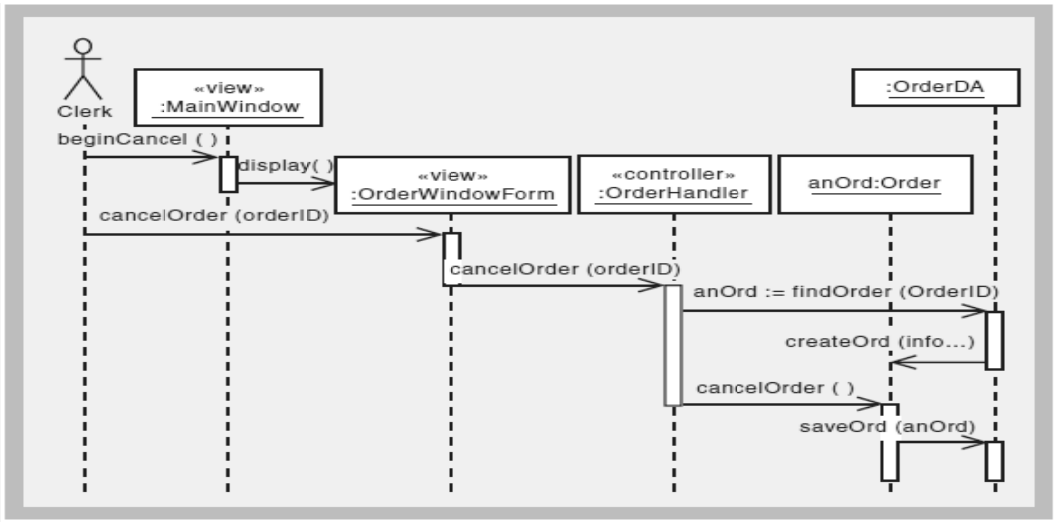
Gambar 8. Simple Sequence Diagram (Satzinger et al., 2005).

Sequence Diagram

Menurut Satzinger, Sebuah Sequence diagram rinci menggunakan semua elemen yang digunakan pada SSD. Perbedaannya adalah system objeck digantikan oleh semua objek internal dan pesan dalam system. Berikut adalah *first-cut sequence diagram* (Gambar 9).



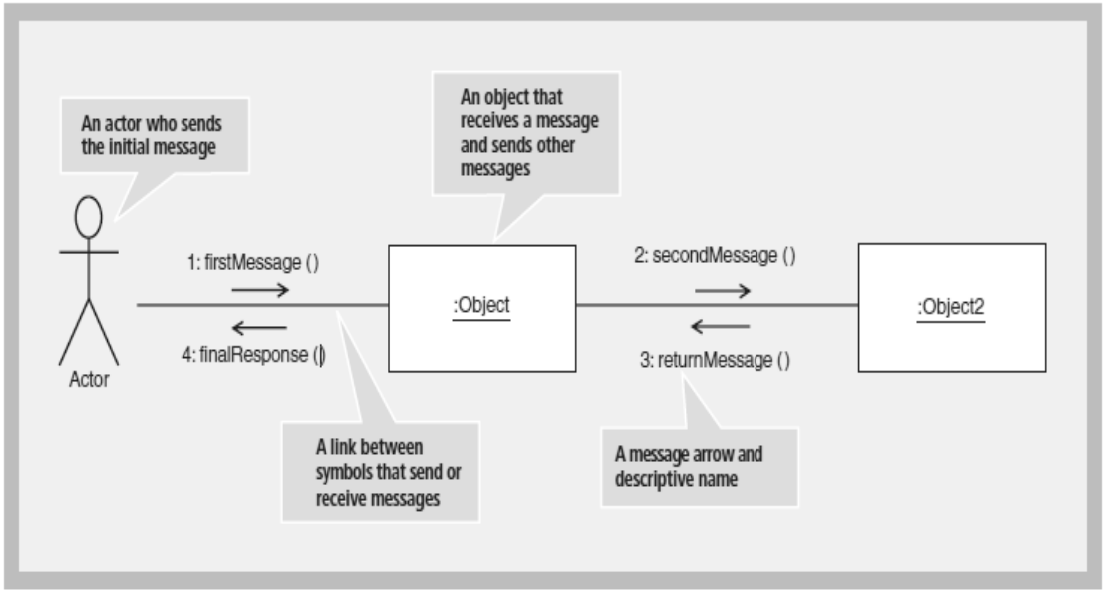
Langkah terakhir dalam desain multilayer dari *use case* adalah menambahkan *view layer* (Gambar 10). Pada kebanyakan *use case, view layer* terdiri dari *user interface* tunggal *window*. Pada beberapa *use case* yang lebih kompleks, dan membutuhkan beberapa *window* untuk masuk dan melihat data yang terkait dengan *use case* atau transaksi bisnis. Pada titik ini dalam desain, kita tidak masuk ke detail tentang isi dari *window* kelas itu sendiri. Cukup mengidentifikasi *window user* interface. Rinci desain *user* interface jauh lebih komppleks. Desain rinci dari kelas *window*, termasuk control *window* dan bentuk tata letak.



Gambar 10. View layer diagram (Satzinger et al., 2005).

Communication Diagram

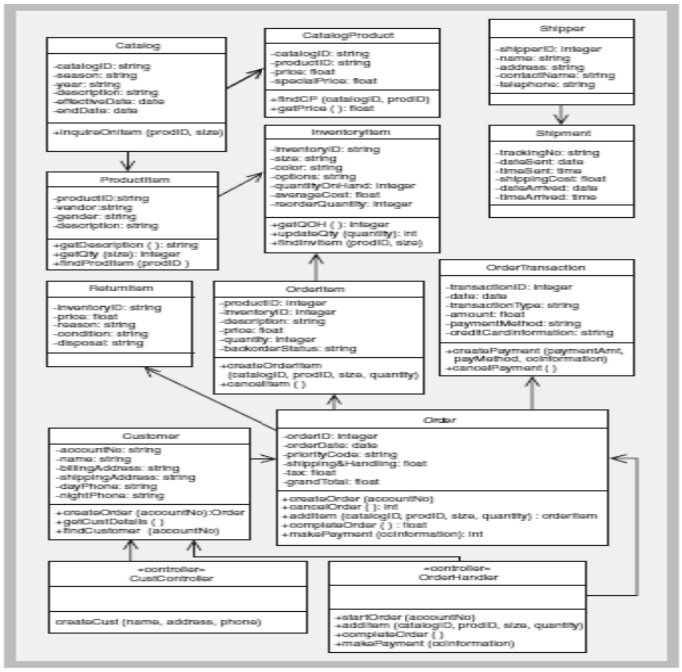
Communication diagram (Gambar 11) berhubungan dengan *Sequence Diagram*. *Communication diagram* menggunakan simbol yang sama untuk *actor, object* dan juga pesan sebagai *Sequence diagram.*



Gambar 11. Communication diagram (Satzinger et al., 2005).

Update Design Class Diagram

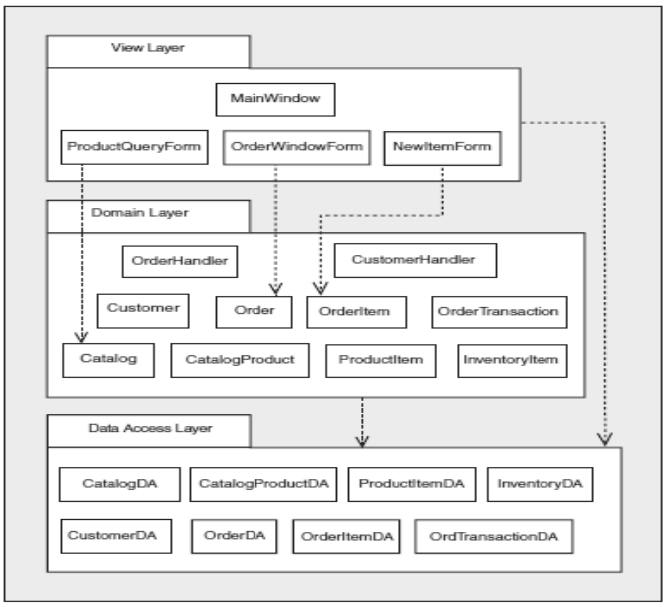
Update design class diagram (Gambar 12) adalah perbaikan class diagram yang sudah dibuat sebelumnya agar dihasilkan class diagram yang lebih sempurna.



Gambar 12. Update design class diagram (Satzinger et al., 2005).

Package Diagram

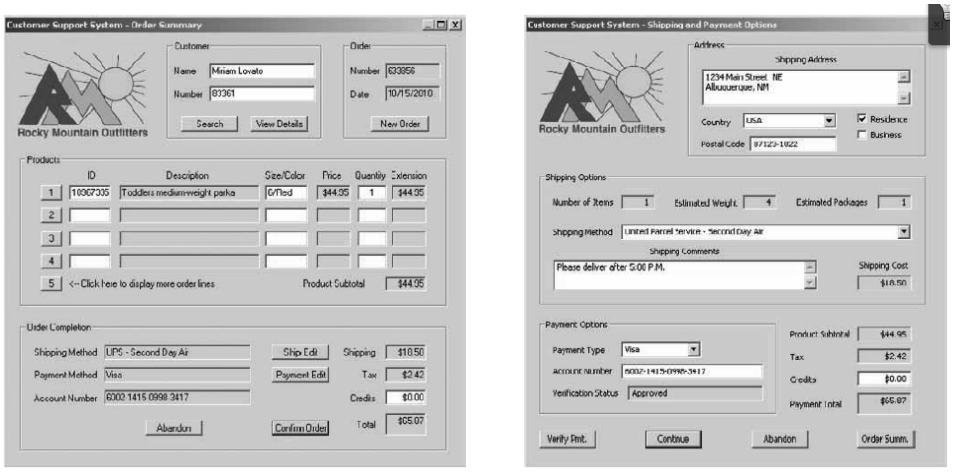
Package diagram (Gambar 13) menunjukkan komponen terkait dan dependensi. Umumnya, kita menggunakan *Package diagram* untuk berhubungan kelas atau komponen sistem lainnya seperti node jaringan. Angka-angka sebelumnya menunjukkan dua menggunakan *Package diagram* untuk membagi sistem menjadi subsistem dan untuk menunjukkan bersarang di dalam paket.



Gambar 13. Package Diagram (Satzinger et al., 2005).

User Interface

Bagian-bagian dari sistem informasi membutuhkan interaksi pengguna untuk membuat *input* dan *output*. Berikut adalah contoh rancangan *user interface* (Gambar 14).



Gambar 15. Contoh rancangan *user interface* (Satzinger et al., 2005).

PENUTUP

Analisis dan pengembangan sistem berorientasi objek terdiri dari dua bagian, yaitu analisis berorientasi objek dan perancangan berorientasi objek dimana kedua bagian tersebut pada intinya focus pada pembuatan *activity diagram, event table, class diagram, usecase diagram, usecase description, state chart diagram, deployment and software architecture, first-cut design class diagram, simple sequence diagram (SSD), sequence diagram (first-cut, view layer, data access layer), communication diagram, updated design class diagram, package diagram, persistent object,* dan *user interface* merupakan satu rangkaian kesatuan yang tidak bisa dipisahkan. Satu tahap kegiatan merupakan bagian untuk melanjutkan ke tahap berikutnya. Setelah selesai menggambarkan seluruh kebutuhan untuk pengembangan system maka diteruskan dengan programming yaitu merubahnya menjadi *source code* program kemudian dikompilasi menjadi program aplikasi yang siap untuk digunakan.

DAFTAR PUSTAKA

McLeod, Raymod Jr., and Schell, George. (2004*). Sistem Informasi Management* (edisi ke-8). Jakarta: Indeks.

O'brien, James A. (2005). *Introducing to Information System*. New York: McGraw Hill.

Satzinger, J., Jackson, R. &Burd, S. (2005). System Analysis and Design in Changing World (5th edition). Boston: Course Technology.