# BAB II

# TINJAUAN PUSTAKA

Sistem Informasi telah berkembang jauh, banyak digunakan oleh institusi dan organisasi sekarang. Komputer sebagai unsur pendukung dari perkembangannya, berkembang dari hanya sekedar alat pengolah data, kemudian berkembang menjadi unsur pendukung sistem informasi manajemen dan sekarang telah merupakan suatu alat yang strategis untuk menunjang kemajuan dan daya saing institusi dan organisasi.

Untuk mengetahui lebih jauh mengenai pemanfaatan analisis dan perancangan sistem informasi akademik perguruan tinggi, maka perlu ditinjau lebih dahulu mengenai penelitian-penelitian sebelumnya, yang berkaitan dengan masalah-masalah sistem akademik perguruan tinggi.

Salah satu sistem informasi yang ada di perguruan tinggi adalah sistem informasi akademik. Dalam sistem informasi akademik ini tidak hanya mencakup proses belajar mengajar saja, tetapi mencakup semua proses dari mulai seleksi calon mahasiswa sampai pelacakan lulusan. Sistem informasi akdemik sebagai inti dari sistem di perguruan tinggi dipandang sebagai sistem yang akan mampu berinteraksi atau mengintegrasikan semua aktivitas atau bagian yang terkait (Choldum 2006).

Yunis dan Krisdanto (2010) melakukan penelitian tentang implementasi arsitektur *enterprise* untuk perguruan tinggi. Implementasi ini memerlukan perencanaan yang matang dan harus disusun secara terintegrasi, tidak hanya

terfokus pada arsitektur aplikasi dan teknologi saja, tetapi juga harus menyeluruh pada semua domain arsitektur yang ada dalam perguruan tinggi. Selain perlu diperhatikan tantangan dan kendala yang mungkin ada pada saat pengembangan tersebut, juga diperhatikan perbaikan kualitas dari SDM di bidang TI/SI dalam perguruan tinggi secara berkelanjutan. Selain itu faktor eksternal yang mempengaruhi kebijakan perguruan tinggi dalam perancangan sistem informasi juga harus diperhatikan, dengan harapan nantinya akan menghasilkan suatu rancangan perangkat lunak dalam bentuk suatu dokumen yang berkualitas dan dapat digunakan untuk implementasi sistem.

Pada tahun 2010, Yunis dan Krisdanto mengemukakan tantangan terbesar yang dihadapi oleh perguruan tinggi di Indonesia adalah bagaimana perguruan tinggi menentukan arah dan sasaran pengembangannya sesuai dengan visi dan misi yang sudah ditetapkan bersama oleh penyelenggara perguruan tinggi, baik pemerintah maupun swasta (Yayasan).

Rahmat et al pada tahun 2010 dalam analisis perubahan manajemen terhadap implementasi sistem informasi/teknologi informasi pada perguruan tinggi, mendefenisikan secara lengkap kebutuhan sistem informasi dalam jangka panjang merupakan hal yang tidak mudah dan sangat beresiko. Tidak mudah karena tidak semua pengguna akademika sudah "*IT literate*". Antara rencana pengembangan teknologi informasi dalam jangka panjang dengan strategi pengembangan perguruan tinggi memiliki keterkaitan yang sangat erat. Bukan hal yang mudah menerjemahkan strategi menjadi perencanaan pengembangan sistem informasi. Selain itu juga beresiko karena kebutuhan bisa berubah-ubah baik

karena pengaruh pasar global, tetapi juga perubahan kebijakan organisasi internal bisa membuat spesifikasi kebutuhan juga ikut berubah

Pada tahun 2006 Musa dan Suhardi dalam sebuah prosiding konferensi nasional teknologi informasi mengemukakan bahwa dalam menerapkan suatu sistem informasi pada institusi pendidikan tingkat tinggi/universitas tidaklah mudah, karena akan berpengaruh terhadap perubahan struktur organisasi, manajemen, infrastruktur teknologi serta proses bisnis suatu universitas, sehingga membutuhkan kerja sama yang kuat antara pihak manajemen dengan unit-unit bisnis organisasi. Hal ini membutuhkan suatu konsep yang matang antara manajemen tingkat atas maupun operasional bersama tim IT dapat mempersatukan suatu persepsi terhadap proses bisnis yang ada agar analisis terhadap kebutuhan sesuai dengan yang diharapkan.

Pada tahun 2008 Adian dalam penelitiannya tentang perencanaan strategis sistem informasi perguruan tnggi mengemukakan bagiamana analisis dan perencanaan startegis sistem informasi yang dapat digunakan sebagai alat pendukung keberhasilan perguruan tinggi dalam mencapai visi dan misi organisasinya. Perencanaan strategis sistem informasi yang dilakukan meliputi: system akademik, sistem Praktek Kerja, sistem Skripsi, web perguruan tinggi, dan sistem bursa kerja.

Pada tahun 2008 Yulia dan Silvia melakukan penelitian tentang analisis dan desain Sistem Kerja Praktek. Dalam penelitian tersebut dikemukakan penggunaan UML dalam melakukan analisa dan desain sistem ini memberikan sarana bagi banyak pihak (koordinator Kerja Praktek, Tata Usaha, dan

Mahasiswa) untuk terlibat dalam perancangan karena, UML menyediakan berbagai diagram dengan perspektif yang berbeda-beda.

Pada tahun 2007 Peniarsih dalam penelitiannya tentang analisis dan perancangan sistem informasi akademik Universitas Suryadarma Jakarta, mengungkapkan bahwa dalam proses penyelenggaraan kegiatan akademik, dituntut adanya suatu kecepatan dan keakuratan dalam pengolahan data mahasiswa. Pengolahan data tersebut antara lain berupa pembuatan Biodata Mahasiswa, Rencana Studi , Nilai Ujian dan akan menghasilkan Kartu Hasil Studi.

Berikut perbandingan beberapa penelitian tentang sistem informasi akademik yag telah ada dengan penelitian yang akan dilakukan: (**a**)Yunis dan Krisdanto (2010), (**b**) Choldum (2006), (**c**) Yulia dan Silvia (2008), (**d**) Peniarsih (2007), (**e**) Adian (2008), (**f**) Penelitian yang akan dilakukan

Tabel 2.1 Perbandingan Penelitian

| **Fokus Penelitian** | **a** | **b** | **c** | **d** | **e** | **f** |
|---|---|---|---|---|---|---|
| Business Process Reengineering | V | V | V | | | v |
| Perencanaan Strategis | V | V | | | V | |
| Identifikasi Proses bisnis | | | | | V | V |
| **Fase Analisis Aistem** | | | | | | |
| 1. Analisis | | | V | V | | v |
| 2. Desain | | | V | V | | v |
| 3. Implementasi | | | | V | | |
| 4. Testing | | | | V | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| **Metode Analisis Pengembangan Sistem** | | | | | | |
| 1. Analisis dan Desain Terstruktur | | | | V | | |
| 2. Analisis dan Desain Berorientasi Objek | | | V | | | v |
| **Hasil Analisis dan Desain dalam bentuk dokumen** | | | | | | |
| 1. Spesifikasi Kebutuhan PL (SKPL) | | | | | | V |
| 2. Deskripsi Perancangan PL (DPPL) | | | | | | V |
| **Kebutuhan Penelitian** | | | | | | |
| 1. Penerimaan Mahasiswa Baru | | | | V | | V |
| 2. Proses Belajar Mengajar | | | | V | V | V |
| 3. Praktek Kerja Lapangan | | | | | V | V |
| 4. Skripsi | | | | | V | V |
| | | | | | | |

## 2.1. Tinjauan Teoritis Sistem Informasi Akadmeik

### 2.1.1 Pengertian Sistem

Sistem didefinisikan menjadi dua kelompok sistem, yaitu yang menekankan pada prosedurnya dan yang menekankan pada komponen atau elemennya. Pendekatan sistem yang lebih menekankan pada prosedur menurut Jogiyanto (1999): Sistem adalah suatu jaringan kerja prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran tertentu.

Pendekatan sistem yang merupakan jaringan kerja dari prosedur lebih menekankan urutan-urutan operasi didalam sistem. Sedangkan pendekatan sistem yang lebih menekankan pada komponen atau elemennya, menurut Davis (1985): Sistem adalah sekelompok elemen-elemen/bagian yang saling berhubungan atau terintegrasi dengan maksud yang sama untuk mencapai suatu tujuan tertentu. Sedangkan menurut Brient (2005) mendefinisikannya sebagai berikut: Sistem adalah sekelompok elemen-elemen yang saling terintegrasi dengan maksud yang sama untuk mencapai suatu tujuan.

Dari beberapa pendapat diatas, dapat diambil kesimpulan bahwa suatu sistem umum memiliki beberapa komponen, dimana masing-masing komponen tersebut saling bekerja sama untuk mencapai suatu tujuan. Berdasarkan definisi dari sistem, maka suatu sistem mempunyai elemen-elemen atau komponen yang mendukungnya yaitu:

1. Input

   Input adalah energi atau bahan baku yang dimasukkan ke dalam sistem.

2. Proses

   Proses adalah suatu sistem mempunyai suatu bagian pengolah yang akan merubah masukkan menjadi keluaran.

3. Output.

   Output adalah hasil dari energi atau bahan baku yang dapat dipergunakan oleh pihak lain dan diklasifikasikan menjadi keluaran yang berguna. Output dapat merupakan input untuk sub sistem yang lain.

4. Umpan Balik.

   Umpan balik merupakan keadaan yang terjadi terhadap sistem akibat dari penerapan suatu sistem. Umpan balik ini dapat menimbulkan keadaan yang menguntungkan atau mengganggu kelangsungan hidup sistem. Penilaian tentang keberhasilan sistem ditentukan oleh proses ini.

5. Mekanisme Kontrol.

   Mekanisme control adalah Kegiatan yang memfokuskan pada pengendalian terhadap pelaksanaan akan kerja didalam proses guna pencapaian sistem, namun yang terpenting dari pengendalian adalah pengendalian yang seminimal mungkin guna efisiensi dengan tingkat kualitas sistem yang tinggi.

6. Batasan.

   Batasan merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luamya.

7. Tujuan Sistem *(Goal System)*.

   Tujuan sistem adalah suatu sistem dapat mempunyai tujuan (*goal*) atau sasaran (*objective*). Kalau suatu sistem tidak mempunyai sasaran, maka operasi sistem tidak akan ada gunanya. Sasaran dari sistem sangat menentukan sekali masukan yang dibutuhkan sistem dan keluaran yang akan dihasilkan sistem. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuannya.

### 2.1.2 Konsep Data dan Informasi

Secara konseptual data dan informasi mempunyai arti yang berbeda. Data merupakan kata jamak dari datum yang berarti gambaran mengenai fakta,

statistik, dan lain sebagainya, yang belum memiliki makna atau arti, Sedangkan informasi didefinisikan sebagai kumpulan dari fakta, statistik dan lain-lain yang memiliki makna atau arti. Jadi yang membedakan data dan informasi adalah makna yang dikandungnya. Oleh karena itu tidak heran jika pemakaian kata data dan informasi sering kali dipertukarkan. Untuk lebih memperjelas perbedaan data dan informasi, maka dibawah ini dijelaskan definisi yang diberikan oleh Kendall dalam bukunya yang berjudul " *System Analysis and Design* " : Data adalah fakta dasar, data baru berarti jika sudah diolah dan dikaitkan dengan konteks tertentu. Informasi adalah suatu hasil pengolahan data dalam bentuk agregat untuk menghasilkan pengetahuan atau kemampuan.

Secara umum informasi dapat didefinisikan sebagai hasil dari pengolahan data dalam suatu bentuk yang lebih berguna dan lebih berarti bagi penerimanya yang menggambarkan suatu kejadian-kejadian yang nyata yang digunakan untuk pengambilan keputusan. Informasi merupakan data yang telah diklasifikasikan atau diolah atau diinterpretasi untuk digunakan dalam proses pengambilan keputusan.

### 2.1.3 Pengertian Sistem Informasi

Sistem informasi adalah suatu sistem dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian yang mendukung fungsi operasi organisasi yang bersifat manejerial dengan kegiatan strategi dari suatu organisasi untuk dapat menyediakan kepada pihak luar tertentu dengan informasi yang diperlukan untuk pengambilan keputusan. Sistem informasi dalam suatu

organisasi dapat dikatakan sebagai suatu sistem yang menyediakan informasi bagi semua tingkatan dalam organisasi tersebut kapan saja diperlukan. Sistem ini menyimpan, mengambil, mengubah, mengolah dan mengkomunikasikan informasi yang diterima, dengan menggunakan sistem informasi atau peralatan sistem lainnya.

Istilah sistem informasi menyiratkan suatu pengumpulan data yang terorganisasi beserta tata cara penggunaanya yang mencakup lebih jauh dari pada skedar penyajian. Istilah tersebut menyiratkan suatu maksud yang ingin dicapai dengan jalan memilih dan mengatur data serta meyususn tata cara penggunaannya. Keberhasilan sutu sistem informasi yang diukur berdasarkan maksud pembuatnya tentu tergantung pada tiga faktor utama, yaitu (1) keserasian dan mutu data, (2) pengorganisasian data dan tata cara penggunaannya.

Setiap sistem informasi meyajikan beberapa hal pokok : (1) pengumpulan dan pemasukan data, (2) penyimpanan dan pengambilan (*retrieval*) data, (3) penerapan data, yang dalam hal sistem informasi terkomputerisasi termasuk penayangan (*display*). Suatu sistem informasi terkomputerisasi pada dasarnya terdiri atas lima komponen yang menjadi sub sistemnya yaitu: (1) pelambangan (*encoding*) data dan pemrosesan masukan, (2) pengolahan data, (3) pengambilan kembali data, (4) pengolahan dan analisis data, dan (5) penayangan data.

Suatu sistem informasi dibuat untuk suatu keperluan tertentu atau untuk memenuhi permintaan pengguna tertentu, maka struktur dan cara kerja sistem informasi berbeda-beda tergantung pada macam keperluan atau macam permintaan yang harus dipenuhi. Oleh karena itu kepentingan harus dilayani

sangat beraneka ragam, maka jenis sistem informasi pun sangat beragam. Namun demikian, sistem informasi memepunyai banyak tampakan (*features*) umum dan menghadapai banyak persoalan yang mirip. Jadi disamping perbedaan yang jelas terdapat banyak persamaan antar berbagai sistem informasi. Suatu persamaan yang menonjol adalah semua sistem informasi menggabungkan berbagai ragam data yang dikumpulkan dari berbagai sumber.

Sistem informasi dapat di bentuk sesuai kebutuhan organisasi masing-masing. Oleh karena itu, untuk dapat menerapkan sistem yang efektif dan efisien diperlukan perencanaan, pelaksanaan, pengaturan, dan evaluasi sesuai keinginan masing-masing organisasi. Oleh karena itu sistem yang efektif dan efisien tidak lain untuk mendapatkan keunggulan dalam berkompetisi. Semua orang dapat menggunakan sistem informasi dalam organisasi, tetapi faktor efisiensi setiap sistem adalah berbeda. Perlu diketahui, perubahan sistem, baik besar maupun kecil, selalu akan melalui tingkatan-tingkatan sebagai berikut : (1) Ide, mengetahui perlu adanya perubahan, (2) Design, merancang cara pemecahannya, (3) Pelaksanaan, menerapkan desain ke dalam sistem. (3) Kontrol, memeriksa tingkat pelaksanaan dijalankan sesuai dengan desain, (4) Evaluasi, memeriksa apakah perubahan yang terjadi sesuai dengan tujuan semula, (5) Tindak lanjut, melaksanakn perubahan sesuai dengan hasil evaluasi yang ada.

### 2.1.4. Tahap dalam Pembentukan Sistem

Setiap sub sistem dari organisasi hidup-lahir, bertumbuh menjadi matang, berfungsi dan akhimya mati. Proses evolusi ini disebut siklus kehidupan sistem

atau *system development life cycle* (SDLC). Menurut Jogianto, proses pengembangan sistem mencakup lima tahapan, yaitu :

1. Tahap Perencanaan.

   Dalam tahap perencanaan merupakan tahapan awal yang dilakukan dalam proses perancangan suatu sistem. Pada tahap ini kegiatan yang dilakukan antara lain adalah: menyadari masalah, mendefinisikan masalah, menentukan tujuan sistem, mengidentifikasi kendala-kendala sistem, membuat studi kelayakan, mempersiapkan usulan penelitian sistem, menyetujui atau menolak penelitian proyek, menetapkan mekanisme pengendalian.

2. Tahap Analisis.

   Pada saat perencanaan telah selesai, tahap selanjutnya beralih pada analisis dari sistem yang telah ada. Analisis sistem adalah penelitian atas sistem yang telah ada dengan tujuan untuk merencanakan sistem yang baru atau diperbarui. Pada tahap ini kegiatan yang dilakukan adalah: mengumumkan penelitian sistem, mengorganisasikan tim proyek, mendefinisikan kebutuhan informasi, mendefinisikan kriteria kinerja sistem, menyiapkan usulan rancangan, menyetujui atau menolak rancangan proyek.

3. Tahap Rancangan.

   Dengan memahami sistem yang ada dan persyaratan-persyaratan sistem baru, kemudian beralih pada tahap membahas rancangan sistem baru. Rancangan sistem adalah penentuan proses dan data yang diperlukan oleh sistem baru. Ini biasanya digunakan suatu alat bantu untuk menggambarkan rancangan sistem yang akan dibuat. Alat bantu tersebut biasanya adalah *Data*

*Flow Diagram* atau *UML* kegiatan yang dikerjakan pada tahap ini antara lain adalah: menyiapkan rancangan sistem yang terinci, mengidentifikasikan berbagai alternatif konfigurasi sistem, mengevaluasi berbagai alternatif konfigurasi sistem, memilih konfigurasi yang terbaik, menyiapkan usulan penerapan, menyetujui atau menolak penerapan sistem.

4. Tahap Implementasi.

   Dalam tahap implementasi kegiatan memperoleh dan mengintegrasikan sumber daya fisik dan konseptual yang menghasilkan suatu sistem yang bekerja. Dalam kegiatan ini ada delapan tahapan kegiatan yaitu: merencanakan penerapan, mengumumkan penerapan, mendapatkan sumber daya perangkat keras, mendapatkan sumber daya perangkat lunak, menyiapkan *database*, menyiapkan fasilitas fisik, mendidik peserta dan pemakai, masuk ke sistem yang baru.

5. Tahap Penggunaan.

   Dalam tahapan ini mencakup 3 (tiga) kegiatan sekaligus, yaitu menggunakan sistem melakukan audit terhadap sistem yang bersangkutan dan melakukan perawatan terhadap sistem. Dalam menggunakan sistem, diharapkan pemakai menggunakan sistem sesuai dengan tujuan yang telah digariskan sebelumnya. Sedangkan pada kegiatan mengaudit sistem, dilakukan studi untuk mengetahui sampai sejauh mana sistem yang bersangkutan dapat memenuhi kriteria yang telah ditentukan sebelumnya. Kegiatan ini biasanya dilakukan berulang-ulang dengan periode tertentu. Pada kegiatan sistem selain dilakukan kegiatan yang berhubungan dengan perawatan sistem yang

bersangkutan, juga dilakukan modifikasi agar sistem tetap dapat mendukung penyelesaian pekerjaan yang diperlukan. Hal tersebut dilakukan antara lain untuk: menjaga agar sistem selalu '*Up-to-date*' dan sesuai dengan pekerjaan, meningkatkan kinerja karena adanya saran-saran baru yang lebih baik, dan memperbaiki kesalahan-kesalahan yang ada.

### 2.1.5. Sistem Informasi Akademik

Menurut Rahmat et al (2010) spesifikasi akomodasi kebutuhan sistem akademik Perguruan Tinggi didasarkan pada beberapa hal sebagai berikut:

1. Akomodasi terhadap kebutuhan yang mendasar yang sudah bisa dipenuhi oleh sistem yang lama.
2. Kebutuhan mendesak saat ini yang belum bisa dipenuhi oleh sistem yang lama.
3. Kebutuhan mendatang untuk mengantisipasi era informasi digital dan teknologi komunikasi yang semakin maju sehingga universitas dituntut untuk memperbaiki sistem informasi universitas dalam rangka meningkatkan "*competitive advantage*".
4. Sistem akademik adalah "*core system*" yang merupakan jantung dari sistem informasi yang terintegrasi dengan sistem-sistem lain.
5. Mengacu pada *Master Plan IT* Perguruan Tinggi dimana sistem informasi perguruan tinggi dikembangkan untuk mengantisipasi kebutuhan informasi dalam jangka panjang.

### 2.1.6. Tantangan Pengembangan Sistem Informasi Perguruan Tinggi

Perguruan Tinggi merupakan sistem yang kompleks yang terdiri dari banyak fakultas, jurusan, dan bagian-bagian yang lain (Budi, 2008). Pengembangan Sistem informasi dalam organisasi pada umumnya dihadapkan pada tantangan yang sangat besar yaitu dalam hal sedikitnya pemahaman organisasi terhadap budaya organisasi dan faktor sosial lainnya yang berhubungan dengan proses bisnis dalam organisasi. Pandangan ini sangat mempengaruhi keberlangsungan dari pengembangan sistem informasi yang akan dilakukan, ada beberapa studi awal yang harus di identifikasi organisasi sebelum mengembangkan sistem informasi, di antaranya:

a. Lingkungan bisnis, yang menjelaskan fungsi-fungsi bisnis yang ada dalam organisasi.
b. Hubungan komunikasi organisasi, menjelaskan hubungan komunikasi antara level organisasi, metode pengawasan, dan persepsi dari masing-masing pekerjaan.
c. Struktur organisasi dan proses bisnis organisasi, yang menggambarkan unsur keterlibatan komponen organisasi dalam skenario bisnis organisasi.

Begitu juga dengan halnya pada Perguruan Tinggi (PT), merupakan jenis organisasi yang spesifik, hal ini dipengaruhi proses bisnis dan fungsi bisnis yang ada dalam perguruan tinggi yang berbeda dengan organisasi lainnya. Berdasarkan eksplorasi pada beberapa sumber yang didapatkan dari berbagai jurnal dan penelitian yang sudah mencoba melakukan analisis dan perancangan sistem akademik untuk perguruan tinggi, didapatkan pemenuhan kebutuhan *stakeholder*

perguruan tinggi. Adapun kebutuhan *stakeholders* perguruan tinggi, yang terkait dengan SI menurut Yunis dan Krisdianto (2008), dapat dikelompokkan menjadi tiga, yaitu kebutuhan untuk mendukung:

a. Kegiatan operasional Tri Dharma dan manajemen perguruan tinggi, kebutuhan: meningkatkan produktivitas, efisiensi dan kualitas hasil kerja. Ciri data yang dikelola: transaksional, berasal dari transaksi-transaksi kegiatan.
b. Layanan informasi internal dan eksternal. Ciri data yang dikelola: sebagian besar berupa data detil (yang sering maupun tidak sering diubah), data dapat juga mengandung data multimedia.
c. Penjaminan mutu, kebutuhan: menyediakan data dan informasi untuk evaluasi mutu dan pengambilan keputusan. Ciri data: agregat (ringkasan) dari data operasional yang akurat, terkini *(up to date*), terstruktur dan terklasifikasi secara tertentu. Data perlu disajikan dalam bentuk yang ringkas, menarik dan mudah dibaca.

Untuk memenuhi kebutuhan tersebut, idealnya, sistem-sistem informasi di perguruan tinggi terintegrasi dan menjamin konsistensi, keakuratan dan kekinian data pada setiap sistem informasi. Pengembangan dan implementasi dari sistem yang terintgerasi, merupakan pekerjaan yang sangat besar, memerlukan sumber daya yang banyak dan membutuhkan waktu lama. Karena itu, perlu untuk memahami proses bisnis yang matang dan tantangan yang ada perlu di identifikasi agar dapat diatasi.

Dalam memberikan pelayanan yang baik kepada pelanggan pihak manajemen mampu mengelola bagaimana mereka menunjukkan perhatiannya

bagi kebutuhan pelanggan di segala kegiatan baik operasional ataupun dari segi TI sebagai operasional utamanya.

Menurut Yunis dan Krisdianto (2010) terdapat beberapa tantangan yang perlu diperhatikan oleh perguruan tinggi di dalam mengimplementasikan sistem informasi akademik, dapat dijelaskan sebagai berikut.

a. Pemahaman akan domain permasalahan

   Salah satu permasalahan yang sering dihadapi dalam implementasi SI adalah tidak bekerjanya sistem secara baik, terlalu banyaknya data dan tidak tersedianya informasi. Banyak diantara perguruan tinggi mengalokasikan dana untuk investasi TI untuk mengatasi, hal ini dikarenakan tidak baiknya dalam mengidentifikasi domain permasalahan. Untuk mengatasi hal ini, yang harus dilakukan adalah memperhatikan secara benar domain permasalahan, untuk itu diperlukan pengetahuan, pengalaman dan komunikasi dari profesional TI perguruan tinggi yang baik dalam mengidentifikasi permasalahan tersebut. Buruknya asumsi dalam mengidentifikasi masalah akan menghasilkan analisis kebutuhan yang salah.

b. Identifikasi proses bisnis

   Didalam lingkungan perguruan tinggi, proses bisnis merupakan identifikasi aktivitas fungsi bisnis yang berhubungan input, proses dan output. Untuk itu diperlukan cara yang benar bagaimana mendeskripsikan apa yang ada dalam proses, bukan bagaimana proses tersebut dilakukan. Pandangan ini perlu diverifikasi untuk kegiatan administrasi akademik, perkuliahan dan pengelolaan mahasiswa.

c. Munculnya standar, *framework* dan metode.

Dengan perkembangan ilmu pengetahuan dan teknologi, PT juga dihadapkan pada bagaimana PT tersebut mengadopsi pengetahuan tersebut. Pada saat ini cukup banyak standar, framework dan metode yang menjanjikan kualitas dari SI yang dikermbangkan, seperti IEEE, *Zachman Framework*, EAP, EA3, GEAF, TOGAF ADM dan lainnya. Dengan memilih dan membandingkan serta merujuk pada pengalaman tentang penggunaan standar dalam pengembangan SI, diharapkan dapat menghasilkan produk *Enterprise Architecture* (EA) atau SI yang berkulitas dan mampu beradaptasi dengan kebutuhan *stakeholder* PT.

d. Analisis kebutuhan

Analisis kebutuhan dalam pengembangan dan implementasi EA sangat dipengaruhi oleh kebutuhan dari *stakeholder* perguruan tinggi, analisis kebutuhan yang benar akan menghasilkan suatu sistem informasi yang efektif dan efisien, baik dari segi waktu, biaya dan tenaga.

Menurut Iwan (2008) pada prinsipnya ada tiga cara dalam memberikan keunggulan tersebut, yaitu:

1. Mampu memahami kebutuhan dan keinginan konsumen/pelanggan termasuk di dalamnya adalah memahami tipe-tipe pelanggan/konsumen.
2. Pengembangan basis data (*database*) yang lebih akurat dibandingkan yang dimiliki oleh pesaing (mencakup data keinginan dan kebutuhan setiap segmen pelanggan dan perubahan kondisi persaingan, bagaimana menempatkan *maturity* IT di dalamnya).

3. Pemanfaatan informasi-informasi yang diperoleh dari riset pasar dalam suatu kerangka strategis.

## 2.2. Tinjauan Teoritis *Object Oriented Analysis and Dsign* (OOAD)

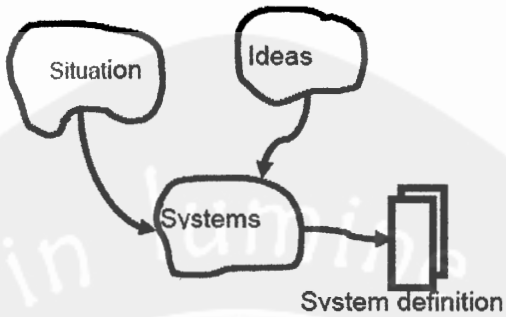
Analisis dan desain pemrograman beroreientasi objek karena dalam banyak kasus masih banyak ditemukan pemrograman yang dilakukan dengan bahasa pemrograman berorientasi objek namun analisis dan desain masih dilakukan mengikuti pendekatan terstruktur/fungsional sehingga sangat perlu dilakukan dalam pengembangan sistem berorientasi objek.

Perlu dibedakan apa yang menjadi tujuan proses analisis dan proses desain. Proses analisis bertujuan untuk memahami masalah, yaitu dengan dengan memahami apa yang sebenarnya ada dalam dunia nyata. Sedangkan proes desain bertujuan memahami pemecahan masalah yang didapatkan dari proses analisis, yaitu dengan mengusulkan secara detail sistem komputer seperti apa yang perlu dibangun untuk mengatasi suatu masalah. Proses analisis dan desain ini memang merupakan suatu proses yang saling berkelanjutan: proses analisis dulu dan kemudian baru proses desain (Julius, 2004)

### 2.2.1. Konsep *Object Oriented Analysis* (OOA)

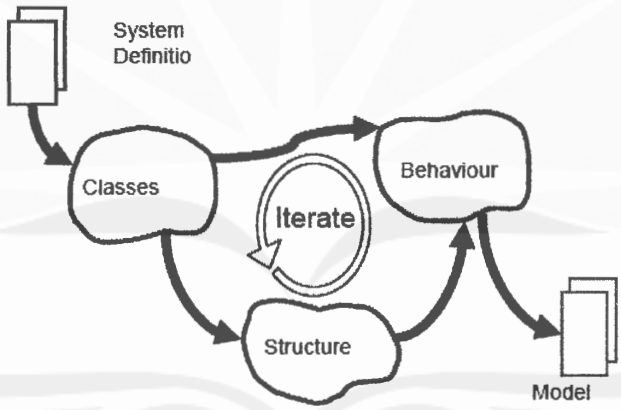
Aktivitas-aktivitas dalam OOA terdiri dari *system choice*, *Problem domain analysis*, *Application domain analysis*. Terdapat tiga aktivitas dalam *system choice* yaitu *Describe the situation*, *Create ideas* dan *Define systems*. Ketiga

aktivitas tersebut menghasilkan sebuah *system definition* yang memenuhi *factor criterion* seperti yang ditunjukkan dalam gambar 2.1 berikut:



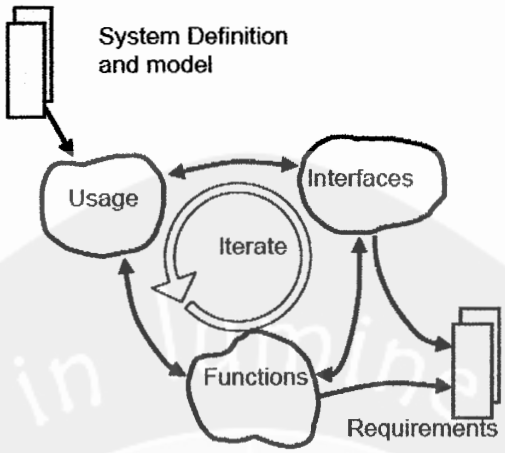

Gambar 2.1 *Procedure Sistem Choice* ( Sulistio, 2009)

*Problem Domain Analysis* meliputi tiga aktivitas utama yaitu *classes, Structure dan Behaviour* seperti yang ditunjukkan dalam gambar 2.2 berukut:




Gambar 2.2 *Activities in Problem Domain Analysis* (Sulistio, 2009)

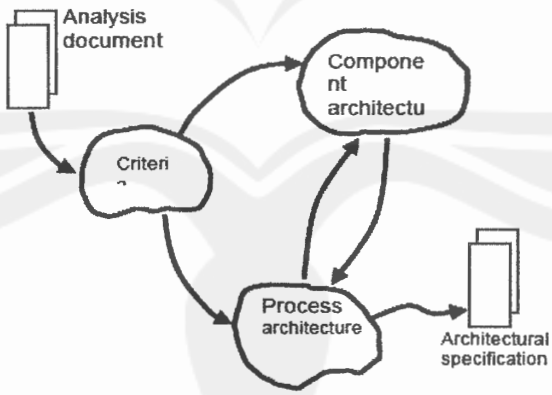
Terdapat 3 aktivitas dalam *application domain analysis*, yaitu *usage, function* dan *interface* yang ditunjukkan dalam gambar 2.3 berikut:




Gambar 2.3 *Application-domain analysis* (Sulistio, 2009)

### 2.2.2. Konsep *Object Oriented Design* (OOD)

Dalam OOD terdapat dua aktivitas utama yaitu *architectural design* dan *component design*. *Architectural design* terdiri dari tiga aktivitas yaitu *Criteria*, *Component Architecture* dan *Processes Architecture* seperti yang ditunjukkan dalam gambar 2.4.




Gambar 2.4 *Activities in architectural design* (Sulistio, 2009)

Tabel kriteria-kriteria yang akan menjadi evaluasi dari sebuah sistem. Kriteria yang diprioritaskan harus lebih diperhatikan (tabel 1).

Terdapat beberapa pola umum yang dapat digunakan untuk mendesain suatu *component architecture* yaitu:

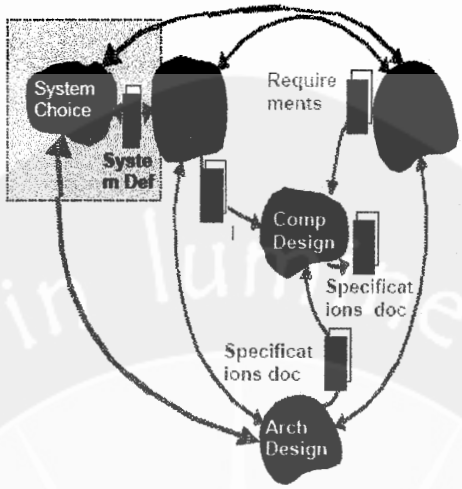
- *The Layered Architecture Pattern*
- *The Generic Architecture Pattern*
- *The Client Server Architecture Pattern*

Beberapa jenis distribusi dalam *Client Server Architecture*, seperti yang terlihat pada tabel 2.2.

Tabel 2.2 Kriteria klasik untuk kualitas perangkat lunak

| Kriteria | Ukuran dari |
|---|---|
| Usable | Kemampuan sistem untuk menyesuaikan diri dengan konteks, organisasi yang berhubungan dengan pekerjaan dan teknis. |
| Secure | Ukuran keamanan sistem dalam menghadapi akses yang tidak terotorisasi terhadap data dan fasilitas. |
| Efficient | Eksploitasi ekonomis terhadap fasilitas platform teknis |
| Correct | Pemenuhan dari kebutuhan |
| Reliable | Pemenuhan ketepatan yang dibutuhkan untuk melaksanakan fungsi. |
| Maintenable | Menemukan dan memperbaiki kerusakan |
| Testable | Memastikan sistem yang dibentuk dapat melaksanakan fungsi yang diinginkan |
| Fleksible | Dapat mengubah system dengan mudah. |
| Comprehensible | Mudah dimengerti |
| Reusable | Kemungkinan menggunkan bagian sistem pada sistem lain yang berhubungan |
| Portable | Dapat dioperasikan diteknikal platform yang lain. |
| Intereroperable | Dapat digabungkan dengan sistem yang lain. |

### *2.2.3. Konsep Object Oriented Analysis & Design*




Gambar 2.5 *Added Activities in OOAD* (Sulistio, 2009)

### 2.2.4. Pemodelan Visual Dengan *Unified Modeling Language* (UML)

#### *2.2.4.1 Unified Modeling Language* (UML)

*Unified Modeling Language* (UML) adalah salah satu alat yang sangat handal di dunia pengembangan sistem perangkat lunak yang berorientasi objek. Hal ini disebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembang sistem untuk membuat cetak biru atau visi mereka dalam bentuk yang baku, mudah dimengerti dan dilengkapi dengan mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancagan mereka dengan yang lain. UML merupakan kesatuan dari bahasa pemodelan yang dikembangkan oleh Booch, *Object Modeling Technique* (OMT) dan *Object Oriented Software Engineering* (OOSE). Metode Grady Booch sangat terkenal dengan nama motode *Desgin Object Oriented.* Metode ini menjadikan

proses analisis dan desain ke dalam empat tahap 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 detil dan semakin banyak notasi dan elemen. Pemodelan OMT yang dikembangkan oleh Rumbaugh didasarkan pada analisis terstruktur dan pemodelan *entiy relationship*. Tahap utama pada metode ini adalah analisis, desain sistem, desain objek dan implementasi.

Keunggulan dalam metode ini adalah dalam penotasian yang mendukung semua konsep OO. Metode OOSE dari Jacobson lebih memberi pendekatan pada Use Case. OOSE memiliki tiga tahapan yaitu membuat model *Requirement*, analisis, desain, 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 yang tidak praktis ditambah dengan elemen-elemen dari metode lain yang lebih efektif dan elemen-elem baru yang belum ada pada metode terdahulu sehingga UML lebih ekspresif dan seragam dari pada metode yang lainya. UML merupakan hasil kerja dari konsorsium dari berbagai organisasi yang berhasil dijadikan sebgaai standar baku dalam OOAD *(Object Oriented analysis Design)*. Kontribusi untuk UML telah dihasilkan banyak perusahaan-perusahaan ternama diantaranya Digital Equipment Corp, Hewlet-Packerd Company, I-Logic, Intellcorp, IBM, Intel Computing dan lain-lain.

### *2.2.4.2.Object*

Objek baik konkrit maupun konseptual selalu ada disekitar kita. Orang akan melihat sebuah mobil lebih sebagai objek dibandingkan dengan sesuatu/serangkaian proses yang membentuk mobil. Sebuah objek memiliki keadaan sesaat (*state*) dan perilaku (*behavior*). State sebuah objek adalah kondisi objek tersebut yang dinyatakan dalam atribut/*properties*. Sedangkan perilaku sebuah objek adalah bagaimana sebuah objek bertindak/bereaksi dan memberikan reaksi. Perilaku sebuah objek dinyatakan dalam *operation*. Atribut dan *operation* bila disatukan akan memberikan fitur/*feature*. Himpunan objek-objek sejenis disebut kelas. Objek adalah contoh/*instance* dari sebuah kelas. Aspek-aspek penting yang sering dibahas di UML yang berkaitan dengan objek sebagai berikut:

1. Abstraksi

   Abstraksi bertujuan untuk memfilter *properties* dan *operation* pada suatu objek, sehingga hanya tinggal *properties* dan *operation* yang dibutuhkan saja.

2. *Inheritance*

   Telah diuraikan diatas bahwa objek adalah contoh/*instance* dari sebuah *class*. Hal ini mempunyai konsekwensi yang penting yaitu sebagai *instance* disebut *class*, sebuah objek mempunyai semua karakteristik dari *classnya*. Inilah yang disebut dengan *inheritance* (pewarisan sifat). Dengan demikian apapun atribut dan *operation* dari *class* akan memiliki pula semua objek yang diinherit/diturunkan dari kelas tersebut. Sifat ini

tidak hanya berlaku untuk objek terhadap *class*, akan tetapi juga berlaku untuk *class* terhadap *class* yang lainya.

3. *Polymorphism*

   Kadangkala *class* yang berbeda mempunyai nama operasi yang berbeda. Meski kedenganrannya sama tapi, apa yang dilakukan berbeda. Konsep inilah yang disebeut *polymorphism*. *Polymorphism* adalah konsep yang sangat handal bagi pengembang perangkat lunak untuk pemisahan secara jelas diantara sub sistem yang berbeda. Dengan demikian sebuah sistem akan biasa dimodifikasi secara mudah karena hanya dibutuhkan *interface* antar *class*.

4. *Encapsulation*

   *Encapsulation* sering disebut dengan penyembunyian informasi (*information hiding*). Konsep ini lebih didasari pada fakta yang ada di dunia nyata bahwa tidak semua hal yang perlu dilihatkan. Di dunia nyata *encapsulation* membantu kita dalam melokalisir masalah. Jika suatu kesalahan/*error* yang terjadi pada sutu objek, kita mungkin hanya perlu memperbaiki objek tersebut tanpa perlu mengotak atik objek yang lain.

5. *Message Sending*

   Dalam sistem OO, Objek-objek saling berkomunikasi satu sama lain dengan mengirim pesan, suatu objek sebuah pesan kepada objek yang lain untuk menjalankan sebuah operasi dan objek menerima akan memberikan respon untuk menjalakan *operation* tersebut.

6. *Associaation*

   *Association* (asosiasi) adalah hubungan objek yang saling membutuhkan. Hubungan ini bisa satu arah ataupun lebih satu arah. *Multiplicity* adalah sebuah aspek penting dalam asosiasi dalam objek. Ini berkaitan dengan jumlah objek-objek dalam satu kelas yang saling berasosiasi.

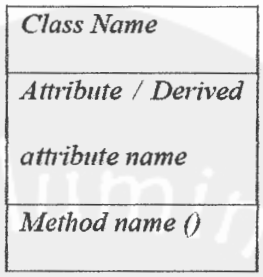
7. *Aggregation*

   *Aggregation* (agregasi) adalah bentuk khusus dari asosiasi yang menggambarkan seluruh bagian dari suatu objek merupakan bagian dari objek yang lain.

### *2.2.4.3.Class Diagram*

*Class Diagram* merupakan *class* yang selalu ada di pemodelan sistem berorientasi objek. *Class diagram* menunjukan hubungan antar *class* dalam sistem yang sedang dibangun, dan bagamana mereka saling berkolaborasi untuk mencapai suatu tujuan. *Class Diagram* tersusun dari elemen *class*: *Interface*, *Depedency*, *Generalization* dan *Acociation.* Relasi *depedency* menunjukan bagaimana ketergantungan terjadi antar *class* yang ada. Relasi *generalization* menunjukan bagaimana suatu *class* menjadi *superclass* dari *class* lainnya, dan *class* yang lain tersebut menjadi *subclass* dari *class* tersebut. Relasi *association* menggambarkan navigasi antar *class*, berapa banyak objek lain bisa berhubungan dengan satu objek, (*multiplicity* antar *class*), dan apakah suatu *class* menjadi bagian dari *class* lainnya (*aggregation*). *Class diagram* digunakan untuk

menggambarkan desain statis dari sistem yang sedang dibangun. Notasi *Class diagram* dapat ditunjukkan dalam gambar 2.6 berikut:



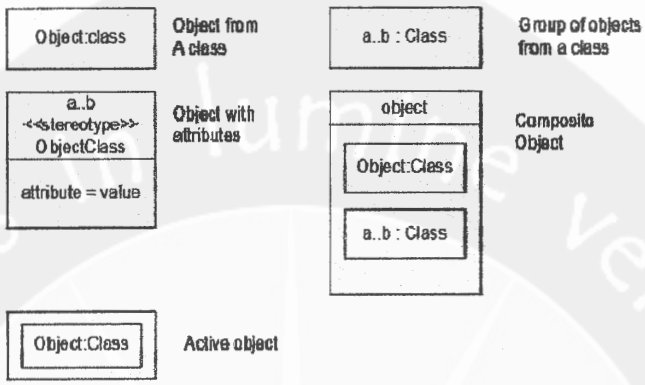

Gambar 2. 6 *Notation for Class Diagram*

***2.2.4.4.Object Diagram***

Diagram objek adalah gambaran objek-objek secara ringkas disebuah sistem pada sautu waktu. Objek diagram sering disebut sebagai *instance diagram* karena menunjukan *instance-instance* dari *class*. Objek diagram bisa digunakan untuk menunjukan contoh konfigurasi dari objek-objek. Hal ini sangat berguna untuk menunjukan hubungan yang mungkin ada diantara objek-objek yang sangat kompleks. Diagram objek bisa digunakan untuk memodelkan pandangan dari rancangan atau proses statis dari suatu sistem. Diagram objek tidak hanya penting untuk visualisasi, spesifikasi dan dokumentasi model struktur, akan tetapi juga penting untuk pembangunan/kontruksi aspek-aspek statis dari suatu sistem melalui *forward engineering* (pembuatan coding program dari sebuah model) maupun *reserve engineering* (pembuatan model dari coding program). Secara umum *object diagram* mengandung objek dan link. *Object diagram* bisa

mengandung *package* atau sub sistem dimana keduanya digunakan untuk mengelompokan elemen-elemen pada model ke bentuk yang lebih besar.

Notasi *Object diagram* dapat ditunjukkan dalam gambar 2.7 berikut:




Gambar 2.7 *Notation for Object Diagram*

### *2.2.4.5 Use Case*

*Use Case Diagram* (UCD) menjelaskan apa yang akan dilakukan oleh sistem yang akan dibangun dan siapa yang akan berinteraksi dengan sistem. Walaupun menjelaskan kegiatan namun hanya menjelaskan apa yang dilakukan oleh *actor* dan sistem, bukan menjelaskan bagaiman *actor* dan sistem melakukan kegiatan tersebut. UCD menjadi dokumen kesepakatan antara *customer, user dan Developer*. *User* menggunkan UCD ini untuk memahmi sistem dan mengevaluasi bahwa benar dilakukan sistem adalah untuk memecahkan masalah user ajukan atau sedang dihadapai. *Developer* mengunkan UCD ini sebagai rujukan yang benar dalam pengembangan sistem. *Use case diagram* pada umumnya tersusun dari beberapa elemen yaitu: *actor, use case, dependency, generalization*, dan

*association. Use case* ini memberikan gambaran sacara statis dari sistem yang sedang dibagun dan merupakan artifak dari proses analisis.

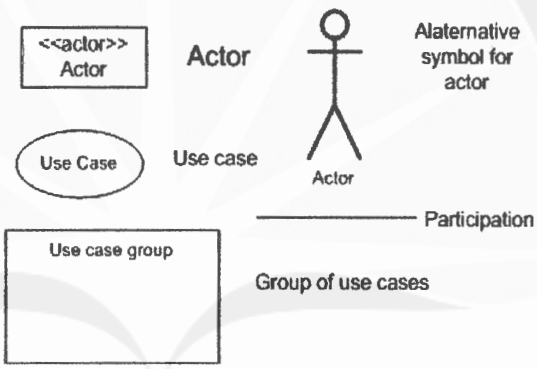
Dalam *use case* terdapat teks untuk menjelaskan urutan kegiatan yang disebut *Use Case specification. Use case specification* terdiri dari:

1. Nama *use case*: mencantumkan nama dari *use case* yang bersangkutan. Sebaiknya diawali dengan kata kerja untuk menunjukan suatu aktivitas.
2. Deskrispsi singkat (*Brief Description*): menjelaskan secara singkat satu atau dua baris/kalimat tentang tujuan dari *use case* ini.
3. Aliran normal (*basic Flow*): ini adalah jantung dari *use case*. Menjelaskan interaksi antar aktor dan sistem dalam kondisi normal, yaitu segala sesuatu bekerja dengan lancar. Tidak ada halangan atau hambatan dalam mencapai tujuan dari *use case*.
4. Aliran alternatif (*alternatif flow*): merupakan pelengkap dari *basic flow* karena tidak ada yang sempurna dalam setiap kali *use case* berlangsung. Didalam *alternatif flow* ini dijelaskan apa yang akan terjadi bila suatu halangan atau hambatan terjadi swaktu *use case* berlangsung. Ini terutama berhubungan dengan *error* yang mungkin terjadi, misalnya karena sistem kekurangan data untuk diolah (usia pegawai belum diinput), terjadi masalah internal sistem komputer (folder terhapus), terjadi masalah external (printer belum *turn of*).
5. *Special requirement*: berisi kebutuhan lain belum tercakup dalam aliran normal dan alternatif. Biasanya secara tegas dibedakan bahwa *basic flow* dan *alternatif flow* menangani kebutuhan fungsional dari *use case*, sementara

spesial *requirement* yang tidak berhubugan dengan kebutuhan fungsional, misalnya kecepatan transaksi maksimum berapa cepat dan berapa lama, kepastian akses yaitu jumlah user yang akan mengakses dalam waktu bersamaan.

6. *Pre-condition*: menjelaskan persyaratan yang harus dipenuhi sebelum *use case* bisa dimulai.
7. *Post-condition*: menjelaskan kondisi yang berubah terjadi saat *use case* selesai dieksekusi.

Notasi *Use case* dapat ditunjukkan dalam gambar 2.8 berikut:




Gambar 2.8 *Notation for Use Case Diagram*

### *2.2.4.6 Sequence Diagram*

*Sequence diagram* menjelaskan secara detil urutan proses yang dilakukan dalam sistem untuk mencapai tujuan dari *use case*: interaksi yang terjadi antar *class*, operasi apa saja yang terlibat, urutan antar operasi, dan informasi yang diperlukan oleh masing-masing operasi. Pembuatan *sequence diagram* merupakan aktivitas yang paling kritikal dari proses perancangan karena artifak inilah

menjadi pedomaan dalam proses implementasi nantinya dan berisi aliran control dari program

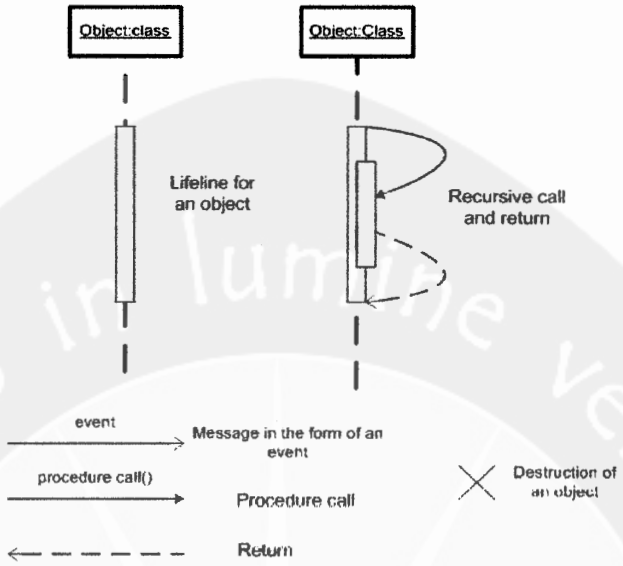
*Sequence diagram* ini berisi elemen: objek, *interaction* dan *message*. *Interaction* menggabungkan dua objek dangan pesannya. Diagram ini menjelaskan aspek dinamis dari sistem yang sedang dibangun.

Cara mudah yang biasa diikuti untuk memulai pembuatan *sequence diagram* adalah sebagai berikut:

1. Gambarkan aktor dan *class* yang terlibat dalam *sequence diagram*.
2. Urutkan sebagai berikut: *actor-object* dari *boundary class-object* dari *control class-object entity class.*
3. Ubah dari tipe analisis *class* menjadi *design class*
4. Ikuti urutan seperti dalam *use case specification* dan mulai identifikasi operas yang diperlukan untuk mengeksekusi dari suatu baris aktivitas dalam *use case specification*, operasi ini akan bergerak bertahap dari *actor* ke *boundry class*, dari *boundry class ke control class*, dari *boundry class* ke satu atau beberapa *entity class*.
5. Dari masing-masing operasi tersebut, identifikasi informasi apa saja yang yang perlu dipindahkan dari *actor* ke *boundry class* ke *control class* hingga ke *entiy class* ke *boundry class*.

   Untuk satu *use case* bisa dibuat beberapa *sequence diagram*, karena satu *use case* biasanya terdiri dari beberapa aktivitas yang harus dilakukan dan masing-masing aktivitas ini bisa direpresentasikan dalam satiap *sequence diagram.*

Notasi *sequence diagram* dapat ditunjukkan dalam gambar 2.10 berikut:




Gambar 2.10 *Notation for Sequence Diagram*

### *2.2.4.7 Colaboration diagram*

*Colaboration diagram* adalah perluasan dari objek diagram. Objek diagram menunjukan objek-objek dan hubungannya dengan satu dengan yang lain. *Colaboration diagram* menujukan *message-message* objek yang dikirimkan satu sama lain. Untuk menunjukan sebuah pesan, buatlah tanda pana didekat garis asosiasi diantara dua objek. Arah pana menunjukan objek yang menerima pesan. Label didekat nama menunjukan pesannya apa. Tipikal *message* meminta kepada objek yang menerimanya untuk menjalankan salah astu operasinya. Sepasang tanda kurung digunakan untuk mengakhiri *message*. Jika ada parameter bisa diletakan diantar tanda kurung.

Diantara *colaboration* dan *sequence diagram* bisa saling mengisi. Dengan demikian pada *colaboration diagram* bisa tambahkan nomor urut pada label

sebuah *message* untuk menunjukan urutan sebuah informasi. Titik dua (:) perlu digunakan untuk memisahkan nomor dengan *message*.

#### *2.2.4.8 Component Diagram*

*Component diagram* mereprensentasikan dunia rill item yaitu *component sotftware*. *Component sofware* menetap di komputer bukan dibenak para analisis. *Component diagram* mengandung *component*, *interface*, dan *relationship*. Hal penting pada komponen adalah *component* mewakili potongan-potongan yang independen yang bisa dipesan dan diperbaharui sewaktu-waktu. Pada UML sbelumnya, *component* digunakan untuk menunjukan struktur fisik seperti *Dynamic Library Link*. Hal tersebut tidak terlalu benar karena struktur fisik ditunjukan dengan artifak. Artifak adalah manifestasi fisik dari *sofware*, biasannya file. binner, dll, dokumen html, file-file data, file-file konfigurasi dan ain-lain.

*Component* dihubungkan melalui *interface* yang diimplementasikan. Biasanya menggunakan notasi *ball-and socket* seperti *class diagram*. *Component* juga bisa di dekompose dengan menggunakan *composite structure diagram.*

#### *2.2.4.9 Deployment Diagram*

*Deployment diagram* menunjukan tata letak sebuah sistem secara fisik, menampakan bagian-bagian *sofware* yang berjalan pada bagian-bagian *hardware*. Bagian utama *hardware* adalah *node*, yaitu nama umum untuk semua jenis sumber komputasi. Ada dua jenis tipe *node*, yaitu *processor* adalah *node* yang

bisa mengeksekusi sebuah *component*, sedangkan *device* tidak. *Device* adalah perangkat keras (printer, monitor dan lain-lain) tipikalnya menjadi *interface* di dunia luar. *Node* mengandung artifak dimana artifak adalah manifestasi fisik dari *software* biasanya *file*.

Pada UML, kubus menunjukan *node*. *Node* bisa diberi nama dan ditambahkan *stereotype* untuk mengindikasikan tipe *resource* yang ada didalamnya. Jika *node* adalah bagian dari *package*, namanya bisa mengandung nama dari *package* tersebut.