PERBANDINGAN METODE TERSTRUKTUR (TRADISIONAL) DAN METODE OBJECT-ORIENTED (OO) PADA ANALISIS DAN DESIGN SISTEM

OLEH :

SUPRIYANTO, S.Tp

Mahasiswa Pascasarjana Ilmu Komputer, Institut Pertanian Bogor (IPB)
Email : supriyanto_ipb@yahoo.com
http://supriliwa.wordpress.com

ABSTRAK

Metodologi yang umumnya digunakan dalam pembangunan sistem berbasis komputer dalam dunia bisnis dan industri saat ini adalah metode analisis dan design terstruktur (*Structured Analisys and Design* / SSAD). Metode ini diperkenalkan pada tahun 1970, yang merupakan hasil turunan dari pemrograman terstruktur. Metode pengembangan dengan metode terstruktur ini terus diperbaiki sampai akhirnya dapat digunakan dalam dunia nyata. Disamping itu, akhir-akhir ini bahasa pemrograman *object-oriented* (OO) mulai populer dan banyak digunakan pada organisasi bisnis maupun institusi pendidikan. Seiring dengan trend sebuah metodologi dibangun untuk membantu programmer dalam mengunakan bahasa pemrograman berorientasi obyek. Metodologi ini dikenal dengan object-oriented analysis and design (OOAD). Metode OOAD melakukan pendekatan terhadap masalah dari perspektif obyek, tidak pada perspektif fungsional seperti pada pemrograman tersrtuktur. Akhir-akhir ini penggunakan OOAD meningkat dibandingkan dengan pengunaan metode pengembangan software dengan metode tradisional. Sebagai metode baru dan sophisticated bahasa pemrograman berorientasi obyek diciptakan, hal tersebut untuk memenuhi peningkatan kebutuhan akan pendekatan berorientasi obyek pada aplikasi bisnis. Tidak ada jaminan bahwa menggunakan metode baru akan lebih sukses dibandingkan dengan metode tradisonal, maka tulisan ini akan mengulas keunggulan dan kekurangan dari kedua metode tersebut.

Kata Kunsi: Analisis dan Desain sistem, Object-Oriented Analysis and Design (OOAD), Waterfall Model, Systems Development Life Cycle (SDLC), Object-Oriented Life Cycle, Class Diagrams, Data Flow Diagrams.

1. PENDAHULUAN

Berbagai tulisan pada buku teks dan artikel terkait dengan Analisis dan Design Sistem, penulis banyak yang membahas tentang penggunaan *Object Oriented Analysis and Design* (OOAD) dibandingkan dengan metode *structured systems analysis and design*. Berikut adalah penulis-penulis yang mengulas tentang penggunaan Analisis dan design sistem dengan metode *Object-Oriented* : Rumbaugh, Blaha, Premerlani, Eddy and Lorensen, 1991; Embley, Kurtz and Woodfield, 1992; Gibson and Hughes, 1994, Norman, 1996; Dewitz, 1996; Bahrami, 1999; Dennis, Wixom and Tegarden, 2002; Brown, 2002; Satzinger, Jackson and Burd, 2005; Booch, 2007; Hoffer, George and Valacich, 2008.

Metodologi OOAD telah banyak digunakan dalam memcahkan permasalahan pengembangan sistem informasi, akan tetapi pada beberapa kasus metode OO tidak sesuai digunakan dan akan lebih baik jika menggunakan metode tradisional dalam analisis dan design sistem informasi. Tulisan ini bermaksud untuk memberikan gambaran tentang kedua metode tersebut, untuk membandingkan kelebihan dan kekurangan dari masing-masing metode. Tulisan ini akan memberikan rekomendasi penggunaan metode yang cocok untuk memecahkan berbagai permasalahan analisis dan design sistem informasi.

2. LATAR BELAKANG

Metodologi adalah prosedur dalam pemecahan masalah dari sistem yang ada untuk membangun sebuah sistem yang baru. Terdapat banyak metodologi yang dapat digunakan dalam design dan pengembangan sistem informasi seperti : *Systems Development Life* Cycle (SDLC), *Rapid Application Development* (RAD), *Object-Oriented Analysis and Design, Prototyping and many others* (Dennis, Wixom, Teagarden, 2002).

Metode SDLC dikenal dengan istilah *Structured Systems Analysis & Design* (SAD) atau metode terstruktur dalam analisis dan design sistem. Metode terstruktur memungkinkan seorang analis untuk menyederhanakan masalah menjadi lebih kecil, pendefinisian secara jelas dan memange berbagai hal yang mungkin. Metode SDLC terstruktur adalah metode pengembangan sistem dengan membuat tahapan-tahapan, dimana setelah selesai satu tahapan baru masuk ke tahapan selanjutnya. Metode ini juga sering dikenal dengan metode waterfall dikarenakan proses pembangunan sistem dilakukan secara bertahap dan tidak diperkenankan untuk kembali ke tahapan yang sudah dilakukan. Pada metode ini tahapan analisis, desin dan implementasi dilaksanakan secara sequensial.

Bahasa pemrograman object-oriented diperkanalkan pada tahun 1960 and 1970 dengan Simula dan Smalltalk. Beberapa tahun kemudian diperkenalkan metodologi *Object-Oriented Analysis and Design* (OOAD) (Larman, 2004). Tahun 1982 pertama kali metode OOAD diperkenalkan sebagai metode independen (G. Booch, 1982) dan terakhir pada akhir 1982OOAD diperkenalkan oleh S. Shlaer dan S. Mellor (1988) dan S. Bilin (1988).

Metodologi OOAD menggunakan persepektif *object-oriented* dibandingkan dengan perspektif fungsional seperti pada metodologi SSAD. Obyek adalah orang, tempat atau *thing* yang digambarkan dari domain permasalahan yang terdiri dari

tiga aspek yaitu : apa yang diketahui (identitas dan atributnya), siapa yang mengetahui (relationship dengan obyek lainnya), dan apa yang dilakukan (*method* yang yang bertugas melakukan aktivitas terhadap data) (Norman, 1998).

Analisis berorientasi obyek adalah proses membangun sebuah model *object-oriented* dari permasalhan dimana inisial obyek direpresentasikan sebagai entitas dan menghubungkan methord terhadap permasalhan yan akan diselesaikan.

OO Design adalah proses membangun sebuah model *object-oriented* dari sistem dalam rangka memenuhi spesifikasi. Jadi pada metodologi ini kita berfikir tentang sesuatu (obyek) dibandingkan fungsi.

3. TUJUAN

Penulisan ini bertujuan untuk membandingkan metode tradisional dan pendekatan berorientasi obyek dalam Analisis dan Design System.

4. ANALISIS DAN DESIGN SISTEM DENGAN METODE TERSTRUKTUR (TRADISIONAL)

System development life cycle (SDLC) atau metodologi *structured systems analysis & design* (SSAD), yang di Indonesia dikenal sebagai analisis dan design sistem dengan metodologi terstuktur. SDLC adalah sebuah *framework* dari aktivitas dan tugas yang dibutuhkan dalam pembuatan sistem informasi. Metodologi ini, sebelumnya lebih dikenal dengan istilah model waterfall dimana setiap fase dilaksanakan dengan aliran menurun menuju fase selanjutnya (Wu dan Wu, 1994).

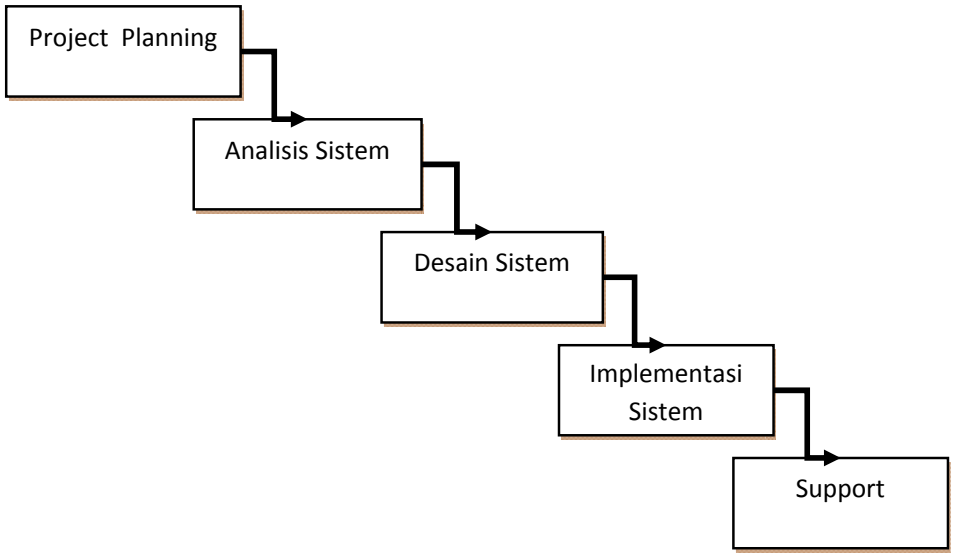
Konsekunsinya adalah metode ini adalah strategi yang terdiri dari berbagai macam teknik, tool, dokuemtasi, dan tugas yang perlu diintegrasikan dalam pembangunan sistem. SSAD adalah konsep dasar dalam membuat dan kemudian memecah komponen-komponen penting selama prose investigasi (Senn, 1989). Teknik dasar metode SSAD dapat dirangkum sebagai berikut :

a. Prinsip pertama dari SSAD adalah dekomposisi fungsional dengan pendekatan top down.
b. Selanjutnya scope dari sistem didefinisikan kedalam model fisik dari sistem yang dianalisis. Analis focus pada dua tujuan yaitu apa yang harus dilakukan oleh sistem yang akan dibangun dan bagaimana sistem akan bekerja (Davis, 1983).
c. Metodologi ini membutuhkan keterlibatan pengguna dalam pembangunan sistem dari awal sampai akhir. Seorang analis perlu bertemu dengan pengguna secara regular untuk memecahkan permasalhan dan menyesuaikan dengan kebutuhan pengguna. Hal ini mendorong seorang analis untuk memiliki kemampuan berkomunikasi yang baik (Bowman, 2004).
d. Dua konsen utama dalam pengembangan sebuah sistem informasi adalah proses dan data yang dimodelkan secara terpisah dari metodologi ini. Proses dimodelkan dengan mengunakan *Data Flow Diagram* (DFD) yang mengilustrasikan aliran data antara proses dan *data store* (media penyimpan data) dan bagaimana proses alirannya dair sumber data ke tujuan. Model data didefinisikan dengan mengunakan *entity-relationship diagram* (ERD) yang mendeskripsikan data (entitas) dan

berbagai asosisasi antar komponen-komponen tersebut.

e. Independensi antara modeling data dan proses berlanjut sampai pada tahap design. Skema dari model konseptual database dibangun, dinormalisasi dan dikumpulkan pada tahapan implementasi. Pada saat yang sama model ditransfromasikan ke dalam modul-modul untuk membangun sistem dan pada tahpan ini termasuk membuat detil logika program. Program kemudian dibangun berdasarkan structure chart dan logika program. Langkah terakhir, untuk memvalidasi sistem agar sesuai dengan kebutuhan pengguna, tujuan dan sasaran, maka perlu dilakukan tahapan pengujian.

Terdapat 5 fase yang sama dengan tahapan umum pemecahan masalah yang pada pendekatan terstruktur yang dikerjakan secara secara sequensial yang merupakan ciri utama pendekatan sistem prediktif. Gambar 1. dan Tabel 1. memperlihatkan tahapan-tahapan aktivitas yang dilaksanakan.



Gambar 1. SDLC dengan Pendekatan *Waterfall* (Satzinger *et al*. 2007)

5. ANALISIS DAN DESAIN SISTEM BERORIENTASI OBYEK (OBJECT ORIENTED ANALYSIS AND DESIGN)

Pendekatan berorientasi obyek memandang sistem informasi sebagai koleksi obyak yang bekerja bersama-sama untuk menyelesaikan pekerjaan (Satzinger et al, 2007). Secara konseptual tidak ada pemisahan antara proses dan program, tidak ada pemisahan antara entitas data atau file. Operasi sistem terdiri dari obyek-obyek. Sebuah obyek adalah “thing” atau benda dalam sistem computer yang dapat merespon pesan.

Metode OOAD dapat dibagi ke dalam dua tahapan utama yaitu :

a. Object-oriented analysis.

Pekerjaan ini konsen dalam pembangunan sebuah model *object-oriented* dari domain permasalahan. Pada tahapan ini

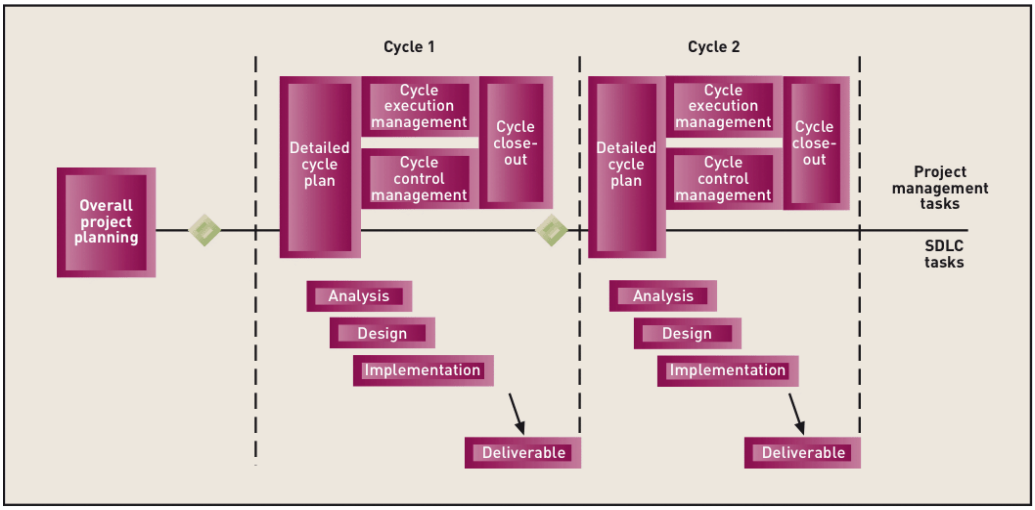
obyek yang diidentifikasi direpresenatikan sebagai entitas

b. Object-oriented design.
Tahapan desain konsen terhadap pembuatan model berorientasi obyek yang siap untuk diimplementasikan, dan memenuhi kebutuhan sistem. Seorang analis dan programmer harus berfikir dengan menggunakan terminologi thing (obyek) dibandingkan sebagai proses atau fungsi.

Permodelan obyek berbasis pada teknik-teknik OOAD yaitu abstraksi, pemodulan (*encapsulation*), mudularitas (*modularity*), penurunan (*inheritance)* dan *polymorphism* dan tahapan-tahapan berulang dalam pengembangan sistem (Rob, 2004). Fokus utama dari permodelan obyek adalah memecah obyek yang besar dan kompleks menjadi beberapa obyek. Sistem berorientasi obyek terdiri dari berbagai obyek yang saling berkolaborasi dalam melakukan pekerjaan.

Dekomposisi obyek memungkinkan seorang analis memecah permasalahan menjadi bagian-bagian kecil sehingga memudahkan dalam memanage bagian tersebut. Berikut adalah penjelasan lebih detail :

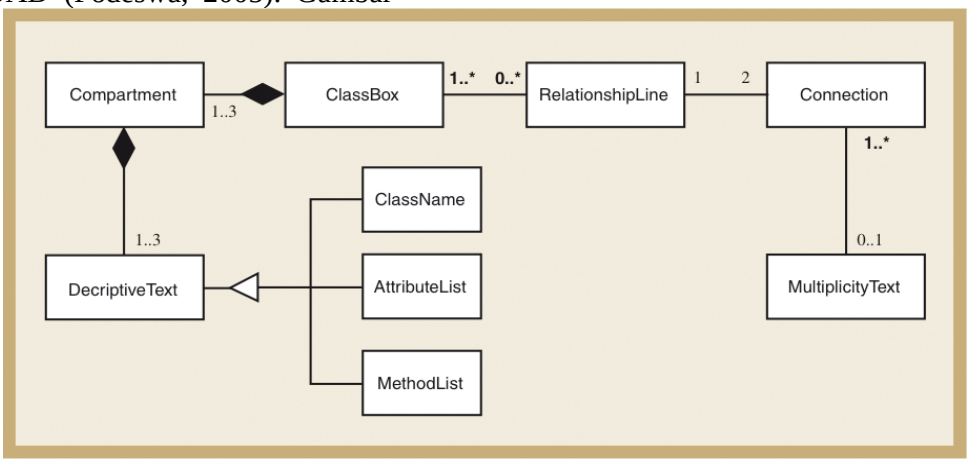
a. Kebalikan dari metode terstruktur, pengembangan sistem berorientasi obyek terdiri dari daur hidup (*lifecycle*) yang iteratif. Pengembangan sistem dengan OOAD, merupakan iterasi dari iterasi pekerjaan analisis, design, konstruksi (desain) dan implementasi, seperti tampak pada Gambar 2. Kebutuhan sistem didefinisikan dengan *use case diagram* yang merupakan tool OO yang digunakan untuk mendeskripsikan skenario interaksi sistem dengan pengguna.



Gambar 2. Pendekatan Iteratif pada Analisis dan Design Sistem (Satzinger, 2007)

b. Obyek direpresentasikan dengan entitas yang sama dikumpulkan dalam bentuk class. *Class* yang berhubungan satu sama lain didefinisikan dengan menggunakan teknik *inheritance* dan *agregasi*. *Class* dan berbagai relasi antar domain didokumentasikan dalam bentuk *class diagram*. Diagram ini merupakan bagian dari *Unified*

Modeling Language (UML) yang merupakan standar dari metode OOAD (Podeswa, 2005). Gambar 3. merupakan contoh *class diagram*.



Gambar 3. Class Diagram

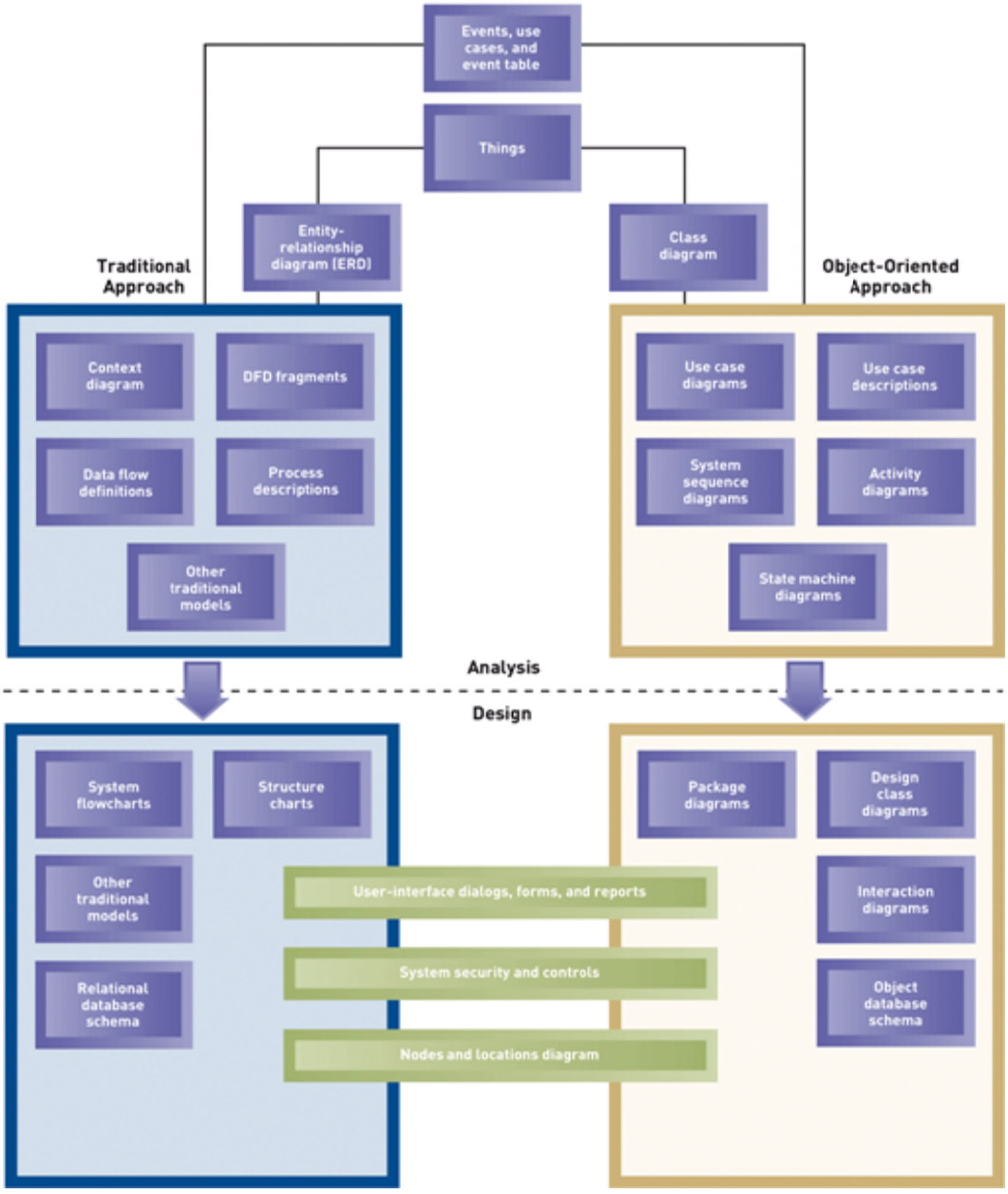
Pada fase desain *class diagram* diperluas menjadi design class. Pada Fase ini, masing-masing *class* didefinisikan, termasuk atribut dan *method* pada *class* tersebut. Proses diletakkan pada tiap-tiap obyek sebagai *method* atau *service*. Konsekuensinya adalah, obyek harus dikolaborasikan dengan obyek lain agar dapat mendeskripsikan berbagai skenario. Kolaborasi ini didokumentasikan dengan *interaction diagram*, dideskripsikan dengan *sequence diagram* atau *collaboration diagram*. Kemudian dibangun sistem informasi dengan menggunakan kombinasi *class* yang telah dibangun sebelumnya (*re-use*) dan dapat menambahkan *library* pada tahapan design. Pada tahap akhir analis dapat mengevaluasi hasil dan melakukan pengujian terhadap sistem. (Booch, 2005).

Berikut adalah rangkuman dari tiga komponen utama dari metodologi OOAD, yaitu :

a. **Permodelan Kebutuhan sistem (*Requirements Modeling*)**
   Mendeskripsikan siapa dan bagaimana sistem akan digunakan, menentukan aktor, use case dan skenario. Aktor adalah orang atau entitas yang berintaksi dengan sistem. Use case dibuat untuk untuk mendefinisikan perilaku dari sistem. Skenario adalah fakta-fakta atau keterangan dari use case yang mendeskripsikan kebutuhan spesifik dari sistem.
b. **Permodelan Informasi (*Information Modeling*)**
   Model menjelaskan entitas dan hubungan antar entitas dalam permasalahan yang akan diselesaikan, termasuk obyek, atribut dan berbagai macam relasi yang terkait.
c. **Permodelan Daur Hidup (*Life Cycle Modeling*)**
   Menjelaskan bagaimana obyek merespon lingkungan, dimana sistem dapat berubah sesuai dengan kebutuhan pengguna. Respon dari perubahan yang diinginkan oleh pengguna terhadap sistem pada suatu aktivitas yang spesifik yang diasosiasikan dengan memasukkan dan mengeluarkan fakta-fakta kejadian dari suatu (Hoover and Olekshy, 2001).

6. PERBANDINGAN ANTARA METODE TERSTRUKTUR DAN OOAD

Terdapat perbedaan dalam analisis dan desain dengan menggunakan pendekatan terstruktur (tradisional) dan metode berorientasi obyek. Pebedaan yang palin mendasar adalah pendekatan terstruktur lebih fokus pada proses sementara pada OOAD lebih fokus pada obyek. Hal ini mengakibatkan ada perbedaan dalam permodelan masalah antar kedua metode. Gambar 4. memperlihatkan perbandingan antara metode terstruktur dan OOAD, beserta model-model yang digunakan pada tahapan analisis dan desain sistem.



Gambar 4. Model pengembangan Terstruktur dan Berorientasi Obyek (Satzinger, 2007)

Tabel berikut ini menunjukkan perbandingan antara metode terstuktur dan *object-oriented* (Jadalowen, 2002) :

Tabel 1. Perbandingan Metode Terstruktur dan *Object Oriented*

Analisis dan Design dengan Metode Terstruktur (SSAD)	Analisis dan Design dengan Metode Berorientasi Obyek (OOAD)
SSAD disusun dengan menggunakan pemrograman terstruktur	OOAD disusun dengan mengunakan object Oriented Programming (OOP)
SSAD berorientasi pada proses, konsekuansinya adalah proses merupakan focus utama dari sistem	OOAD berorientasi pada data, konsekuensinya, data merupakan focus utama dari sistem
SSAD dipecahkan dengan menggunakan data flow diagram (DFD)	OOAD memecah sistem dengan menggunakan use case diagram
Komponen sistem merupakan turunan dari DFD	Untuk OOAD komponen utama sistem diturunkan dari class diagram dan Unified Modelling Language (UML. (Podeswa, 2005)
Tahapan-tahapan sudah dapat didefinisikan : planning, analisis, design dan implementasi dari awal sampai akhir SDLC	OOAD menerapkan pendekatan iteratif dan perbaikan terus menerus dari awal pembuatan sistem sampai sistem dapat digunakan.
Terdapat pemisahan antara sistem dan proses	Menggunakan teknik encapsulation data dan proses ke dalam sebuah obyek.

5. KELEBIHAN DAN KEKURANGAN SAAD

Berikut adalah rangkuman keuntungan dan kekurangan penggunaan metode terstruktur dalam analisis dan desain sistem :

a. Kelebihan

1) Milestone diperlihatkan dengan jelas yang memudahkan dalam manajemen proyek
2) SSAD merupakan pendekatan visual, ini membuat metode ini mudah dimengerti oleh pengguna atau programmer.
3) Penggunaan analisis grafis dan tool seperti DFD menjadikan SSAD menjadikan bagus untuk digunakan.
4) SSAD merupakan metode yang diketahui secara umum pada berbagai industry.
5) SSAD sudah diterapkan begitu lama sehingga metode ini sudah matang dan layak untuk digunakan.
6) SSAD memungkinkan untuk melakukan validasi antara berbagai kebutuhan
7) SSAD relatif simpel dan mudah dimengerti.

b. Kekurangan

1) SSAD berorientasi utama pada proses, sehingga mengabaikan kebutuhan non-fungsional.
2) Sedikit sekali manajemen langsung terkait dengan SSAD
3) Prinsip dasar SSAD merupakan pengembangan non-iterative (*waterfall*), akan tetapi kebutuhan akan berubah pada setiap proses.
4) Interaksi antara analisis atau pengguna tidak komprehensif, karena sistem telah didefinisikan dari awal, sehingga tidak adaptif

terhadap perubahan (kebutuhan-kebutuhan baru).

5) Selain dengan menggunakan desain logic dan DFD, tidak cukup tool yang digunakan untuk mengkomunikasikan dengan pengguna, sehingga sangat sulit bagi pengguna untuk melakukan evaluasi.
6) Pada SAAD sulit sekali untuk memutuskan ketika ingin menghentikan dekomposisi dan mulai membuat sistem.
7) SSAD tidak selalu memenuhi kebutuhan pengguna.
8) SSAD tidak dapat memenuhi kebutuhan terkait bahasa pemrograman berorientasi obyek, karena metode ini memang didesain untuk mendukung bahasa pemrograman terstruktur, tidak berorientasi pada obyek (Jadalowen, 2002).

6. KELEBIHAN DAN KEKURANGAN OOAD

a. Kelebihan

1) Dibandingkan dengan metode SSAD, OOAD lebih mudah digunakan dalam pembangunan sistem
2) Dibandingkan dengan SSAD, waktu pengembangan, level organisasi, ketangguhan,dan penggunaan kembali (*reuse*) kode program lebih tinggi dibandingkan dengan metode OOAD (Sommerville, 2000).
3) Tidak ada pemisahan antara fase desain dan analisis, sehingga meningkatkan komunikasi antara user dan developer dari awal hingga akhir pembangunan sistem.
4) Analis dan programmer tidak dibatasi dengan batasan implementasi sistem, jadi desain dapat diformulasikan yang dapat dikonfirmasi dengan berbagai lingkungan eksekusi.
5) Relasi obyek dengan entitas (*thing*) umumnya dapat di mapping dengan baik seperti kondisi pada dunia nyata dan keterkaitan dalam sistem. Hal ini memudahkan dalam mehami desain (Sommerville, 2000).
6) Memungkinkan adanya perubahan dan kepercayaan diri yang tinggi terhadap kebernaran software yang membantu untuk mengurangi resiko pada pembangunan sistem yang kompleks (Booch, 2007).
7) Encapsulation data dan method, memungkinkan penggunaan kembali pada proyek lain, hal ini akan memperingan proses desain, pemrograman dan reduksi harga.
8) OOAD memungkinkan adanya standarisasi obyek yang akan memudahkan memahami desain dan mengurangi resiko pelaksanaan proyek.
9) Dekomposisi obyek, memungkinkan seorang analis untuk memcah masalah menjadi pecahan-pecahan masalah dan bagian-bagian yang dimanage secara terpisah. Kode program dapat dikerjakan bersama-sama. Metode ini memungkinkan pembangunan software dengan cepat, sehingga dapat segera masuk ke pasaran dan kompetitif. Sistem yang dihasilkan sangat fleksibel dan mudah dalam memelihara.

b. Kekurangan

1) Pada awal desain OOAD, sistem mungkin akan sangat simple.
2) Pada OOAD lebih fockus pada *coding* dibandingkan dengan SSAD.
3) Pada OOAD tidak menekankan pada kinerja team seperti pada SSAD.
4) Pada OOAD tidak mudah untuk mendefinisikan class dan obyek yang dibutuhkan sistem.
5) Sering kali pemrogramam berorientasi obyek digunakan untuk melakukan anlisisis

terhadap fungsional siste, sementara metode OOAD tidak berbasis pada fungsional sistem.

6) OOAD merupakan jenis manajemen proyek yang tergolong baru, yang berbeda dengan metode analisis dengan metode terstruktur. Konsekuensinya adalah, team developer butuh waktu yang lebih lama untuk berpindah ke OOAD, karena mereka sudah menggunakan SSAD dalam waktu yang lama (Hantos, 2005).
7) Metodologi pengembangan sistem dengan OOAD menggunakan konsep *reuse*. *Reuse* merupakan salah satu keuntungan utama yang menjadi alasan digunakannya OOAD. Namun demikian, tanpa prosedur yang emplisit terhadap *reuse*, akan sangat sulit untuk menerapkan konsep ini pada skala besar (Hantos, 2005).

7. KESIMPULAN

Berdasarkan survey terakhir bahwa manager IT mengungkapkan bahwa 39% organisasi telah mengadopsi metodologi OO pada beberapa aplikasi, akan tetapi hanya 5% proyek IT dikembangkan menggunakan metodologi *object-oriented* (Sircar, Nerur, and Mahapatra, 2001). Pada berbagai aplikasi yang spesifik, pekerjaan pertama yang harus dilakukan adalah memilih metode yang tepat digunakan pada masalah tertentu.

Unified Process (UP) dirancang untuk digunakan pada pengembangan sistem informasi yang kompleks dan luas serta untuk menangani masalah pada *unified modeling language* (UML) (Sircar, 2001). Namun demikian perusahaan yang ingin berpindah dari metode SSAD ke metode OOAD, membutuhkan pemahaman terhadap perubahan yang substansial. Konsekuensinya adalah anlis dan programmer merubah pola fikir dari perspektif terhadap fungsional menjadi perspektif terhadap obyek. Lebih spesifik lagi dapat dikatakan bahwa, bagi programmer yang sudah terbiasa menggunakan SSAD perlu detraining untuk mempelajari sintax dan fitur pada bahasa pemrograman berorientasi obyek. Untuk memudahkan dalam memperlajari, maka perlu digunakan tool-tool yang berorientasi obyek (Sircar, 2001). Namun demikian penggunaan metode OOAD akan mendapatkan banyak keuntungan, yang tidak didapatkan pada metode tradisional SSAD.

DAFTAR PUSTAKA

Bahrami, A., Object-Oriented Systems Development, Irwin McGraw-Hill, Boston, Massachusetts, 1999.

Bailin, S., Remarks on Object-Oriented Requirements Specification, Computer Technology Associates, Laurel, MD, 1988.

Booch, G., "Object-Oriented Design", Ada Letters Volume 1 (3), 1982, 64-76.

Booch, G., Rumbaugh, J., and Jacobson, I., The Unified Modeling Language User Guide, 2nd ed., Addison-Wesley, Upper Saddle River, New Jersey, 2005.

Booch, G., Maksimchuk, R., Engle, M., Young, B., Conallen, J., Houston, K., Object-Oriented Analysis and Design with Applications, 3rd ed., Addison Wesley, Reading MA, 2007.

Bowman, Kevin, Systems Analysis: A Beginner's Guide, Palgrave Macmillan, Gordonsville, VA. 2004.

Brown, D., An Introduction to Object-Oriented Analysis, Objects and UML in Plain English, John Wiley and Sons, New York, New York, 2002.

Coad, P., and Yourdon, E., Object-Oriented Analysis, 2nd ed., Yourdon Press, Englewood Cliffs, New Jersey, 1991.

Davis, W., Systems Analysis and Design A Structured Approach, Addison-Wesley, Reading, Massachusetts, 1983.

Dennis, A., Wixom, B., and Tegarden, D., Systems Analysis and Design An Object-Oriented Approach with UML, John Wiley & sons, New York, New York, 2002.

Dewitz, S., Systems Analysis and Design and the Transition to Objects, Irwin McGraw-Hill, Boston, Massachusetts, 1996.

Embley, D., Kurtz, B., and Woodfield, S., Object-Oriented Systems Analysis A Model-Driven Approach, Yourdon Press Prentice Hall, Englewood Cliffs, New Jersey, 1992.

Gibson, M., and Hughes, C., Systems Analysis and Design: A Comprehensive Methodology with Case, Boyd and Fraser, Danvers, Massachusetts, 1994.

Glass, R.L., "A Snapshot of Systems Development Practice", IEEE Software, Vol. 16 (3), 1999, 110-112.

Glass, R.L., "The Naturalness of Object Orientation: Beating a Dead Horse?" IEEE Software, Vol. 19 (3), 2002, 103-104.

Hantos, P., Inherent Risks in Object-Oriented Development, February 2005; The Aerospace Corporation, www.stc.hill.af.mil.

Hoffer, J., George, J., and Valacich, J., Modern Systems Analysis and Design, 5th ed., Pearson Prentice Hall, Upper Saddle River, New Jersey, 2008.

Hoover, H., and Olekshy, T., Object-Oriented Analysis and Design a Practitioner's Approach, May 2001, Avra Software Lab Inc., www.avrasoft.com.

Jadalowen, I., Structured Analysis and Structured Design (SSAD) Summary, 2002; Software Engineering Research Network, pages.cpsc.ucalgary.ca.

Larman, C., Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development, 3rd ed., Prentice Hall, Upper Saddle River, New Jersey, 2004.

Norman, R., Object-Oriented Systems Analysis and Design, Prentice Hall, Upper Saddle River, New Jersey, 1996.

Phelan, P., In What Way is Object-Oriented Methodology Better than Structured Methodology, November 2002; Expert Knowledgebase, www.techtarget.com.

Podeswa, H., B. O. O. M. Business Object-Oriented Modeling for Business Analysts, Course Technology, Boston, MA, 2005.

Rob, M., "Issues of Structured vs. Object-Oriented Methodology of Systems Analysis and Design", Issues in Information Systems, Volume V (1), 2004, 275-280.

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., and Lorensen, W., Object-Oriented Modeling and Design, Prentice-Hall, Englewood Cliffs, New Jersey, 1991.

Satzinger, J., Jackson, R., and Burd, S., Object-Oriented Analysis and Design with the Unified Process, Course Technology, Boston, Massachusetts, 2007.

Senn, J., Analysis and Design of Information Systems, McGraw-Hill, New York, New York, 1989.

Shah, V., Sivitanides, M., Martin, R., "Pitfalls of Object-Oriented Development", Business Quest A Journal of Applied Topics in Business and Economics, November 1997, www.westga.edu.

Shlaer, S., and Mellor, S., Object-Oriented Systems Analysis: Modeling the World in Data. Yourdon Press, Englewood Cliffs, New Jersey, 1988.

Sircar, S., Nerur, S., and Mahapatra, R., "Revolution or Evolution? A Comparison of Object-Oriented and Structured Systems Development Methods ", MIS Quarterly, Vol. 25 (4), 2001, 457-471.

Sommerville, I., Software Engineering, 6th ed. International Computer Science Series, Addison Wesley, Reading, MA, 2000.

Wu, S. and Wu, M., Systems Analysis and Design, Course Technology, Cambridge Massachusetts, 1994.