

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/344693016>

Modul Praktikum Rekayasa Perangkat Lunak

Book · July 2020

CITATIONS

10

READS

3,456

1 author:



[Muhammad Wali](#)

AMIK Indonesia

17 PUBLICATIONS 84 CITATIONS

SEE PROFILE

MODUL PRAKTIKUM

REKAYASA PERANGKAT LUNAK

digunakan sebagai acuan pembelajaran praktikum mata kuliah rekayasa dan perancangan perangkat lunak

MUHAMMAD WALI



Modul Praktikum Rekayasa Perangkat Lunak

Penulis

Muhammad Wali

Penyunting

Ananda Dwi Rahma

Penata Letak

Rosalita

Pendesain Sampul

Hanung Norenza Putra

Ellunar Publisher

Email: ellunar.publisher@gmail.com

Website: www.ellunarpublisher.com

Bandung; Ellunar, 2020

82 hlm., 14,8 x 21 cm

ISBN: 978-623-204-530-9

Cetakan pertama, Juli 2020

Undang-Undang Republik Indonesia Nomor 28 Tahun 2014 Tentang Hak Cipta

Lingkup Hak Cipta Pasal 1

Hak Cipta adalah hak eksklusif pencipta yang timbul secara otomatis berdasarkan prinsip deklaratif setelah suatu ciptaan diwujudkan dalam bentuk nyata tanpa mengurangi pembatasan sesuai dengan ketentuan peraturan perundang-undangan.

Ketentuan Pidana Pasal 113

(1) Setiap Orang yang dengan tanpa hak melakukan pelanggaran hak ekonomi sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf i untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 1 (satu) tahun dan/atau pidana denda paling banyak Rp100.000.000 (seratus juta rupiah).

(2) Setiap Orang yang dengan tanpa hak dan/atau tanpa izin Pencipta atau pemegang Hak Cipta melakukan pelanggaran hak ekonomi Pencipta sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf c, huruf d, huruf f, dan/atau huruf h untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 3 (tiga) tahun dan/atau pidana denda paling banyak Rp500.000.000,00 (lima ratus juta rupiah).

(3) Setiap Orang yang dengan tanpa hak dan/atau tanpa izin Pencipta atau pemegang Hak Cipta melakukan pelanggaran hak ekonomi Pencipta sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf a, huruf b, huruf e, dan/atau huruf g untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 4 (empat) tahun dan/atau pidana denda paling banyak Rp1.000.000.000,00 (satu miliar rupiah).

(4) Setiap Orang yang memenuhi unsur sebagaimana dimaksud pada ayat (3) yang dilakukan dalam bentuk pembajakan, dipidana dengan pidana penjara paling lama 10 (sepuluh) tahun dan/atau pidana denda paling banyak Rp4.000.000.000,00 (empat miliar rupiah).

Prakata

Puji syukur penulis panjatkan ke hadapan Tuhan Yang Maha Esa, karena atas rahmat dan restu-Nya penulis dapat menyelesaikan buku praktikum rekayasa perangkat lunak ini. Belum adanya buku praktikum sebagai acuan untuk mata kuliah Rekayasa Perangkat Lunak dan Perancangan Sistem Informasi yang mendorong penulis untuk menyelesaikan buku ini.

Buku Praktikum Rekayasa Perangkat Lunak tersusun atas beberapa bagian yakni dimulai dari perangkat lunak dan rekayasanya, Microsoft Visio dan StarUML, objek data, atribut dan relasi, analisis sistem, diagram alir data i, bagan alir (*flowchart*), bagan alir (*flowchart*) lanjutan, dan *use case diagram*. Buku praktikum ini ditujukan kepada khalayak yang ingin memahami mengenai rekayasa perangkat lunak, terutama bagi mahasiswa Program Studi Teknik Informatika, Universitas Serambi Mekkah (USM) dan Program Studi Manajemen Informatika, AMIK Indonesia. Buku praktikum ini diharapkan dapat menjadi acuan untuk membantu mahasiswa dalam perkuliahan.

Buku praktikum ini dapat diselesaikan dengan baik berkat dukungan dari berbagai pihak. Penulis dalam kesempatan ini ingin menyampaikan ucapan terima kasih kepada semua pihak dan terutama kepada Dekan Fakultas Teknik Universitas Serambi Mekkan (USM) dan Direktur AMIK Indonesia beserta dosen-dosen. Semoga buku praktikum ini dapat bermanfaat bagi semua pihak

yang membutuhkan. Penulis menyadari, dalam penulisan buku praktikum ini masih banyak terdapat kekurangan. Penulis sangat mengharapkan saran yang bersifat membangun demi kesempurnaan buku praktikum ini.

Daftar Isi

Prakata.....	iii
Daftar Isi.....	v
BAB I PERANGKAT LUNAK DAN REKAYASANYA.....	1
1.1 Perangkat Lunak	2
1.2 Rekayasa Perangkat Lunak.....	13
1.3 Software Engineering	15
1.4 Software Engineering dan Computer Science	15
1.5 Software Engineering dan System Engineering.....	16
BAB II MICROSOFT VISIO DAN STARUML.....	18
2.1 Microsoft Visio.....	19
2.2 StarUML	24
BAB III OBJEK DATA, ATRIBUT, DAN RELASI.....	35
3.1 Objek Data, Atribut, dan Relasi.....	36
3.2 Skenario	36
BAB IV ANALISIS SISTEM	40
4.1 Analisis Sistem.....	41
4.2 Analisis Kebutuhan.....	41
BAB V DIAGRAM ALIR DATA I.....	47
5.1 DAD dan Komponen Pembentuk DAD.....	48
5.2 Konsep Arus Data	51
5.3 Diagram-Diagram di Dalam Dad.....	55
BAB VI BAGAN ALIR (FLOWCHART)	69

6.1 Flowchart	70
6.2 Macam-Macam Flowchart	70
BAB VII BAGAN ALIR (FLOWCHART) II.....	76
7.1 Pendahuluan	77
BAB VIII USE CASE DIAGRAM	79
8.1 Unified Modelling Language (UML)	80
8.2 Use Case Diagram.....	82
Daftar Pustaka.....	85
Biodata Penulis	87

BAB I

PERANGKAT LUNAK DAN REKAYASANYA



Overview

Menjelaskan perangkat lunak, Rekayasa Perangkat Lunak, dan mengenal perangkat lunak serta perbedaan antara Rekayasa Perangkat Lunak dengan ilmu *science* dan hubungan *software engineering* dan *system engineering*.



Tujuan

-
-
1. Mahasiswa mengetahui apa yang dimaksud Perangkat Lunak.
 2. Mahasiswa mengetahui apa yang dimaksud Rekayasa Perangkat Lunak.
 3. Mahasiswa mengetahui apa yang dilakukan Rekayasa Perangkat Lunak.
 4. Mahasiswa mengetahui bagaimana Perangkat Lunak direkayasa (proses).
 5. Mahasiswa mengetahui perbedaan Software Engineering vs Computer Science.

1.1 Perangkat Lunak

Dalam dunia teknologi informasi, kita sering mendengar kata *software*. Banyak pendapat mengatakan bahwa tanpa *software*, maka suatu komputer tidak dapat digunakan atau dioperasikan. Untuk mengetahui definisi atau pengertian dari *software* silakan simak sedikit penjelasannya di bawah ini.

1.1.1 Pengertian Software

Nama lain dari *software* adalah perangkat lunak. Karena disebut juga sebagai perangkat lunak, maka sifatnya pun berbeda dengan *hardware* atau perangkat keras. Jika perangkat keras adalah komponen yang nyata, yang dapat dilihat dan disentuh oleh secara langsung manusia, maka *software* atau perangkat lunak tidak dapat disentuh dan dilihat secara fisik. *Software* memang tidak tampak secara fisik dan tidak berwujud benda, tetapi bisa untuk dioperasikan.

Pengertian *software* komputer adalah sekumpulan data elektronik yang disimpan dan diatur oleh komputer. Data elektronik yang disimpan oleh komputer itu dapat berupa program atau instruksi yang akan menjalankan suatu perintah. Melalui *software* atau perangkat lunak inilah suatu komputer dapat menjalankan suatu perintah.

1.1.2 Jenis-Jenis Software atau Perangkat Lunak

Software atau perangkat lunak komputer berdasarkan distribusinya dibedakan menjadi beberapa macam, yaitu *software* berbayar, *software* gratis atau *free* (*freeware*, *free software*, *shareware*, dan *adware*).

a. Software Berbayar

Software berbayar merupakan perangkat lunak yang didistribusikan untuk tujuan komersial. Setiap pengguna yang ingin menggunakan atau mendapatkan *software* tersebut dengan cara membeli atau membayar pada pihak yang mendistribusikannya. Pengguna yang menggunakan *software* berbayar umumnya tidak diijinkan untuk menyebarkan *software* tersebut secara bebas tanpa izin dari penerbitnya. Contoh *software* berbayar ini misalnya adalah sistem Microsoft Windows, Microsoft Office, Adobe Photoshop, dan lain-lain.

b. Freeware

Freeware atau perangkat lunak gratis adalah perangkat lunak komputer berhak cipta yang gratis digunakan tanpa batasan waktu, berbeda dari *shareware* yang mewajibkan penggunaanya membayar (misalnya setelah jangka waktu

percobaan tertentu atau untuk memperoleh fungsi tambahan). Para pengembang perangkat gratis sering kali membuat perangkat gratis *freeware* “untuk disumbangkan kepada komunitas”, tetapi juga tetap ingin mempertahankan hak mereka sebagai pengembang dan memiliki kontrol terhadap pengembangan selanjutnya. *Freeware* juga didefinisikan sebagai program apa pun yang didistribusikan gratis, tanpa biaya tambahan. Sebuah contoh utama adalah Suite Browser, Mail Client, dan Mozilla News, juga didistribusikan di bawah GPL (*free software*).

c. Free Software

Free software lebih mengarah kepada bebas penggunaan, tetapi tidak harus gratis. Pada kenyataannya, namanya adalah karena bebas untuk mencoba perangkat lunak sumber terbuka (*open source*) dan di sanalah letak inti dari kebebasan. Program-program di bawah GPL, sekali diperoleh dapat digunakan, disalin, dimodifikasi, dan didistribusikan secara bebas. Jadi *free software* tidak mengarah kepada gratis pembelian, tetapi pada penggunaan dan

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

distribusi. Begitu keluar dari lisensi, kita dapat menemukan berbagai cara untuk mendistribusikan perangkat lunak, termasuk *freeware*, *shareware*, atau *adware*. Klasifikasi ini memengaruhi cara di mana program dipasarkan dan independen dari lisensi perangkat lunak mana mereka berasal.

Perbedaan yang nyata antara *free software* dan *freeware*. Konflik muncul dalam arti kata *free* dalam Bahasa Inggris, yang berarti keduanya bebas dan gratis. Oleh karena itu, seperti yang disebutkan sebelumnya, *free software* tidak perlu bebas, sama seperti *freeware* tidak harus gratis.

d. Shareware

Shareware juga bebas, tetapi lebih dibatasi untuk waktu tertentu. *Shareware* adalah program terbatas didistribusikan baik sebagai demonstrasi atau versi evaluasi dengan fitur atau fungsi yang terbatas atau dengan menggunakan batas waktu yang ditetapkan (misalnya, 30 hari). Dengan demikian, memberikan pengguna kesempatan untuk menguji produk sebelum membeli dan kemudian membeli versi lengkap dari program. Sebuah contoh yang sangat jelas dari tipe ini adalah perangkat lunak antivirus. Perusahaan-

perusahaan ini biasanya memudahkan pelepasan produk evaluasi yang hanya berlaku untuk jumlah hari tertentu. Setelah melewati maksimum, program akan berhenti bekerja dan Anda perlu membeli produk jika Anda ingin tetap menggunakannya.

Kita juga dapat menemukan perangkat lunak bebas sepenuhnya, tetapi termasuk dalam program periklanan. Distribusi jenis ini disebut *adware*. Sebuah contoh yang jelas adalah program Messenger dari Microsoft yang memungkinkan penggunaan perangkat lunak bebas dalam pertukaran untuk masuk dengan cara iklan *banner* atau *pop-up*.

e. Firmware

Firmware adalah istilah yang mengacu kepada rutin-rutin perangkat lunak yang disimpan di dalam Memori Hanya Baca (MHB). Tidak seperti Memori Akses Acak, MHB tidak akan dapat berubah meski tidak dialiri listrik. Rutin-rutin yang mampu menyalakan komputer (*startup*) serta instruksi input/output dasar (semacam BIOS atau sistem operasi *embedded*) disimpan di dalam perangkat tegar. Modifikasi

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

memang dapat dilakukan, tetapi hal tersebut tergantung dari jenis ROM apa yang digunakan. Perangkat tegar yang disimpan dalam ROM tidak dapat diubah, tetapi perangkat tegar yang disimpan dalam ROM yang dapat diubah semacam EEPROM atau Flash ROM masih dapat diubah sesuka hati.

f. Malware

Perangkat perusak (Bahasa Inggris: *malware*, berasal dari lakuran kata *malicious* dan *software*) adalah perangkat lunak yang diciptakan untuk menyusup atau merusak sistem komputer, peladen, atau jejaring komputer tanpa izin termaklum (*informed consent*) dari pemilik. Istilah ini adalah istilah umum yang dipakai oleh pakar komputer untuk mengartikan berbagai macam perangkat lunak atau kode perangkat lunak yang mengganggu atau mengusik. Istilah "*virus computer*" kadang-kadang dipakai sebagai frasa pemikat (*catch phrase*) untuk mencakup semua jenis perangkat perusak, termasuk virus murni (*true virus*).

Perangkat lunak dianggap sebagai perangkat perusak berdasarkan maksud yang terlihat dari pencipta dan bukan berdasarkan ciri-

ciri tertentu. Perangkat perusak mencakup virus komputer, cacing komputer, kuda Troya (*Trojan horse*), kebanyakan kit-akar (*rootkit*), perangkat pengintai (*spyware*), perangkat iklan (*adware*) yang tak jujur, perangkat jahat (*crimeware*), dan perangkat lunak lainnya yang berniat jahat dan tidak diinginkan. Menurut undang-undang, perangkat perusak kadang-kadang dikenali sebagai “pencemar komputer”. Hal ini tertera dalam kode undang-undang di beberapa negara bagian Amerika Serikat, termasuk California dan West Virginia.

Perangkat perusak tidak sama dengan perangkat lunak cacat (*defective software*), yaitu perangkat lunak yang mempunyai tujuan sah, tetapi berisi kutu (*bug*) yang berbahaya.

Hasil penelitian awal dari Symantec yang diterbitkan pada tahun 2008 menyatakan bahwa, "Kelajuan peluncuran kode yang berbahaya dan perangkat lunak lainnya yang tidak diinginkan, mungkin akan melebihi aplikasi perangkat lunak yang sah." Menurut F-Secure, "Jumlah perangkat perusak yang dibuat pada tahun 2007 sama dengan pembuatan dalam 20 tahun sekaligus.

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

Jalur pembobolan perangkat perusak yang paling umum digunakan oleh penjahat kepada pengguna adalah melalui internet, surel, dan Waring Wera Wanua (*world wide web*)."

Kelaziman perangkat perusak sebagai wahana bagi kejahatan internet terancang bersama dengan ketidakmampuan pelantar pemburu perangkat perusak biasa untuk melindungi sistem terhadap perangkat perusak yang terus menerus dibuat, mengakibatkan penerapan pola pikir baru bagi perniagaan yang berusaha di internet. Kesadaran bahwa pihak perniagaan tetap harus menjalankan usaha dengan sejumlah pelanggan internet yang memiliki komputer berjangkit. Hasilnya adalah penekanan lebih besar pada sistem kantor belakang (*back-office system*) yang dirancang untuk melacak kegiatan penipuan dalam komputer pelanggan yang berkaitan dengan perangkat perusak canggih.

g. Pengendali Perangkat Keras (*Device Driver*)

Pemacu peranti (Bahasa Inggris: *device driver*) adalah istilah teknologi informasi yang mengacu kepada komponen perangkat lunak yang mengizinkan sebuah sistem komputer untuk berkomunikasi dengan sebuah perangkat keras.

Sebagian besar perangkat keras tidak akan dapat berjalan atau sama sekali tidak dapat berjalan tanpa *driver* yang cocok yang terinstal di dalam sistem operasi. *Device driver*, umumnya akan dimuat ke dalam ruangan *kernel* (*kernel space*) sistem operasi selama proses *booting* dilakukan, atau secara sesuai permintaan ketika ada intervensi pengguna atau memasukkan sebuah perangkat *plug-and-play*. Beberapa sistem operasi juga menawarkan *device driver* yang berjalan di dalam ruangan pengguna (*userspace*) sistem operasi. Beberapa *driver* telah dimasukkan ke dalam sistem operasi secara *default* pada saat instalasi, tetapi banyak perangkat keras, khususnya yang baru, tidak dapat didukung oleh *driver-driver* bawaan sistem operasi. Adalah tugas pengguna yang harus menyuplai dan memasukkan *driver* ke dalam sistem operasi. *Driver* juga pada umumnya menyediakan layanan penanganan interupsi perangkat keras yang dibutuhkan oleh perangkat keras.

h. Perangkat Lunak Aplikasi (*Application Software*)

Perangkat lunak aplikasi (Bahasa Inggris: *application software*) adalah suatu subkelas

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas yang diinginkan pengguna. Biasanya dibandingkan dengan perangkat lunak sistem yang mengintegrasikan berbagai kemampuan komputer, tetapi tidak secara langsung menerapkan kemampuan tersebut untuk mengerjakan suatu tugas yang menguntungkan pengguna. Contoh utama perangkat lunak aplikasi adalah pengolah kata, lembar kerja, dan pemutar media.

Beberapa aplikasi yang digabung bersama menjadi suatu paket kadang disebut sebagai suatu paket atau *suite* aplikasi (*application suite*). Contohnya adalah Microsoft Office dan OpenOffice.org yang menggabungkan suatu aplikasi pengolah kata, lembar kerja, serta beberapa aplikasi lainnya. Aplikasi-aplikasi dalam suatu paket biasanya memiliki antarmuka pengguna yang memiliki kesamaan sehingga memudahkan pengguna untuk mempelajari dan menggunakan tiap aplikasi. Sering kali, mereka memiliki kemampuan untuk saling berinteraksi satu sama lain sehingga menguntungkan pengguna. Contohnya, suatu lembar kerja dapat

dibenamkan dalam suatu dokumen pengolah kata walaupun dibuat pada aplikasi lembar kerja yang terpisah.

i. Sistem Operasi (*Operating System*)

Sebuah sistem operasi memberikan segala layanan yang mengeksploitasi sumber daya yang dibutuhkan satu atau lebih proses kepada pengguna. Sumber daya tersebut berkaitan erat dengan sistem komputer. Hal ini dikarenakan sistem operasi mengatur komponen-komponen pendukung sistem komputer seperti memori, I/O modul ataupun I/O *device* dan komponen pembentuk lainnya sehingga terselenggaranya eksekusi proses dan menyembunyikan kerumitan pengaturan perangkat keras dari pengguna dan pembuat aplikasi. Hal tersebut menyebabkan perlunya memahami bagaimana sistem komputer bekerja untuk mengetahui bagaimana sistem operasi melaksanakan tugasnya.

Sistem operasi (Bahasa Inggris: *operating system* atau OS) adalah seperangkat program yang mengelola sumber daya perangkat keras komputer atau *hardware* dan menyediakan layanan umum untuk aplikasi perangkat lunak.

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

Sistem operasi adalah jenis yang paling penting dari perangkat lunak sistem dalam sistem komputer. Tanpa sistem operasi, pengguna tidak dapat menjalankan program aplikasi pada komputer mereka, kecuali program aplikasi *booting*.

Sistem operasi mempunyai penjadwalan yang sistematis mencakup perhitungan penggunaan memori, pemrosesan data, penyimpanan data, dan sumber daya lainnya.

Untuk fungsi-fungsi perangkat keras seperti sebagai masukan, keluaran, dan alokasi memori, sistem operasi bertindak sebagai perantara antara program aplikasi dan perangkat keras komputer. Meskipun, kode aplikasi biasanya dieksekusi langsung oleh perangkat keras dan sering kali akan menghubungi OS atau terputus oleh itu. Sistem operasi yang ditemukan pada hampir semua perangkat yang berisi komputer dari ponsel dan konsol permainan video untuk superkomputer dan server web.

1.2 Rekayasa Perangkat Lunak

Perangkat lunak kini sudah menjadi kekuatan yang menentukan. Perangkat lunak menjadi mesin yang

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

mengendalikan pengambilan keputusan di dalam dunia bisnis, berfungsi sebagai dasar dari semua bentuk pelayanan serta penelitian keilmuan modern. Saat ini perangkat lunak memiliki dua peran. Di satu sisi berfungsi sebagai sebuah produk dan di sisi lain sebagai kendaraan yang mengantarkan sebuah produk.

Sebenarnya, apa yang dimaksud dengan perangkat lunak? Perangkat lunak (*software*) adalah: (1) perintah (program komputer) yang bila dieksekusi memberikan fungsi dan unjuk kerja seperti yang diinginkan, (2) struktur data yang memungkinkan program memanipulasi informasi secara proporsional, dan (3) dokumen yang menggambarkan operasi dan penggunaan program.

Sedangkan yang dimaksud dengan Rekayasa Perangkat Lunak adalah suatu disiplin ilmu yang membahas semua aspek produksi perangkat lunak, mulai dari tahap awal *requirement capturing* (analisa kebutuhan pengguna), *specification* (menentukan spesifikasi dari kebutuhan pengguna), desain, *coding*, *testing*, sampai pemeliharaan sistem setelah digunakan.

Jadi yang perlu digarisbawahi, Rekayasa Perangkat Lunak merupakan serangkaian proses yang amat panjang untuk membuat atau menciptakan suatu perangkat lunak, bukan merupakan cabang ilmu *computer science* yang mempelajari tentang *technical coding*.

1.3 Software Engineering

Software engineering adalah teknologi yang harus digunakan oleh setiap orang yang akan membangun *software* dengan melalui serangkaian proses menggunakan sekumpulan metode dan alat bantu atau *tools* (Pressman, 1997). Proses dari *software engineering*, yaitu:

- a. *Management process*, terdiri dari: *Project*
 - *Management Configuration Management*
 - *Quality Assurance Management*
- b. *Technical process*, digambarkan sebagai metode yang akan diterapkan dalam tahap tertentu dari pengembangan siklus hidup *software*.
 - *Analysis Methods Design*
 - *Methods Programming Methods*
 - *Testing Methods*
- c. Metode teknis yang mengarah ke paradigma.

1.4 Software Engineering dan Computer Science

Perbedaan *software engineering* dan *computer science*:

- a. *Computer science* fokus pada teori dan dasar-dasar, sedangkan *software engineering* fokus pada praktik, pembangunan, dan pengiriman penggunaan *software*.
- b. Teori *computer science* masih belum cukup untuk menetapkan sebagai sebuah tiang fondasi untuk *software engineering*.

1.5 Software Engineering dan System Engineering

Perbedaan *software engineering* dan *system engineering*:

- a. *System engineering* fokus pada semua aspek pembangunan sistem dasar komputer meliputi *hardware*, *software*, dan *process engineering*. *Software engineering* adalah bagian dari proses ini yang berfokus pada pembangunan prasarana perangkat lunak, kontrol, aplikasi, dan *database* pada sistem.
- b. *System engineers* terlibat dalam spesifikasi sistem, perancangan arsitektur, integrasi, dan penyebaran.



Rangkuman

Rekayasa Perangkat Lunak adalah suatu disiplin ilmu yang membahas semua aspek produksi perangkat lunak, mulai dari tahap awal *requirement capturing* (analisa kebutuhan pengguna), *specification* (menentukan spesifikasi dari kebutuhan pengguna), desain, *coding*, *testing*, sampai pemeliharaan sistem setelah digunakan.

Jadi yang perlu digarisbawahi, Rekayasa Perangkat Lunak merupakan serangkaian proses yang amat panjang untuk membuat atau menciptakan suatu perangkat lunak, bukan merupakan cabang ilmu *computer science* yang mempelajari tentang *technical coding*.



Soal Latihan

1. Jelaskan Rekayasa Perangkat Lunak?
2. Jelaskan tujuan dari Rekayasa Perangkat Lunak?
3. Programmer apakah termasuk dalam Rekayasa Perangkat Lunak?
Jelaskan!
4. Sebutkan perangkat lunak *middleware*?
5. Apa perbedaan *software engineer* dan *computer science*?

BAB II MICROSOFT VISIO DAN STARUML



Overview

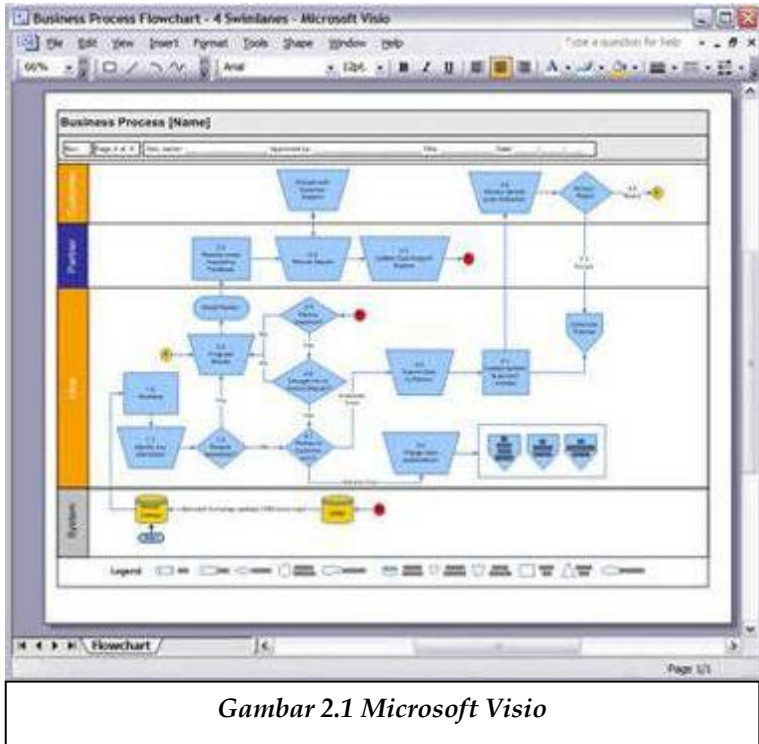
Mahasiswa mengetahui *tools* yang digunakan untuk perancangan perangkat lunak. *Software* atau *tools* yang akan digunakan pada praktikum RPL adalah Microsoft Visio dan StarUML.



Tujuan

1. Mahasiswa mengenal *tools* yang akan digunakan pada praktikum RPL.
2. Mahasiswa mengenal macam-macam *software* yang digunakan untuk perancangan sistem.

2.1 Microsoft Visio



Microsoft Visio (atau sering disebut Visio) adalah sebuah program aplikasi komputer yang sering digunakan untuk membuat diagram, diagram alir (*flowchart*), *brainstorm*, dan skema jaringan yang dirilis oleh Microsoft Corporation. Aplikasi ini menggunakan grafik vektor untuk membuat diagram-diagramnya.

Visio merupakan *software* yang mendukung untuk mendeskripsikan data dan fungsi/proses yang terlibat dalam

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

proses rekayasa suatu perangkat lunak. Adapun data dan fungsi yang dimodelkan dalam proses dalam rekayasa perangkat lunak adalah:

a. Pemodelan Data

1) ERD (Diagram Keterhubungan Antarobjek Data)

Entity Relationship Diagram merupakan diagram yang dapat digunakan untuk melakukan aktivitas pemodelan data dan menggambarkan hubungan antara objek data. Dalam ERD terdapat tiga komponen yang digunakan yaitu entitas, atribut, dan relasi. Contoh:

- Entitas (dosen)
- Atribut (NIP, nama dosen, alamat, nomor telepon, jabatan)

b. Kamus Data

Merupakan catatan yang digunakan untuk menyimpan deskripsi atau atribut dari semua objek data yang didefinisikan. Contoh: Dosen: {NIP, Nama Dosen, Alamat, No_Telpon, Jabatan}.

c. DFD

Mendeskripsikan seluruh fungsi yang terlibat dalam perangkat lunak. DFD atau diagram aliran data merupakan gambaran bagaimana data ditransformasikan pada sebuah sistem.

2.1.1 Sejarah Microsoft Visio

Microsoft Visio (atau sering disebut Visio) adalah sebuah program aplikasi komputer yang sering digunakan untuk membuat diagram, diagram alir (*flowchart*), *brainstorm*, dan skema jaringan yang dirilis oleh Microsoft Corporation. Aplikasi ini menggunakan grafik vektor untuk membuat diagram- diagramnya.

Visio aslinya bukanlah buatan Microsoft Corporation, melainkan buatan Visio Corporation, yang diakuisisi oleh Microsoft pada tahun 2000. Versi yang telah menggunakan nama Microsoft Visio adalah Visio 2002, Visio 2003, dan Visio 2007 yang merupakan versi terbaru. Visio 2007 Standard dan Professional menawarkan antarmuka pengguna yang sama. Akan tetapi, seri Professional menawarkan lebih banyak pilihan *template* untuk pembuatan diagram yang lebih lanjut dan juga penataan letak (*layout*). Selain itu, edisi Professional juga memudahkan pengguna untuk mengoneksikan diagram-diagram buatan mereka terhadap beberapa sumber data dan juga menampilkan informasi secara visual dengan menggunakan grafik.

2.1.2 Versi-versi Visio

- a. Visio 5.0
- b. Visio 2000
- c. Microsoft Visio 2002 (dikenal juga dengan sebutan

Visio XP)

- d. Microsoft Office Visio 2003
- e. Microsoft 2007
- f. Microsoft Office Visio 2010
- g. Microsoft Office Visio 2013
- h. Microsoft Office Visio 2015

2.1.3 Format berkas

- *.VSD
- *.VSS
- *.VST
- *.vdx
- *.vsx
- *.vtx

2.1.4 Bentuk-Bentuk Diagram pada Visio

Adapun bentuk-bentuk diagram pada Microsoft Visio antara lain:

a. Diagram Jaringan (*Network Diagram*)

Membuat susunan bentuk jaringan komputer atau lainnya sesuai kebutuhan hanya dengan menggabungkan beberapa bentuk kombinasi bentuk dari Visio yang sudah tersedia.

b. Flowchart Dasar (*Basic Flowchart*)

Membuat dokumen prosedur, menganalisis proses, menunjukkan alur kerja atau informasi, lagu, efisiensi

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

biaya, dan banyak lagi.

c. Rencana Denah (*Floor Plan*)

Membuat denah rencana peletakan pintu, jendela, dan alat-alat listrik secara visual untuk bangunan.

d. Bagan Struktur Organisasi (*Organization Chart*)

Membentuk bagan kelompok kerja dalam organisasi dan hubungan komunikasi antar departemen.

e. Diagram Basis Data (*Database Model Diagram*)

Membentuk model dari *database* secara visual dengan penggambaran bentuk skema sesuai aslinya.

f. Diagram Situs Jaringan (*Website Diagram*)

Menggambar bentuk susunan halaman *website* secara hierarki dan alur penggunaannya yang dapat dengan mudah diubah sesuai dengan kebutuhan web saat ini yang dinamis.

g. Diagram Blok (*Block Diagram*)

Melakukan *brainstorming*, rencana, dan berkomunikasi.

h. Peta Petunjuk (*Directional Maps*)

Peta petunjuk yang dapat menunjukkan arah dengan disertai petunjuk alam seperti pohon, bangunan, sungai, dan jalan raya sebagai petunjuknya.

i. Diagram Proses Mesin (*Proccess Enginerig Diagram*)

Menunjukkan bagaimana proses dari alat-alat dalam perindustrian serta alat-alat yang digunakan seperti

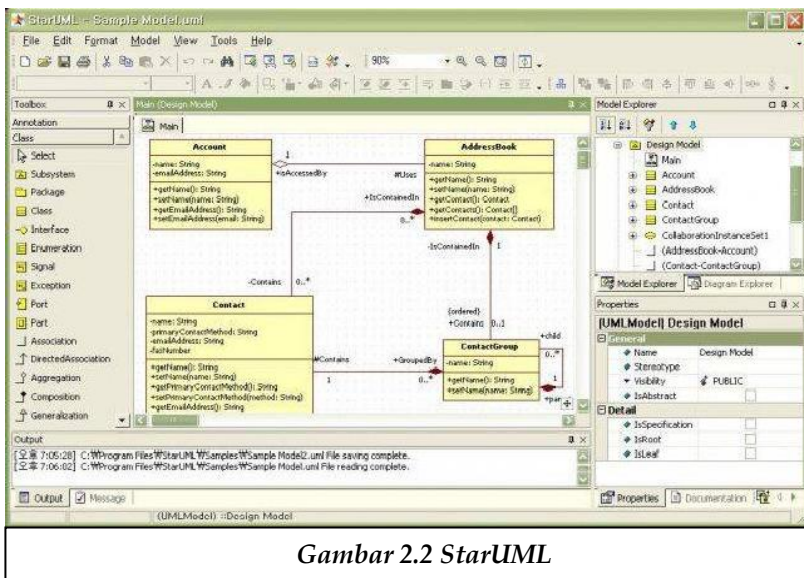
MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

mesin, pipa, dan penampungnya.

j. Diagram Software (*Software Diagram*)

Membantu mengembangkan tim desain perangkat lunak untuk mengatur tampilan *user interface*.

2.2 StarUML



Gambar 2.2 StarUML

StarUML, adalah salah satu *tool* pendesainan yang mendukung *Unified Modelling Language* (UML) yang bersifat *open source*. Hampir semua kegiatan dalam modul ini dilakukan menggunakan StarUML yang fungsinya bisa dibandingkan dengan Rational Rose, Together, atau UML *tool* lainnya.

Adapun *software* atau *tools* lainnya baik itu *tool* komersial

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

maupun *open source* antara lain:

- Rational Rose (www.rational.com)
- Together (www.togethersoft.com)
- Object Domain (www.objectdomain.com)
- Jvision (www.object-insight.com)
- Objectteering (www.objectteering.com)
- MagicDraw (www.nomagic.com/magicdrawuml)
- Visual Object Modeller (www.visualobject.com)

2.2.1 Pengertian UML

UML (*Unified Modeling Language*) adalah sebuah bahasa untuk menentukan, visualisasi, kontruksi, dan mendokumentasikan *artifact* (bagian dari informasi yang digunakan atau dihasilkan dalam suatu proses pembuatan perangkat lunak). *Artifact* dapat berupa model, deskripsi atau perangkat lunak dari sistem perangkat lunak, seperti pada pemodelan bisnis dan sistem nonperangkat lunak lainnya.

UML merupakan suatu kumpulan teknik terbaik yang telah terbukti sukses dalam memodelkan sistem yang besar dan kompleks. UML tidak hanya digunakan dalam proses pemodelan perangkat lunak, tetapi hampir dalam semua bidang yang membutuhkan pemodelan.

2.2.2 Bagian-Bagian UML

Bagian-bagian utama dari UML adalah *view*, *diagram*, *model element*, dan *general mechanism*.

a. View

View digunakan untuk melihat sistem yang dimodelkan dari beberapa aspek yang berbeda. *View* bukan melihat grafik, tetapi merupakan suatu abstraksi yang berisi sejumlah *diagram*. Beberapa jenis *view* dalam UML antara lain *use case view*, *logical view*, *component view*, *concurrency view*, dan *deployment view*.

b. Use Case View

Mendesripsikan fungsionalitas sistem yang seharusnya dilakukan sesuai yang diinginkan *external actors*. *Actor* yang berinteraksi dengan sistem yang dapat berupa *user* atau sistem lainnya. *View* ini digambarkan dalam *use case diagram* dan kadang-kadang dengan *activity diagrams*. *View* ini digunakan terutama untuk pelanggan, perancang (*designer*), pengembang (*developer*), dan penguji sistem (*tester*).

c. Logical View

Mendesripsikan bagaimana fungsionalitas dari sistem, struktur statis (*class*, *object*, dan *relationship*) dan kolaborasi dinamis yang terjadi

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

ketika *object* mengirim pesan ke *object* lain dalam suatu fungsi tertentu. *View* ini digambarkan dalam *class diagrams* untuk struktur statis dan dalam *state*, *sequence*, *collaboration*, dan *activity diagram* untuk model dinamisnya. *View* ini digunakan untuk perancang (*designer*) dan pengembang (*developer*).

d. Component View

Mendeskripsikan implementasi dan ketergantungan modul. Komponen yang merupakan tipe lainnya dari *code module* diperlihatkan dengan struktur dan ketergantungannya juga alokasi sumber daya komponen dan informasi administratif lainnya. *View* ini digambarkan dalam *component view* dan digunakan untuk pengembang (*developer*).

e. Concurrency View

Membagi sistem ke dalam proses dan prosesor. *View* ini digambarkan dalam diagram dinamis (*state*, *sequence*, *collaboration*, dan *activity diagrams*) dan diagram implementasi (*component* dan *deployment diagrams*) serta digunakan untuk pengembang (*developer*), pengintegrasi (*integrator*), dan penguji (*tester*).

f. Deployment View

Mendeskripsikan fisik dari sistem seperti komputer dan perangkat (*nodes*) dan bagaimana hubungannya dengan lainnya. *View* ini digambarkan

dalam *deployment diagrams* dan digunakan untuk pengembang (developer), pengintegrasi (*integrator*), dan penguji (tester).

i. Diagram

Diagram berbentuk grafik yang menunjukkan simbol elemen model yang disusun untuk mengilustrasikan bagian atau aspek tertentu dari sistem. Sebuah diagram merupakan bagian dari suatu *view* tertentu dan ketika digambarkan biasanya dialokasikan untuk *view* tertentu. Adapun jenis diagram antara lain:

- Use Case Diagram

Use case adalah abstraksi dari interaksi antara *system* dan *actor*. *Use case* bekerja dengan cara mendeskripsikan tipe interaksi antara *user* sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. *Use case* merupakan konstruksi untuk mendeskripsikan bagaimana sistem akan terlihat di mata *user*. Sedangkan *use case diagram* memfasilitasi komunikasi di antara analis dan pengguna serta antara analis dan klien.

- Class Diagram

Class adalah dekripsi kelompok objek-objek

dengan properti, perilaku (operasi) dan relasi yang sama. Sehingga dengan adanya *class diagram* dapat memberikan pandangan global atas sebuah sistem. Hal tersebut tercermin dari *class-class* yang ada dan relasinya satu dengan yang lainnya. Sebuah sistem biasanya mempunyai beberapa *class diagram*. *Class diagram* sangat membantu dalam visualisasi struktur kelas dari suatu sistem.

- Component Diagram

Component software merupakan bagian fisik dari sebuah sistem, karena menetap di komputer tidak berada di benak para analis. Komponen merupakan implementasi *software* dari sebuah atau lebih *class*. Komponen dapat berupa *source code*, komponent biner, atau *executable component*. Sebuah komponen berisi informasi tentang *logic class* atau *class* yang diimplementasikan sehingga membuat pemetaan dari *logical view* ke *component view*. Sehingga *component diagram* merepresentasikan dunia riil yaitu *component software* yang mengandung *component*, *interface*, dan *relationship*.

- Deployment Diagram

Menggambarkan tata letak sebuah sistem secara fisik, menampilkan bagian-bagian

software yang berjalan pada bagian-bagian *hardware*, menunjukkan hubungan komputer dengan perangkat (*nodes*) satu sama lain dan jenis hubungannya. Di dalam *nodes*, *executable component* dan *object* yang dialokasikan untuk memperlihatkan unit perangkat lunak yang dieksekusi oleh *node* tertentu dan ketergantungan komponen.

- State Diagram

Menggambarkan semua *state* (kondisi) yang dimiliki oleh suatu *object* dari suatu *class* dan keadaan yang menyebabkan *state* berubah. Kejadian dapat berupa *object* lain yang mengirim pesan. *State class* tidak digambarkan untuk semua *class*, hanya yang mempunyai sejumlah *state* yang terdefinisi dengan baik dan kondisi *class* berubah oleh *state* yang berbeda.

- Sequence Diagram

Sequence diagram digunakan untuk menggambarkan perilaku pada sebuah skenario. Kegunaannya untuk menunjukkan rangkaian pesan yang dikirim antara *object* juga interaksi antara *object*, sesuatu yang terjadi pada titik tertentu dalam eksekusi sistem.

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

- Collaboration Diagram

Menggambarkan kolaborasi dinamis seperti *sequence diagrams*. Dalam menunjukkan pertukaran pesan, *collaboration diagrams* menggambarkan *object* dan hubungannya (mengacu ke konteks). Jika penekannya pada waktu atau urutan gunakan *sequence diagrams*, tetapi jika penekanannya pada konteks gunakan *collaboration diagram*.

- Activity Diagram

Menggambarkan rangkaian aliran dari aktivitas, digunakan untuk mendeskripsikan aktivitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktivitas lainnya seperti *use case* atau interaksi.

2.2.3 Tujuan Penggunaan UML

- a. Memberikan bahasa pemodelan yang bebas dari berbagai bahasa pemrograman dan proses rekayasa.
- b. Menyatukan praktik-praktik terbaik yang terdapat dalam pemodelan.
- c. Memberikan model yang siap pakai, bahasa pemodelan visual yang ekspresif untuk mengembangkan dan saling menukar model dengan mudah serta dimengerti secara umum.

- d. UML bisa juga berfungsi sebagai sebuah cetak biru (*blue print*) karena sangat lengkap dan detail. Dengan cetak biru ini maka akan bisa diketahui informasi secara detail tentang *coding* program atau bahkan membaca program dan menginterpretasikan kembali ke dalam bentuk diagram (*reverse engineering*).



Rangkuman

Microsoft Visio adalah salah satu *software* yang berfungsi untuk membuat desain berupa *flowchart*, diagram, bagan, dan lain-lain. Dengan Visio, semua proses pekerjaan akan lebih mudah terutama bagi kalangan IT dan analis. Bahkan untuk orang awam sekalipun, Visio mudah digunakan.

StarUML adalah suatu *open source project* sebagai pengembang kecepatan, fleksibilitas, keleluasaan, fitur, dan kebebasan platform UML/MDA yang berbasis Win32. Tujuan project StarUML adalah untuk membentuk suatu alat permodelan perangkat lunak dan platform yang mampu bersaing/menggantikan alat UML yang komersial, seperti Rational Rose, Together, dan yang lainnya.

UML kebanyakan ditulis dengan bahasa pemrograman Delphi. Akan tetapi, bagaimanapun juga, StarUML ada multi-bahasa *project* dan tidak mengharuskan bahasa pemrograman yang spesifik. Banyak bahasa pemrograman yang dapat digunakan untuk membantu mengembangkan *open source project* StarUML ini, seperti C/C++, Java, Visual Basic, Delphi, JScript, VBScript, C#, VB.NET, dan lain-lain.

Data Flow Diagram (DFD) adalah suatu diagram yang menggunakan notasi- notasi untuk menggambarkan arus dari data sistem, yang penggunaannya sangat membantu untuk memahami sistem secara logika, terstruktur, dan jelas.

DFD merupakan alat bantu dalam menggambarkan atau menjelaskan sistem yang sedang berjalan logis. UML dengan DFD keduanya berkaitan, tetapi berbeda. UML merupakan alat yang digunakan untuk memodelkan suatu sistem/*software*, sedangkan DFD merupakan salah satu dari model yang terdapat di dalam StarUML.



Soal Latihan

1. Jelaskan fungsi dari Microsoft Visio!
2. Jelaskan fungsi dari StarUML!
3. Sebutkan perbedaan Microsoft Visio dan StarUML!
4. Jelaskan hubungan Rekayasa Perangkat Lunak dengan Microsoft Visio dan StarUML!
5. Microsoft Visio atau StarUML yang memiliki model DAD?
Jelaskan!

BAB III

OBJEK DATA, ATRIBUT, DAN RELASI



Overview

Mahasiswa akan mempelajari contoh jenis dan aplikasi perangkat lunak sehingga dalam praktiknya dapat membedakan antara objek data, atribut, dan hubungannya.



Tujuan

-
-
1. Mahasiswa mampu menjelaskan analisis system dalam rekayasa perangkat lunak.
 2. Mahasiswa mampu menjelaskan objek data, atribut, dan hubungan antarobjek.

3.1 OBJEK DATA, ATRIBUT, DAN RELASI

Model data terdiri dari tiga informasi yang saling tergantung: objek data, atribut yang menggambarkan objek data tersebut, dan hubungan yang menghubungkan objek data yang satu dengan yang lain.

- a. Objek data (*entity*) adalah representasi dari hampir semua informasi gabungan yang harus dipahami oleh perangkat lunak. Dengan informasi gabungan kita mengartikan sesuatu yang memiliki sejumlah sifat atau atribut yang berbeda.
- b. Atribut adalah karakteristik dari *entity* atau *relationship* yang menyediakan detail tentang *entity* atau *relationship* tersebut sehingga dapat dibedakan.
- c. Relasi adalah hubungan yang terjadi antara satu atau lebih objek data (*entity*).

3.2 SKENARIO

Tentukan objek, atribut, dan relasi pada gambar (3.1)



Gambar 3.1 Pemeriksaan Pasien

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

- Langkah 1. Tentukan objek data dari gambar (3.1).
Objek : Dokter dan pasien
- Langkah 2. Tentukan atribut yang dimiliki.
Dokter : NIP, nama, alamat, spesialisasi
Pasien : NoRM, nama, alamat, jenis_kelamin, tgl_lahir, gol_darah
- Langkah 3. Tentukan relasi dari kedua objek tersebut.
Relasi : Dokter memeriksa pasien (memeriksa merupakan nama relasi)



Rangkuman

Microsoft Visio adalah salah satu *software* yang berfungsi untuk membuat desain berupa *flowchart*, diagram, bagan, dan lain-lain. Dengan Visio, semua proses pekerjaan akan lebih mudah terutama bagi kalangan IT dan analis. Bahkan untuk orang awam sekalipun, Visio mudah digunakan.



Soal Latihan

1. Tuliskan objek, atribut, dan relasi dari gambar 3.2.



Gambar 3.2 Bus Sekolah

2. Seorang pengendara sepeda motor melakukan pelanggaran di

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

jalan raya karena melewati garis marka. Polisi menghentikan laju pengendara dan mengenakan sanksi sesuai dengan undang-undang. Tentukan objek, atribut, dan relasi dari deskripsi tersebut.

BAB IV

ANALISIS SISTEM



Overview

Mahasiswa akan mempelajari tahapan analisis sistem dalam Rekayasa Perangkat Lunak dengan benar serta menjelaskan tujuan analisis sistem dalam Rekayasa Perangkat Lunak. Mahasiswa akan menganalisis kebutuhan sistem dalam Rekayasa Perangkat Lunak dan dapat menyebutkan kebutuhan fungsional dan nonfungsional dalam Rekayasa Perangkat Lunak.



Tujuan

-
-
1. Mahasiswa mampu menjelaskan pengertian analisis sistem.
 2. Mahasiswa mampu menjelaskan tahapan analisis sistem.
 3. Mahasiswa mampu menjelaskan analisis kebutuhan sistem dalam Rekayasa Perangkat Lunak.

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

4.1 ANALISIS SISTEM

Analisis sistem adalah kegiatan untuk mendefinisikan kebutuhan terkait sistem yang akan dikembangkan. Orang yang bertugas menganalisis sistem disebut sebagai *system analyst*.

4.1.1 Tahapan Analisis Sistem

Dalam melakukan analisis sistem tahapan yang dilalui adalah:

- a. **IDENTIFY** → mengidentifikasi masalah
- b. **UNDERSTAND** → memahami kerja sistem yang ada
- c. **ANALYZE** → menganalisis system
- d. **REPORT** → membuat laporan hasil

4.1.2 Tujuan dari Analisis Sistem Antara Lain

- a. Menjabarkan kebutuhan pemakai.
- b. Meletakkan dasar-dasar untuk proses perancangan perangkat lunak.
- c. Mendefinisikan semua kebutuhan pemakai sesuai dengan lingkup kedua belah pihak.

4.2 ANALISIS KEBUTUHAN

4.1.3 Analisis Kebutuhan Fungsional (Functional Requirement)

Merupakan kegiatan dalam mendeskripsikan

fungsionalitas atau layanan yang diharapkan akan diberikan oleh sistem. Persyaratan fungsional untuk system perangkat lunak bisa dinyatakan dalam sejumlah cara. Persyaratan tersebut bergantung pada jenis perangkat lunak, *user* yang menggunakan, dan jenis sistem yang akan dikembangkan.

Contoh:

Terdapat sejumlah persyaratan fungsional untuk sistem perpustakaan (Kotonya dan Sommerville, 1998) bagi mahasiswa dan dosen untuk memesan buku dan dokumen dari perpustakaan lain:

- *User* dapat mencari semua atau satu set awal *database* atau memilih subset darinya.
- Sistem akan menyediakan *viewer* yang sesuai bagi *user* untuk membaca dokumen pada penyimpanan (*store*) dokumen.

Pada prinsipnya spesifikasi persyaratan fungsional untuk sebuah sistem harus lengkap dan konsisten.

4.1.4 Analisis Kebutuhan Nonfungsional (Non Functional Requirement)

Merupakan persyaratan yang tidak langsung berhubungan dengan fungsi spesifik yang disediakan

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

oleh sistem. Persyaratan ini mungkin berhubungan dengan properti sistem yang muncul belakangan seperti keandalan, waktu tanggap, dan penempatan pada media penyimpanan. Alternatifnya, persyaratan ini dapat mendefinisikan batasan pada sistem seperti kemampuan *pirnti* I/O dan representasi data yang dipakai pada *interface* system.

Kebutuhan nonfungsional ini meliputi kebutuhan:

a. *Development Requirement*

Tools yang digunakan (*hardware* dan *software*) untuk pengembangan sistem. Contoh: Eclipse, NetBeans, StarUML, dsb.

b. *Deployment Requirement*

Terkait dengan lingkungan di mana sistem akan digunakan. Contoh: sistem harus mampu berjalan dengan spesifikasi RAM 4GB, OS Ubuntu, dsb.

d. *Performance Requirement*

Terkait dengan ukuran kualitas maupun kuantitas khususnya terkait dengan kecepatan, skalabilitas, dan kapasitas. Contoh: sistem harus mampu diakses oleh 100 orang dalam waktu bersamaan.

a. *Dokumentation Requirement*

Terkait dengan dokumen apa saja yang akan

disertakan pada produk akhir. Contoh: dokumen teknis (dokumen perencanaan proyek, analisis, desain, pengujian), *user manual*, dan dokumen pelatihan.

b. *Support Requirement*

Kebutuhan yang terkait dengan dukungan yang diberikan setelah sistem informasi digunakan. Contoh: perlu adanya pelatihan bagi calon pengguna.

Skenario

Minimarket milik Pak Budi menjual mulai peralatan rumah tangga, alat tulis, dan barang kelontong untuk kebutuhan sehari-hari. Pak Budi ingin membuat sistem yang bisa digunakan untuk transaksi jual beli. Sistem tersebut akan digunakan oleh Pak Budi sebagai direktur dan karyawannya sebagai kasir. Jika Anda diminta membangun sistem tersebut, analisislah kebutuhan sistemnya terlebih dahulu.

- Langkah 1. Identifikasi objek (entitas) lengkap dengan atribut yang terlibat di dalam sistem tersebut.

Objek : direktur, barang, kasir

- Langkah 2. Tuliskan jawaban Anda ke dalam model kamus data. Kamus data *minimarket*:

Direktur : {nama, alamat, no telp}

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

Barang : {kode_barang, nama_barang, jenis, stok}

Kasir : {kode_kasir, nama, alamat, no_telp}

- Langkah 3. Tuliskan kebutuhan fungsional sistem.

Kasir dapat menggunakan sistem untuk transaksi penjualan. Sistem dapat digunakan untuk menyimpan data barang baru. Direktur dapat melihat laporan data barang dan data penjualan.

- Langkah 4. Tuliskan kebutuhan nonfungsional. Sistem-sistem ini berbasis desktop.

Sistem dibangun dengan menggunakan bahasa pemrograman BASIC. Sistem dapat berjalan pada komputer dengan spesifikasi minimal RAM 1GB.



Rangkuman

Microsoft Visio adalah salah satu *software* yang berfungsi untuk membuat desain berupa *flowchart*, diagram, bagan, dan lain-lain. Dengan Visio, semua proses pekerjaan akan lebih mudah terutama bagi kalangan IT dan analis. Bahkan untuk orang awam sekalipun, Visio mudah digunakan.



Soal Latihan

-
-
1. Tentukan objek data dan atribut yang bisa ditemukan dalam apotek. Tuliskan hasil analisis kebutuhan fungsional dan nonfungsional untuk sistem informasi apotek.
 2. Analisislah kembali dengan mengembangkan ide-ide baru pada sistem informasi akademik (SIA) di kampus Anda. Buatlah kebutuhan fungsional dan nonfungsionalnya. Tuliskan teknologi baru yang Anda gunakan, jika ada.

BAB V

DIAGRAM ALIR DATA I



Overview

Mahasiswa dapat menjelaskan pengertian DAD dan kegunaannya, serta mahasiswa dapat memahami komponen-komponen yang terlibat dalam pembuatan DAD. Mahasiswa dapat menciptakan DAD dalam perancangan sistem serta mempelajari menggambar DAD menggunakan Microsoft Visio.



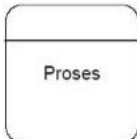
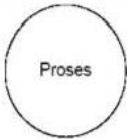
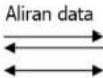
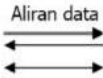
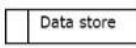



Tujuan

-
-
1. Mengenalkan DAD (Diagram Alir Data) sebagai *tool* perancangan sistem.
 2. Memahami tahapan-tahapan levelisasi DAD.
 3. Menggunakan Microsoft Visio untuk menggambar DAD.

5.1 DAD dan Komponen Pembentuk DAD

DAD (Diagram Aliran Data) atau yang juga dikenal dengan sebutan DFD (*Data Flow Diagram*) merupakan alat perancangan sistem yang berorientasi pada alur data dengan konsep dekomposisi dapat digunakan untuk penggambaran analisis maupun rancangan sistem yang mudah dikomunikasikan oleh profesional sistem kepada pemakai maupun pembuat program. Pada umumnya, DAD digunakan untuk merancang sistem yang menggunakan *data store* dalam mengelola informasi dalam sistem. Komponen DAD, menurut Yourdan dan DeMarco, adalah sebagai berikut.

Gane/Sarson	Yourdon/De Marco	Keterangan
		Entitas eksternal, dapat berupa orang/unit terkait yang berinteraksi dengan sistem tetapi diluar sistem
		Orang, unit yang mempergunakan atau melakukan transformasi data. Komponen fisik tidak diidentifikasi.
		Aliran data dengan arah khusus dari sumber ke tujuan
		Penyimpanan data atau tempat data direfer oleh proses.

Tabel 5.1 Komponen DAD

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

Keterangan:

- a. Entitas Luar adalah kesatuan di luar sistem yang akan memberikan input atau menerima *output* dari sistem, dapat berupa orang, organisasi, sumber informasi lain, atau penerima akhir dari suatu laporan.
- b. Proses adalah transformasi input menjadi *output* merupakan kegiatan atau pekerjaan yang dilakukan oleh orang atau mesin komputer, di mana aliran data masuk ditransformasikan ke aliran data keluar. Penamaannya sesuai dengan proses yang sedang dilakukan.

Ada beberapa hal yang perlu diperhatikan tentang proses:

1. Proses harus memiliki input dan *output*.
2. Proses dapat dihubungkan dengan komponen entitas luar, *data store* atau proses melalui alur data.
3. Sistem/bagian/divisi/departemen yang sedang dianalisis oleh profesional sistem digambarkan dengan komponen proses.
4. Penomoran proses dapat dilihat pada tabel 5.2.

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

Nama Level	Nama Diagram	Nomor Proses
0	Konteks	0
1	Diagram level 1	1.0, 2.0, 3.0
2	Diagram rinci 1.0	1.1, 1.2, 1.3
2	Diagram rinci 2.0	2.1, 2.2, 2.3
2	Diagram rinci 3.0	3.1, 3.2, 3.3
3	Diagram rinci 1.1	1.1.1, 1.1.2, 1.1.3
3	Diagram rinci 1.2	1.2.1, 1.2.2, 1.2.3
3	Diagram rinci 1.3	1.3.1, 1.3.2, 1.3.2
dst		

Tabel 5.2. Penomoran Proses

c. Aliran data/Arus data digunakan untuk menjelaskan perpindahan data atau paket data dari satu bagian ke bagian lain. Aliran data dapat berbentuk sebagai berikut:

- Formulir atau dokumen yang digunakan perusahaan Laporan tercetak yang dihasilkan sistem
- *Output* di layar komputer Masukan untuk komputer Komunikasi ucapan
- Surat atau memo
- Data yang dibaca atau direkam di *file* Suatu isian yang dicatat pada buku agenda
- Transmisi data dari suatu komputer ke komputer lain Catatan: aliran data tidak

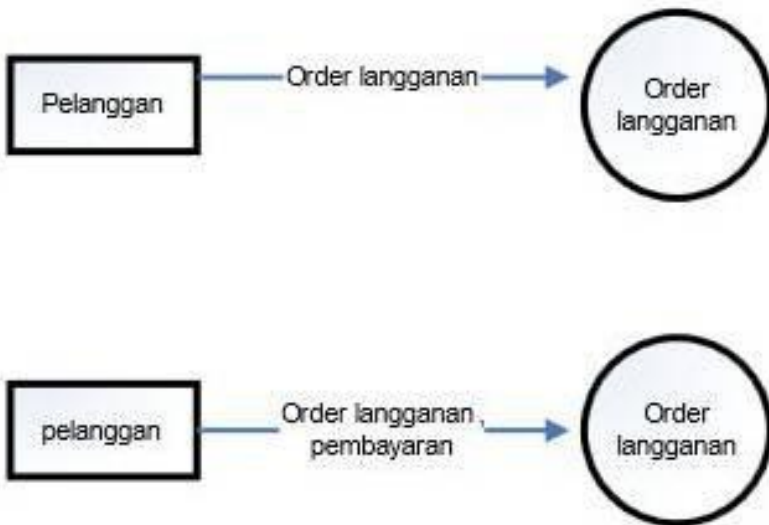
MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

dalam bentuk kalimat.

5.2 Konsep Arus Data

5.1.1 Packet of Data (Paket Data)

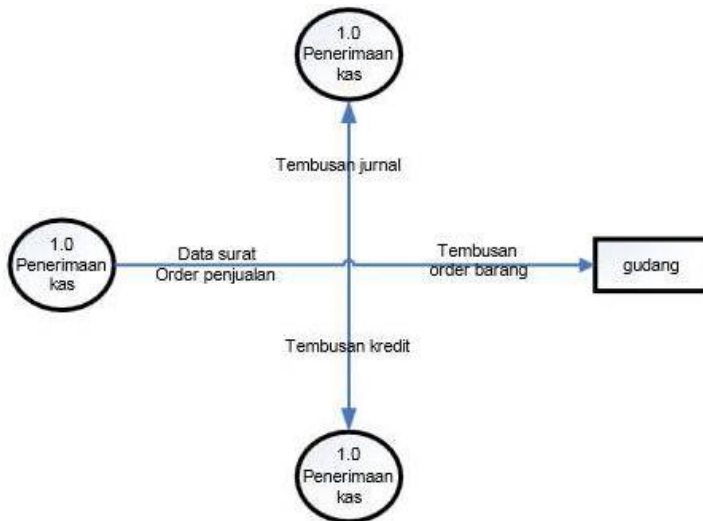
Bila dua data mengalir dari suatu sumber yang sama menuju ke tujuan yang sama, maka harus dianggap sebagai suatu arus data yang tunggal.



Gambar 5.1 Paket Data

5.1.2 Diverging Data Flow (Arus Data Menyebar)

Arus data yang data yang menyebar menunjukkan sejumlah tembusan dari arus data yang sama dari sumber sama ke tujuan berbeda.

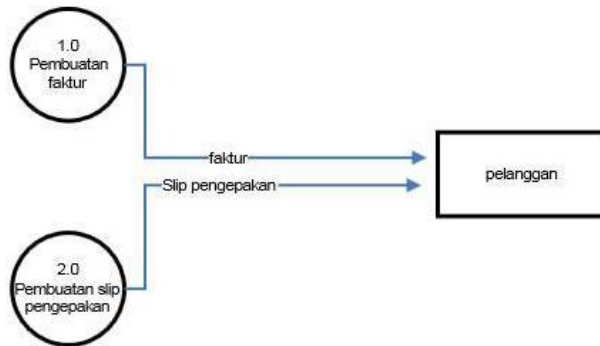


Gambar 5.2 Arus Data Menyebar

5.1.3 Convergen Data Flow (Arus Data Mengumpul)

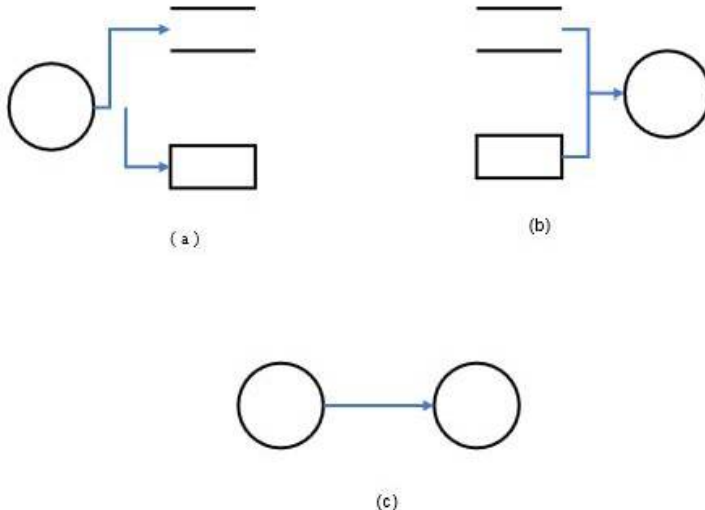
Arus data yang mengumpul, yaitu arus data yang berbeda dari sumber yang berbeda mengumpul ke tujuan yang sama.

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK



Gambar 5.3 Arus Data Mengumpul

Arus data harus dihubungkan pada proses, baik dari maupun yang menuju proses.



Gambar 5.4 Sumber dan Tujuan dalam DAD

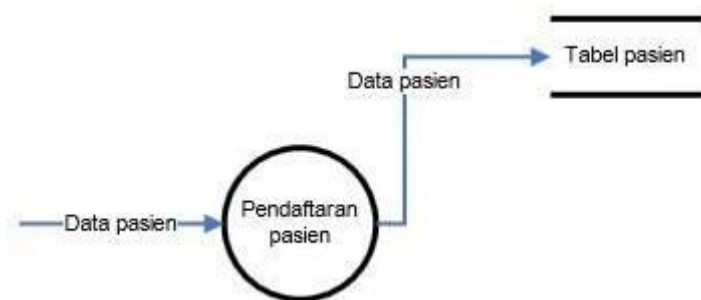
Keterangan:

- Dari proses ke bukan proses.
- Dari bukan proses ke proses.
- Dari proses ke proses.

5.1.5 Data Storage

Merupakan komponen untuk membuat model sekumpulan data. Dapat berupa suatu *file* atau suatu sistem *database* dari suatu komputer, suatu arsip/dokumen, suatu agenda/buku. Yang perlu diperhatikan tentang *data store*:

- Alur data dari proses menuju *data store*, hal ini berarti *data store* berfungsi sebagai tujuan/tempat penyimpanan dari suatu proses (proses *write*).

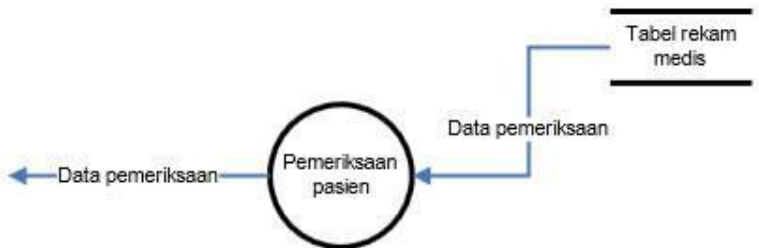


Gambar 5.5 Proses Write

- Alur data dari *data store* ke proses, hal ini berarti *data store* berfungsi sebagai sumber/proses yang

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

memerlukan data (proses *read*).



Gambar 5.6 Proses Read

3. Alur data dari proses menuju *data store* dan sebaliknya berarti berfungsi sebagai sumber dan tujuan.

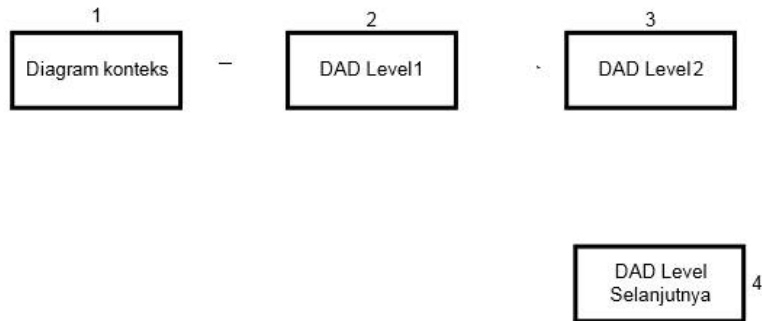


Gambar 5.7 Proses Write dan Read

5.3 DIAGRAM-DIAGRAM DI DALAM DAD

Secara umum konsep untuk menggambarkan DAD sebuah sistem dimulai dari menggambar diagram konteks, DAD Level 1, DAD Level 2, dan level selanjutnya (sesuai kebutuhan).

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK



Gambar 5.8 Alur Penggambaran DAD

Disebut juga diagram tingkat atas, merupakan diagram sistem yang menggambarkan aliran-aliran data yang masuk dan keluar dari sistem dan yang masuk dan keluar dari entitas luar. Hal yang harus diperhatikan:

- Memberikan gambaran tentang seluruh sistem
- Terminal yang memberikan masukan ke sistem disebut *source*
- Terminal yang menerima keluaran disebut *sink/destination*
- Hanya ada satu proses
- Tidak boleh ada *data store*

Merupakan diagram yang merepresentasikan seluruh elemen sistem sebagai sebuah *bubble* tunggal dengan data input dan *output* yang ditunjukkan oleh anak panah

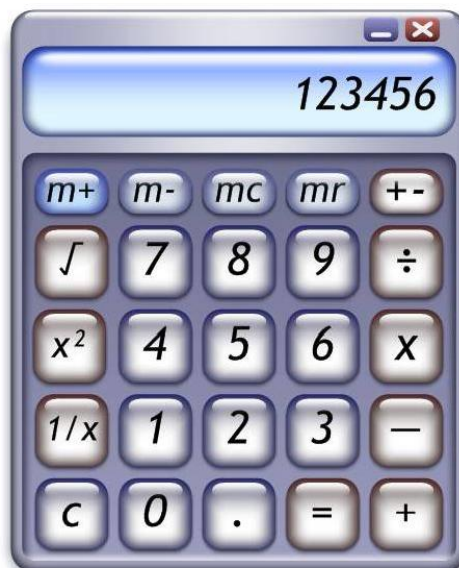
MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

yang masuk dan keluar secara berurutan, meliputi:

- a. Apa saja yang dibutuhkan, mencakup sistem/proses yang dipandang secara keseluruhan.
- b. Siapa saja pihak yang berhubungan langsung dengan sistem (yang memberi dan menerima), mencakup eksternal entitas yang terkait dengan sistem.
- c. Data apa yang diberikan ke sistem (input) dan yang dihasilkan (*output*), mencakup arus data.

SKENARIO 1

Diberikan aplikasi kalkulator, seperti terlihat pada gambar 5.9. Gambarkan diagram konteks dari kalkulator.



Gambar 5.9 Kalkulator

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

- Langkah 1. Menentukan sistem atau proses yang dipandang secara keseluruhan dari gambar (3.9).
Nama sistem/proses “sistem kalkulator” atau “sistem perhitungan”
- Langkah 2. Menentukan pengguna sistem.
Pengguna “user” dalam contoh kasus ini pengguna bisa siapa saja
- Langkah 3. Data apa saja yang diberikan ke sistem.
- Data nilai (angka), operator (fungsi perkalian, pertambahan, dsb.)
- Langkah 4. Menggambar DAD Level Konteks.



Gambar 5.10 Diagram Konteks Kalkulator

SKENARIO 2

Minimarket milik Pak Budi menjual mulai peralatan rumah tangga, alat tulis, dan barang kelontong untuk kebutuhan sehari-hari. Pak Budi ingin membuat sistem yang bisa digunakan untuk transaksi penjualan di kasir. Sebagai seorang analis, Anda diminta untuk menggambarkan

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

diagram konteks dari sistem kasir tersebut

- Langkah 1. Menentukan sistem atau proses yang dipandang secara keseluruhan dari gambar (5.3).

Nama sistem/proses: “sistem informasi *minimarket*”

- Langkah 2. Menentukan pengguna sistem.

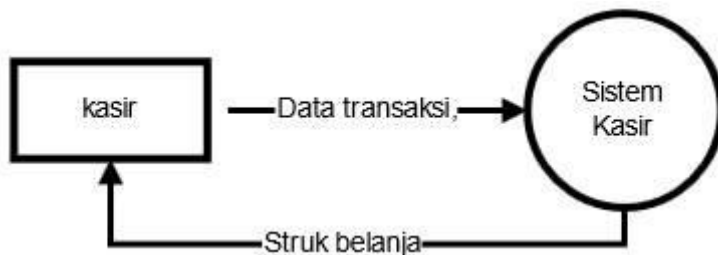
Pengguna sistem: kasir

- Langkah 3. Data apa saja yang diberikan ke sistem.

Input : data transaksi

Output : struk belanja

- Langkah 4. Menggambar DAD Level Konteks.



Gambar 5.11 Diagram Konteks Sistem Kasir

5.1.7 DAD Level I

Setelah pembuatan DAD Level Konteks, selanjutnya adalah pembuatan DAD Level 1, di mana pada DAD Level adalah penggambaran dari diagram konteks yang lebih rinci (*Overview Diagram*) atau biasanya disebut sebagai dekomposisi. Diagram ini

adalah dekomposisi dari diagram konteks.

Cara:

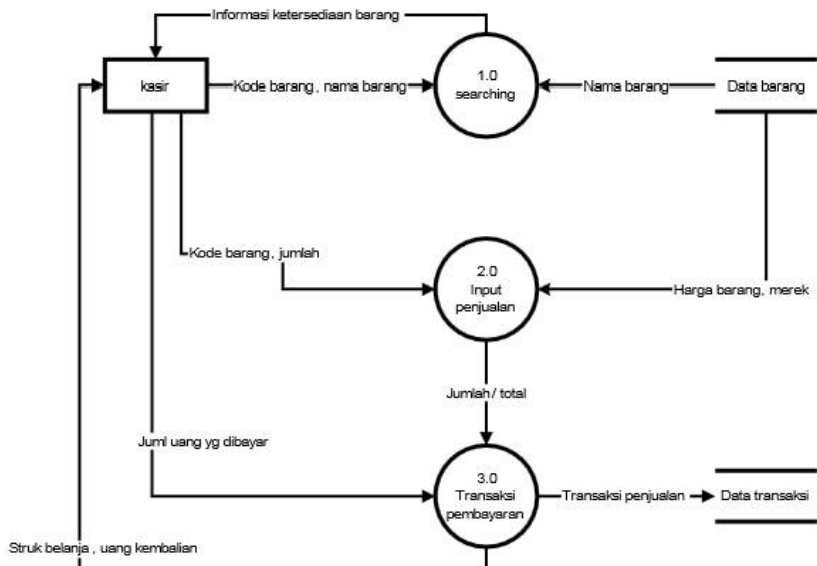
1. Tentukan proses-proses yang ada pada sistem.
2. Tentukan apa yang diberikan/diterima masing-masing proses ke/dari sistem sambil memperhatikan konsep keseimbangan (alur data yang keluar/masuk dari suatu level harus sama dengan alur data yang masuk/keluar pada level berikutnya).
3. Apabila diperlukan, munculkan *data store* sebagai sumber maupun tujuan alur data.
4. Gambarkan diagram level satu.
5. Hindari perpotongan arus data.
6. Beri nomor pada proses utama (nomor tidak menunjukkan urutan proses). Misal, 1.0, 2.0, 3.0, dst.

SKENARIO

Minimarket milik Pak Budi menjual mulai peralatan rumah tangga, alat tulis, dan barang kelontong untuk kebutuhan sehari-hari. Pak Budi ingin membuat sistem yang bisa digunakan untuk transaksi penjualan di kasir. Sebagai seorang analis sistem, Anda diminta untuk menggambarkan DAD Level 1 dari sistem kasir tersebut.

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

- Langkah 1. Menentukan proses-proses/*event* yang terjadi pada kasir.
Proses: proses pencarian barang, input transaksi, pembayaran, cetak struk.
- Langkah 2. Menentukan arus data yang mengalir (input dan *output*) di setiap proses.
- Langkah 3. Menggunakan *storage* untuk menyimpan data.
- Langkah 4: Menggambarkan ke dalam DAD.



Gambar 5.13 DAD Level 1 Sistem Kasir

5.1.8 DAD Level II

DAD Level 2 merupakan diagram yang dibentuk dari dekomposisi proses yang terdapat pada DAD Level 1. Tidak semua proses yang terdapat pada DAD Level 1 harus di *down grade* (dekomposisi) ke dalam DAD Level 2, melainkan sesuai dengan kebutuhan. Jika proses yang terdapat di Diagram Level 1 butuh mencakup banyak proses di dalamnya, maka hal ini perlu dikerjakan ke dalam Diagram Level 2.

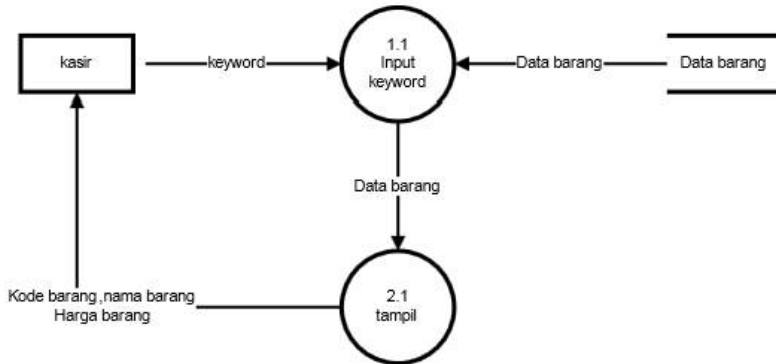
SKENARIO 1

Minimarket milik Pak Budi menjual mulai peralatan rumah tangga, alat tulis, dan barang kelontong untuk kebutuhan sehari-hari. Pak Budi ingin membuat sistem yang bisa digunakan untuk transaksi penjualan di kasir. Sebagai seorang analis sistem, Anda diminta untuk menggambarkan DAD Level 2 dari sistem kasir tersebut.

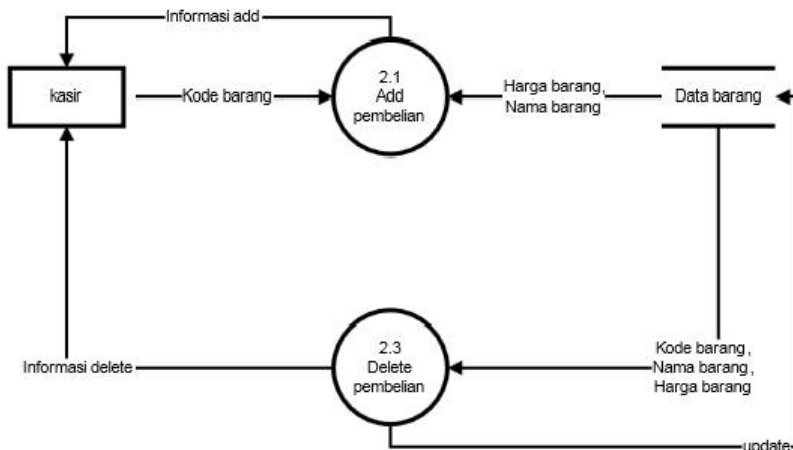
- Langkah 1. Menentukan proses/*event* dari proses yang terdapat pada DAD Level 1. Proses-proses yang terdapat proses *searching* adalah proses input *keyword* dan proses tampil.
- Langkah 2. Menentukan arus data yang mengalir (input dan *output*) di setiap proses.

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

- Langkah 3. Menggunakan *storage* untuk menyimpan data.
- Langkah 4. Menggambarkan ke dalam DAD



Gambar 5.15 DAD Level 2 Proses Searching



Gambar 5.16 DAD Level 2 Proses Input Penjualan

5.1.9 DAD Level 3 dan Level Selanjutnya

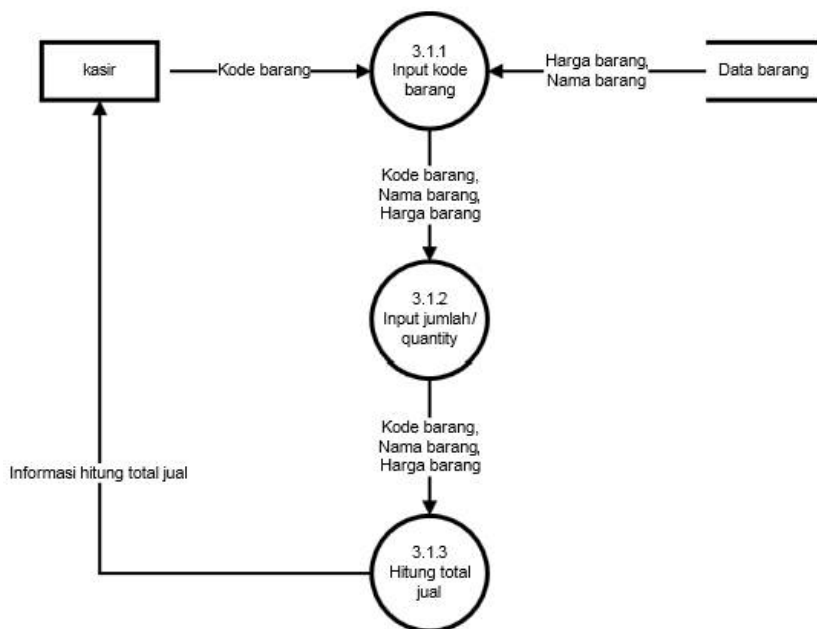
DAD Level 3 merupakan diagram yang dibentuk dari dekomposisi proses yang terdapat pada DAD Level 2.

Catatan: DAD Level 3, Level 4, dst. merupakan dekomposisi dari level sebelumnya. Proses dekomposisi dilakukan sampai dengan proses siap dituangkan ke dalam program. Aturan yang digunakan sama dengan DAD Level 2.

SKENARIO

- Langkah 1. Menentukan proses/*event* dari proses yang terdapat pada DAD Level 2. Proses-proses yang terdapat pada proses *searching* adalah proses input *keyword* dan proses tampil.
- Langkah 2. Menentukan arus data yang mengalir (input dan *output*) di setiap proses.
- Langkah 3. Menggunakan *storage* untuk menyimpan data.

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK



Gambar 5. 17 DAD Level 3 Proses Add Pembelian



Rangkuman

Microsoft Visio adalah salah satu *software* yang berfungsi untuk membuat desain berupa *flowchart*, diagram, bagan, dan lain-lain. Dengan Visio, semua proses pekerjaan akan lebih mudah terutama bagi kalangan IT dan analis. Bahkan untuk orang awam sekalipun, Visio mudah digunakan.



Soal Latihan

-
-
1. Gambarkan diagram konteks dari *search engine* Google.
 2. Pada saat ingin menggunakan Facebook, pengguna diwajibkan untuk mendaftar terlebih dahulu (*sign up*). *User* harus menginputkan data-data pribadi untuk proses pendaftaran. Gambarkan diagram konteks dari proses pendaftaran pada Facebook.
 3. Gambarkan diagram konteks dan DAD Level 1 untuk sistem ATM.
 4. Sistem apotek ini bisa digunakan oleh apoteker untuk melayani berbagai macam penjualan obat. Sistem ini dapat juga digunakan oleh asisten apoteker untuk manajemen data stok obat. Lakukan

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

analisis sistem, kemudian gambarkan diagram konteks dan Level 1 dari sistem informasi apotek tersebut.

5. Di sebuah daerah yang jauh dari perkotaan, terdapat sebuah perpustakaan. Semakin hari pengguna perpustakaan tersebut semakin bertambah, sehingga pemilik perpustakaan membutuhkan aplikasi komputer yang dapat digunakan untuk aktivitas di perpustakaan. Sebagai seorang analis, Anda diminta untuk membuat perancangan perangkat lunak, meliputi:
 - a. Analisis Sistem
 - b. Analisis Kebutuhan Fungsional dan Nonfungsional
 - c. DAD (Konteks, DAD Level 1, DAD Level 2)

Ilustrasi aktivitas yang terjadi pada perpustakaan dapat anda lihat pada gambar 5.18. di bawah ini.

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK



Gambar 5.18. Ilustrasi aktivitas yang terjadi pada perpustakaan

BAB VI

BAGAN ALIR (FLOWCHART)



Overview

Mahasiswa dapat menjelaskan fungsi *flowchart* dalam perancangan sistem serta dapat membedakan aplikasi mana yang menggunakan *flowchart* sebagai *tools* perancangan sistem. Mahasiswa akan mempelajari jenis *flowchart* serta aplikasi yang sesuai dan membuat *flowchart* dari studi kasus.



Tujuan

1. Mengenalkan *flowchart* sebagai *tool* perancangan sistem.
2. Pengenalan jenis-jenis *flowchart* dan aplikasi yang sesuai.
3. Mengenalkan studi kasus dengan menggunakan *tools flowchart*.

6.1 Flowchart

Flowchart merupakan alat bantu yang bisa digunakan untuk kegiatan analisis sistem dan perancangan (desain) sistem. Suatu skema representasi suatu proses atau algoritma. *Flowchart* merupakan salah satu *tool* yang digunakan untuk *quality control*. *Flowchart* adalah bentuk gambar/diagram yang mempunyai aliran satu atau dua arah secara sekuensial. *Flowchart* digunakan untuk merepresentasikan maupun mendesain program. Oleh karena itu *flowchart* harus bisa merepresentasikan komponen-komponen dalam bahasa pemrograman. Baik *flowchart* maupun algoritma bisa dibuat sebelum maupun setelah pembuatan program.

Flowchart dan algoritma yang dibuat sebelum membuat program digunakan untuk mempermudah pembuat program untuk menentukan alur logika program, sedangkan yang dibuat setelah pembuatan program digunakan untuk menjelaskan alur program kepada orang lain. *Flowchart* berbeda dengan DAD, yang paling jelas adalah dari tahapan/langkah alur data serta simbol-simbol yang digunakan. Untuk itu, perhatikan penjelasan berikut sehingga bisa mengetahui apa saja perbedaan yang ada pada *flowchart* dan DAD.

6.2 Macam-Macam Flowchart



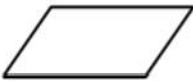

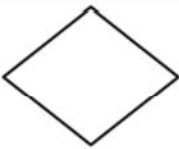
- a. Bagan Alir Sistem (*System Flowchart*) merupakan bagan yang

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

menunjukkan arus pekerjaan secara keseluruhan dari sistem.
Contoh: pendaftaran pasien di RS, pendaftaran mata kuliah praktikum, dll.

- b. Bagan Alir Dokumen (*Document Flowchart*) merupakan bagan alir yang menunjukkan arus data dari laporan dan formulir-formulir termasuk tembusannya. Contoh: pelaporan bulanan perusahaan.
- c. Bagan Alir Skematik (*Schematic Flowchart*) menggambarkan prosedur di dalam sistem. Bagan ini menggunakan simbol bagan alir sistem, juga menggambarkan komputer dan peralatan lainnya. Contoh: bagan alir proses robot, bagan alir proses pencetakan dokumen.
- d. Bagan Alir Program (*Program Flowchart*) merupakan bagan yang menjelaskan secara rinci langkah-langkah dari proses program. Contoh: bagan alir untuk proses penghitungan faktorial, bagan alir untuk proses penghitungan suhu ruangan.
- e. Bagan Alir Proses (*Process Flowchart*) merupakan bagan alir yang banyak digunakan di teknik (misal, teknik industri).

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

Nama	Simbol	Keterangan
Oval		Menunjukkan notasi untuk awal dan akhir bagan alir
Aliran data		Menunjukkan petunjuk dari aliran fisik pada program
Data		Menunjukkan suatu operasi input atau suatu operasi output.
Rectangle		Menunjukkan suatu proses yang akan digunakan
Diamond		Menotasikan suatu keputusan (atau cabang) yang akan dibuat. Program akan memilih satu dari dua rute.

Tabel 6.1 Simbol Flowchart

Keterangan:

- Tanda oval, biasanya disebut sebagai *start and end symbol*. Biasanya diberi nama “mulai” dan “selesai”, atau “start” dan “end”, ataupun frasa lain yang menunjukkan bahwa program/bagan alir tersebut mulai dan selesai.
- *Diamond* sebagai tanda pemilihan adalah *conditional (or decision)*. Simbol ini mengandung pertanyaan “yes/no” atau “ya/tidak” atau “benar/salah”. Simbol ini memiliki dua

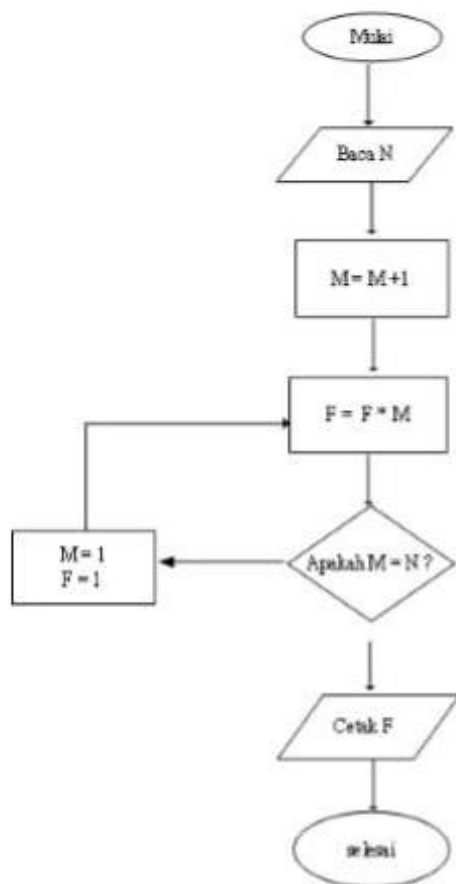
MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

simbol aliran data yang keluar, biasanya dari sudut yang bawah dan sudut yang kanan. Satunya menyatakan “benar (*true*)” dan satunya menyatakan “salah (*false*)”.

SKENARIO

Menghitung faktorial N ($N!$) di mana $N! = 1*2*3*4*5 \dots N$. N Fungsi dari bagan alir untuk menghitung N faktorial ini akan mempermudah kita dalam menuliskan pada program komputer, karena dari bagan alir ini, kita akan mudah untuk memahami algoritma yang digunakan, karena *flowchart* ini merupakan tahapan suatu instruksi seperti halnya algoritma suatu program.

- Langkah 1. Tentukan inisialisasinya.
Misalnya, N = bilangan faktorial yang dicari, M = bilangan 1, 2, 3, 4, ... N
- Langkah 2. Tentukan rumus untuk menghitung faktorial.
- Langkah 3. Gambarkan ke dalam *flowchart*.



Gambar 6.1 Flowchart



Soal Latihan

1. Gambarkan. Buatlah *flowchart* rental VCD!

BAB VII

BAGAN ALIR (FLOWCHART) II



Overview

Mahasiswa dapat membuat dokumen perancangan sistem secara lengkap, mendokumentasikan analisis sistem, mendokumentasikan analisis kebutuhan fungsional dan kebutuhan nonfungsional, mendokumentasikan DAD, dan mendokumentasikan *flowchart*.



Tujuan

1. Membuat hal-hal yang berkaitan dengan perancangan sistem.
2. Mendokumentasikan hal-hal yang berkaitan dengan perancangan sistem.

7.1 Pendahuluan

Dalam membangun perangkat lunak dilakukan perancangan untuk melakukan proses pembuatan maupun pengembangan sistem. Proses tersebut antara lain mulai dari menganalisis sistem yang akan dibangun/dikembangkan, kemudian membuat perancangan data. Yaitu dengan membuat analisis sistem, batasan sistem, DAD, dan *flowchart* yang akan digunakan untuk membangun atau mengembangkan perangkat lunak tersebut.

SKENARIO

- Langkah 1. Menuliskan analisis sistem.
- Langkah 2. Menuliskan analisis kebutuhan fungsional dan nonfungsional sistem.
- Langkah 3. Menggambarkan DAD lengkap (diagram konteks, DAD Level 1, DAD Level 2, dst. sesuai dengan kebutuhan).
- Langkah 4. Menggambarkan *flowchart*.



Soal Latihan

1. Buatlah dokumentasi perancangan sistem informasi perpustakaan yang telah Anda kerjakan di BAB III.
2. Buatlah dokumentasi perancangan sistem informasi akademik pada kampus Anda.

BAB VIII

USE CASE DIAGRAM



Overview

Mahasiswa dapat membedakan metode perancangan perangkat lunak berbasis objek dengan yang konvensional. Mahasiswa dapat mengoperasikan *tool* pembuatan UML. Mahasiswa akan mempelajari fungsi *use case diagram* dan semua notasi yang digunakan di dalamnya serta dapat membuat *use case diagram* untuk contoh kasus yang diberikan.



Tujuan

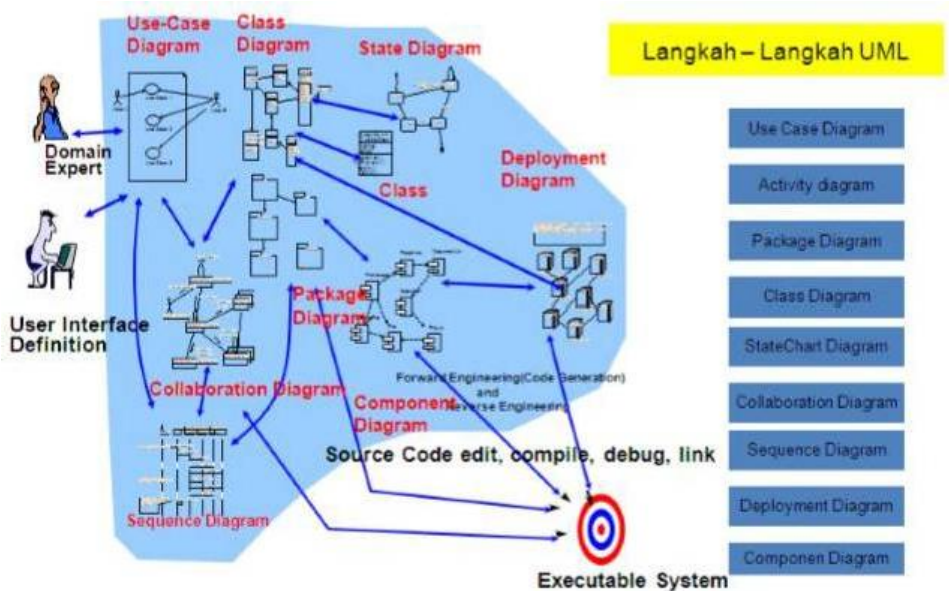
1. Mengenalkan UML sebagai paradigma baru perancangan perangkat lunak.
2. Mengenalkan konsep *use case diagram*.
3. Menjelaskan cara membuat *use case diagram* menggunakan *tool*.

8.1 Unified Modelling Language (UML)

Unified Modelling Language (UML) adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang, dan mendokumentasikan sistem perangkat lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML, kita dapat membuat model untuk semua jenis aplikasi perangkat lunak, di mana aplikasi tersebut dapat berjalan pada perangkat keras, sistem operasi dan jaringan apa pun, serta ditulis dalam bahasa pemrograman apa pun.

Namun, karena UML juga menggunakan *class* dan *operation* dalam konsep dasarnya, maka ia lebih cocok untuk penulisan perangkat lunak dalam bahasa-bahasa berorientasi objek seperti C++, Java, C# atau VB.NET. Walaupun demikian, UML tetap dapat digunakan untuk pemodelan aplikasi prosedural dalam VB atau C.

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK



Gambar 8.1 Use Case Model Tabel 8.1 Use Case Model

Use Case Text	Use Case Diagram
<ul style="list-style-type: none"> • Lebih detail • Tidak ada visualisasi (berbentuk teks) • Cocok bagi developer 	<ul style="list-style-type: none"> • Tidak ada visualisasi (berbentuk teks) • Cocok bagi developer • Lebih abstrak, kurang detail • Bentuk visual (gambar) • Cocok untuk klien

8.2 Use Case Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah “apa” yang diperbuat sistem dan bukan “bagaimana”. *Use case* merepresentasikan interaksi antara aktor dengan sistem. *Use case* menggambarkan pekerjaan tertentu, misalnya *login* ke sistem, membuat daftar belanja, dan sebagainya.

Aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. *Use case diagram* dapat sangat membantu ketika kita sedang menyusun analisis kebutuhan sebuah sistem, mengomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua fitur yang ada pada sistem. Kadangkala notasi *use case* kurang detail, terutama untuk beberapa kegiatan tertentu.

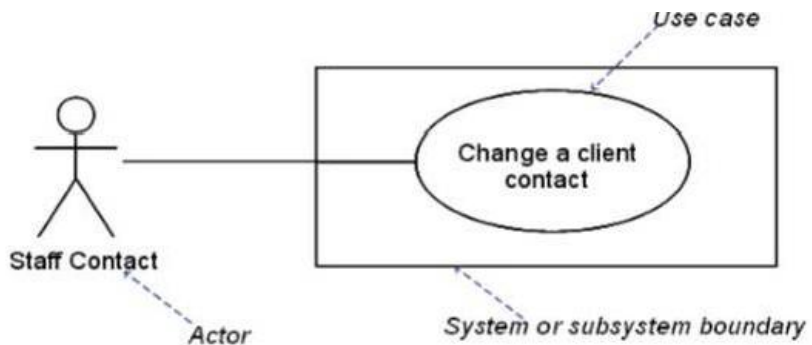
Use case dapat memasukkan (*include*) fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali *use case* yang meng-*include* dieksekusi secara normal. Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang sejenis.

Use case juga dapat meng-*extend* *use case* lain dengan

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

behaviour-nya sendiri. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain.

Use case diagram menggambarkan interaksi antara aktor dengan proses atau sistem yang dibuat. *Use case diagram* mempunyai beberapa bagian penting, seperti *actor*, *use case*, *association*, *generalization*.



Gambar 8.2 Actor, Use Case, dan System Boundary-nya

- Actor merupakan bagian dari *use case* yang bertindak sebagai subjek (pelaku) dalam suatu proses.
- Use case* adalah proses-proses yang terjadi dalam suatu *software*. *Use case* juga menggambarkan apa yang sedang dilakukan oleh seorang *actor*.
- Relasi menggambarkan hubungan antara *actor* dan *use case*.



Soal Latihan

1. Buatlah dokumentasi perancangan sistem informasi perpustakaan yang telah Anda kerjakan di BAB III.
2. Buatlah dokumentasi perancangan sistem informasi akademik pada kampus Anda.

Daftar Pustaka

- Garcia-Molina, H. 2008. *Database systems: the complete book*. Pearson Education India.
- Holzner, S. 2007. *Special edition using microsoft® office visio® 2007*. Que Corp.
- Lemke, J. 2004. *Microsoft Office Visio 2003 step by step*. Microsoft Press.
- Li, Q. and Chen, Y.L. 2009. *Data flow diagram*. In *Modeling and Analysis of Enterprise and Information Systems* (pp. 85-97). Springer, Berlin, Heidelberg.
- Lyu, M.R. 1996. *Handbook of software reliability engineering* (Vol. 222). CA: IEEE computer society press.
- Lyu, M.R. 2007, May. *Software reliability engineering: A roadmap*. In *Future of Software Engineering (FOSE'07)* (pp. 153-170). IEEE.
- Musa, J.D. 2004. *Software reliability engineering: more reliable software, faster and cheaper*. Tata McGraw-Hill Education.
- Nassi, I. and Shneiderman, B. 1973. *Flowchart techniques for structured programming*. ACM Sigplan Notices, 8(8), pp.12-26.
- Phillips, C.L. and Nagle, H.T. 2007. *Digital control system analysis and design*. Prentice Hall Press.
- Qamariyah, N., Hermadi, I. and Buono, A. 2012. *Development of Learning Simulation System for Dentistry Student (A case Study in Dentistry Students of Brawijaya University)*.

MODUL PRAKTIKUM : REKAYASA PERANGKAT LUNAK

- Rhoads, G.B., Digimarc Corp. 2000. *Computer system linked by using information in data objects*. U.S. Patent 6,122,403.
- Roos, R. 2002. *Java Data Objects*. Pearson Education.
- Saadat, H. 1999. *Power system analysis*. WCB/McGraw-Hill.
- Seacord, R.C., Plakosh, D. and Lewis, G.A. 2003. *Modernizing legacy systems: software technologies, engineering processes, and business practices*. Addison-Wesley Professional.
- Triandini, E. and Suardika, I.G. 2012. *Step by Step Desain Proyek Menggunakan UML*. Penerbit Andi.
- von der Maßen, T. and Lichter, H. 2002, September. *Modeling variability by UML use case diagrams*. In Proceedings of the International Workshop on Requirements Engineering for product lines (pp. 19-25).
- Wong, S. 2007. *StarUML Tutorial*. Connexions Web site, Sep.

Biodata Penulis

Muhammad Wali, saat ini bekerja sebagai dosen dan memegang posisi Lembaga Penelitian dan Pengabdian Masyarakat (LPPM) di AMIK Indonesia. Ia juga bekerja sebagai konsultan pengembangan perangkat lunak di Provinsi Aceh dan aktif dalam berbagai komunitas sosial teknologi informasi di Indonesia.