RESUME TEORI
MODEL DRIVEN ENGINEERING

Prabowo Gosal 52015031

STMIK KHARISMA Makassar
Program studi INFORMATIKA 2015

Apa itu Software?

Software adalah istilah khusus untuk data yang diformat, dan disimpan secara digital, termasuk program komputer, dokumentasinya, dan berbagai informasi yang bisa dibaca, dan ditulis oleh computer yang dibuat oleh Software Developer

Menurut Roger S. Pressman lagi, *Software* adalah sebuah produk yang dibangun *Developer Professional* dan kemudian mendukung semua perangkat lunak dalam jangka waktu yang panjang (Pressman, 2005).

Pengertian perangkat lunak menurut Al Bahra bin Ladjamudin (2006:3) menjelaskan bahwa perangkat lunak adalah objek tertentu yang dapat dijalankan seperti kode sumber, kode objek atau sebuah program yang lengkap

Karakteristik Software

1. Perangkat lunak dibangun dan dikembangkan, tidak dibuat dalam bentuk klasik.
2. Perangkat lunak tidak pernah usang
3. Sebagian besar perangkat lunak dibuat secara custum-built, serta dapat dirakit dari komponen yang sudah ada.

Jenis-jenis Software

1. Perangkat Lunak Sistem (System Software)

   Perangkat lunak system merupakan sekumpulan program yang ditulis untuk melayani program-program yang lain.
2. Perangkat Lunak Real-Time (Real Time Software)

   Program-program yang memonitor / menganalisis / mengontrol kejadian-kejadian yang sesungguhnya (sedang berlangsung) pada dunia disebut dengan perangkat lunak real-time.
3. Perangkat Lunak Bisnis (Business Software)

   Pemrosesan informasi bisnis merupakan area aplikasi yang paling luas. Aplikasi dalam area ini akan menyusun kembali struktur data yang ada dengan cara tertentu untuk memperlancar operasi bisnis atau pengambilan keputusan manajemen.

4. Perangkat Lunak Teknik dan Ilmu Pengetahuan (Engineering and Scientific Software)

Perangkat lunak teknik dan ilmu pengetahuan ditandai dengan algoritma number crunching. Perangkat lunak ini memiliki jangkauan aplikasi mulai dari astronomi sampai vulkanologi, dari analisis otomotif sampai dinamika orbit pesawat luar angkasa, dan dari biologi molecular sampai pada pabrik yang sudah diotomatisasi.

5. Perangkat Lunak Embedded (Embedded Sofware)
   Embedded software ada dalam real-only memory dan dipakai untuk mengontrol hasil serta system untuk keperluan konsumen dan pasar industry.
6. Perangkat Lunak Komputer Personal (Personal Computer Software)
   Pasar perangkat lunak komputer personal telah berkembang selama decade terakhir. Pengolahan kata, spreadsheet, grafik komputer, multimedia, hiburan, manajaemen database, aplikasi keuangan bisnis dan personal, jaringan eksternal atau akses database merupakan beberapa contoh yang dapat disebutkan.
7. Perangkat Lunak kecerdasan Buatan (Artifical Intelligence Sofware{AI})
   Perangkat lunak kecerdasan buatan (Artifical Intelligence Sofware{AI}) menggunakan algoritma non-numerik untuk memecahkan masalah kompleks yang tidak sesuai untuk perhitungan atau analisis secara langsung. Area kecerdasan buatan yang aktif adalah sistem pakar (expert system), sering disebut dengan sistem berbasis ilmu pengetahuan.

Software Process

Pengertian Software Process

1. Menurut Reidar Conradi pada bukunya yang berjudul Software Process Technology: 7th European Workshop, Software Process adalah Proses perangkat lunak yang direpresentasikan sebagai sebuah proses program dan dapat mengeksekusi secara otomatis.
2. Menurut Ian Sommerville pada bukunya yang berjudul Software Engineering (9th Edition), Software Process adalah Pendekatan sistematis yang digunakan dalam rekayasa perangkat lunak serta merupakan sebuah aktifitas terurut yang menuju kepada produksi dari pembuatan produk rekayasa perangkat lunak.
3. Sedangkan menurut M Deploy pada tulisannya yang berjudul Software Engineering Process, Software process adalah serangkaian langkah-langkah yang terurut untuk membuat sebuah produk rekayasa perangkat lunak. Software process adalah suatu hirarki yang tiap langkah dapat memiliki sub-sub langkah dalam proses rekayasa perangkat lunak.

Terdapat 4 aktifitas umum yang mendasar pada semua proses rekayasa perangkat lunak, yaitu:

1. Software specification, yaitu pengguna dan perekayasa menentukan perangkat lunak yang akan dibuat dan dibatasi pada proyek tersebut.
2. Software development, dimana perangkat lunak tersebut dirancang dan diprogram.
3. Software validation, dimana perangkat lunak di cek apakah sudah memenuhi apa yang dibutuhkan oleh pengguna
4. Software evolution, dimana perangkat lunak diubah,diperbaiki untuk mengatasi perubahan pengguna dan mengikuti perkembangan jaman.

Model Proses Software

Model proses software adalah representasi abstrak dari proses. Merupakan gambaran dari proses dari beberapa perspektif tertentu

Model Proses Software Generik antara lain :

1. Model waterfall

Membagi dan membedakan fase spesifikasi dan pengembangan.Langkah-langkah pada model ini adalah :

- Analisa dan definisi kebutuhan
- Desain sistem dan software
- Implementasi dan unit testing
- Integrasi dan testing sistem
- Operasi dan maintenance

Kekurangan dari model waterfall adalah kesulitan untuk mengakomodasi perubahan setelah proses berjalan.

Masalah yang sering dihadapi pada model ini, antara lain :

1) Tidak fleksibel dalam pembagian proyek ke dalam tingkat yang berbeda
2) Sulit untuk merespon perubahan kebutuhan konsumen, sehingga model ini hanya cocok jika kebutuhan sudah dimengerti dengan baik

2. Pengembangan Evolusioner

Spesifikasi dan pengembangan yang terpisah, ada 2 methode dalam Pengembangan Evolusioner, yaitu:

- Pengembangan Exploratory
  bekerja dengan konsumen dan melibatkan sistem akhir dari spesifikasi skema inisial. Dimulai dengan kebutuhan yang dimengerti dengan baik
- Throw-away prototyping
  Berkonsentrasi pada eksperiment serta mengerti kebutuhan sistem. Dimulai dengan kebutuhan yang tidak dimengerti dengan baik

Permasalahan

- Tidak ada visibilitas proses
- Sistem biasanya tidak terstruktur dengan baik
- Kemampuan khusus (misalnya bahasa untuk prototipe cepat) kemungkinan diperlukan

Aplikasi

- Untuk sistem interaktif berukuran kecil atau tingkat medium.
- Untuk bagian dari sistem besar (misalnya user interface).
- Untuk sistem dengan daur hidup pendek.

3. Pengembangan sistem Formal

Model sistem matematis yang secara formal diterjemahkan ke dalam implementasi.

Pengembangan Sistem Formal Berbasis transformasi dari spesifikasi matematis melalui representasi yang berbeda untuk program yg dapat dieksekusi. Sedangkan transformasi merupakan ‘pemelihara kebenaran’ sehingga dapat menunjukkan program sesuai spesifikasinya.

Permasalahan

- Perlu kemampuan dan training khusus untuk mengaplikasikan teknik ini
- Secara formal sulit untuk menentukan beberapa aspek dari sistem seperti antarmuka user

Aplikasi

Sistem kritis terutama dimana keselamatan dan keamanan harus dibuat sebelum sistem beroperasi

4. Pengembangan Reuse-based

Penembangan sistem ini Berbasis systematic reuse dimana sistem diintegrasikan dalam komponen yang sudah ada atau sistem COTS (Commercial-off-theshelf).

Tingkatan/ Level Proses pada pengembangan Reuse-based antara lain:

- Analisakomponen
- Modifikasi kebutuhan
- Desain sistem dengan reuse
- Pengembangan dan integrasi

Kebutuhan sistem selalu berkembang selama proyek berlangsung, sehingga iterasi proses dimana level sebelumnya dilakukan rework merupakan bagian dari proses untuk sistem yang besar.

pada proses ini ada 2 pendekatan yang dapat dilakukan, yaitu:

1. Pengembangan Incremental

   Pelepasan sistem tidak dalam bentuk pelepasan tunggal, tetapi pengembangan dan pelepasan dibagi ke dalam ‘increment’ dimana setiap ‘increment’ melepaskan bagian dari fungsional yang dibutuhkan. Kebutuhan user diprioritaskan dan kebutuhan prioritas tertinggi akan dimasukkan dalam ‘increment’ awal. Jika pengembangan ‘increment’ dimulai, kebutuhan dibekukan terlebih dahulu dan setelah itu kebutuhan untuk ‘increment’ selanjutnya dapat dilanjutkan.

   Keuntungan Pengembangan Incremental:

   - Nilai konsumen dapat diserahkan pada setiap ‘increment’ sehingga fungsional sistem tersedia lebih dahulu.
   - ‘increment’ awal berfungsi sebagai prototype untuk membantu memperoleh kebutuhan ‘increment’ selanjutnya.
   - Resiko lebih rendah dari keseluruhan kegagalan proyek.
   - Layanan sistem prioritas tertinggi cenderung menerima testing terbanyak

2. Pengembangan Spiral

   Proses direpresentasikan sebagai spiral bukan sebagai urutan aktivitas dengan melihat sistem sebelumnya (backtracking). Setiap loop dalam spiral merepresentasikan fase dalam proses. Tidak ada fase yang tetap seperti spesifikasi atau desain- loop , dalam

spiral dipilih tergantung pada apa yang dibutuhkan. Resiko ditaksir secara eksplisit dan penyelesaian sepanjang proses.

Tiap loop dalam Spiral terdiri dari beberapa Sektor, yaitu :

1. Setting Obyektif
   yaitu : menentukan tujuan dari Fase yan telah ditentukan.Batasan-batasan pada proses dan produk serta resiko telah diketahui. Alternatif strategi telah disiapkan berdasarkan resiko-resiko yang telah diketahui dan sudah direncanakan.
2. Penaksiran dan pengurangan resiko
   Resiko ditaksir secara detail dan aktifitas didigunakan untuk mengurangi resiko.
3. Pengembangan dan Validasi
   Model pengembangan untuk sistem dipilih yang berupa model generik.
4. Perencanaan
   Proyek direview dan fase berikutnya dari spiral direncanakan.

Timeline Programing Language

Sebelum 1940

Pada jaman ini terdapat bahasa pemrograman yang pertama kali muncul sebelum adanya komputer modern, artinya bahasa pemrograman lebih tua dari komputer itu sendiri. Pada awal kemunculannya, bahasa pemrograman masih dalam bentuk kode-kode bahasa mesin.

Bahasa mesin merupakan bahasa yang terdiri atas kode-kode mesin dan hanya dapat diinterpretasikan langsung oleh mesin komputer. Bahasa mesin ini tergolong bahasa tingkat rendah, karena hanya berupa kode 0 dan 1 seperti disampaikan pada bagian atas.

Periode 1940-an

Dengan bahasa mesin ditemukan banyak kesulitan untuk pengembangan dan perbaikan pada program yang dibuat saat itu, Tahun 1940-an komputer bertenaga listrik dibuat, dengan kecepatan yang sangat terbatas dan kapasitas memori yang mencukupi untuk programmer memprogram, kemudian terciptalah bahasa assembly (Assembly language).

Bahasa assembly adalah bahasa simbol dari bahasa mesin. Setiap kode bahasa mesin memiliki simbol sendiri dalam bahasa assembly.

Misalnya

- Move untuk memindahkan isi data,
- ADD untuk penjumlahan,
- MUL untuk perkalian,
- SUB untuk pengurangan, dan lain-lain.

Penggunaan bahasa Asembly dirasa belum sempurna karena selain sulit untuk diimplementasikan, ternyata bahasa ini juga sulit jika sang programer ingin mengembangkan program buatannya.

Pada tahun 1948, Konrad Zuse mempublikasikan sebuah paper tentang bahasa pemrograman miliknya yakni Plankalkül. Bagaimanapun, bahasa tersebut tidak digunakan pada masanya dan terisolasi terhadap perkembangan bahasa pemrograman yang lain.

Beberapa bahasa pemrograman yang berkembang pada masa itu antara lain:

- Plankalkül (Konrad Zuse) – 1943
- ENIAC coding system – 1943
- C-10 – 1949

Periode tahun 1950-an sampai dengan tahun 1960-an

Mulai tahun 1950 dibuatlah bahasa pemrograman modern, yang turun-temurun dan tersebar luas hingga saat ini. Bahasa ini menggunakan istilah atau reserved word yang dekat dengan bahasa manusia seperti

- READ untuk membaca,
- WRITE untuk menulis dsb.

Dalam perkembangannya Bahasa Tingkat Tinggi juga terdiri dari beberapa metode pemrograman, yaitu Procedural Programing dan Object Oriented Programing. Letak perbedaannya yaitu, jika pada procedural programing program dijalankan dengan menggabungkan variable, procedure-procedure yang saling keterkaitan dan berjalan berurut, sedangkan pada OOP seluruh task dijalankan berdasarkan kedalam object.

- FORTRAN (1955), the “FORmula TRANslator”, ditemukan oleh John W. Backus dll.
- LISP, the “LISt Processor”, ditemukan oleh John McCarthy dll.
- COBOL, the COmmon Bussines Oriented Language, dibuat oleh the Short Range Commitee, dan Grace Hopper berperan sangat besar disini.

Overview:

- Regional Assembly Language – 1951
- Autocode – 1952
- FORTRAN – 1954
- FLOW-MATIC – 1955
- COMTRAN – 1957
- LISP – 1958
- ALGOL – 1958
- COBOL – 1959
- APL – 1962
- SIMULA – 1962

- BASIC – 1964
- PL/I -1964

Periode 1967-1978: Menetapkan Paradigma Fundamental

Periode diantara tahun 60-an sampai dengan 70-an membawa pengaruh yang besar dalam perkembangan bahasa pemrograman.

Kebanyakan dari pola bahasa pemrograman yang utama yang saat ini banyak digunakan:

- Simula, ditemukan pada akhir 60-an oleh Nygaard dan Dahl sebagai superset dari Algol 60, merupakan bahasa pemrograman pertama yang didesain untuk mendukun pemrograman berorientasi object.
- C, sebuah tahapan awal dari sistem bahsa pemrograman, yang dikembangkan oleh Dennis Ritchie dan Ken Thompson di Bell Labs antara tahun 1969 dan 1973.
- Smalltalk (pertengahan tahun 70-an) menyajikan desain ground-up yang lengkap dari sebuah bahasa yang berorientasi objek.
- Prolog, didesain pada tahun 1977 oleh Colmerauer, Roussel, and Kowalski, merupakan bahasa pemrograman logika yang pertama.
- ML membangun sebuah sistem polimorfis (ditemukan oleh Robin Miller pada tahun 1973) diatas sebuah Lisp, yang merintis bahasa pemrograman fungsional bertipe statis.

Beberapa bahasa pemrograman yang berkembang dalam periode ini termasuk:

- Pascal – 1970
- Forth – 1970
- C – 1970
- Smaltalk – 1972
- Prolog – 1972
- ML – 1973
- SQL – 1978

Periode 1980-an: konsolidasi, modul, performa

1980s adalah tahun dari konsolidasi relatif. C++ dikombinasikan dengan sistem programming dan berorientasi obyek. Pemerintah Amerika Serikat menstandardisasi Ada, sebuah sistem pemrograman yang bertujuan untuk digunakan para kontraktor untuk bertahan. Di Jepang

dan di tempat lain, penjumlahan luas yang telah di selidiki disebut" generasi ke lima" bahasa-bahasa yang menyatukan logika pemrograman konstruksi.

Masyarakat bahasa fungsional gerak ke standarisasi ML dan Cedal. Dibandingkan dengan menemukan paradigma-paradigma baru, semua pergerakan ini menekuni gagasan-gagasan yang ditemukan di dalam dekade sebelumnya.

Bagaimanapun, satu kecenderungan baru di dalam disain bahasa adalah satu fokus yang ditingkatkan di pemrograman untuk sistem besar-besaran melalui penggunaan dari modul, atau kesatuan organisasi besar-besaran dari kode. Modula, Ada, dan ML semua sistem modul terkemuka yang dikembangkan pada 1980-an.

Beberapa bahasa pemrograman yang berkembang dalam periode ini termasuk:

- Ada – 1983
- C++ – 1983
- Eiffel – 1985
- Perl – 1987
- FL (Backus) – 1989

Periode 1990-an: Visual

Pada periode ini bahasa selain berorientasi objek juga sudah dikembangkan berbasi Visual sehingga semakin mudah untuk membuat program aplikasi, diawali oleh Python dan Microsoft Visual Basic 1 pada tahun 1991, Delphi yang dikembangkan dari Pascal for windows akhirnya pada tahun 1997 Visual Basic 5 diluncurkan dengan kemudahan koneksi ke database, OO Cobol sudah ditemukan dalam versi windows. Bagi kebanyakan programmer database tidak dapat dipungkiri bahwa era 1990an merupakan era yang paling produktif semenjak bahasa pemrogrammar diciptakan.

Beberapa bahasa pemrograman yang berkembang dalam periode ini termasuk

- Haskel – 1990
- Python – 1991
- Java – 1991
- Ruby – 1993
- OO Cobol

- Lua – 1993
- ANSI Common Lisp – 1994
- •avaScript – 1995
- PHP – 1995
- C# – 2000
- JavaFX Scrip, Live Script,
- Visual Basic

Periode 2000an hingga tulisan ini dibuat

Pada saat ini ada kecenderungan para vendor bahasa pemrograman untuk menggiring programmer hanya dengan mengggunakan produk mereka untuk membuat program meski kita sadari bahwa sulit rasanya untuk membuat program yang tangguh hanya dengan satu bahasa pemrograman, hal ini tentunya dilakukan dengan tujuan kelangsungan usaha mereka, namun terlepas dari semua itu terdapat dua konsepsi besar dalam periode ini dimana kemudahan berbasis visual sudah mulai digiring ke basis internet dan mobile, dengan bermunculan webservice dan berbasis net dan a mobile flatform.

- **Konsep pertama** yang dicermati adalah konsepsi Microsoft dimana dengan Visual Net akan menyediakan berbagai bahasa pemrograman seperti VB Net , VC++ Net, ASP NET yang di compile dengan berbagai bahasa akan tetapi berjalan pada satu sistem operasi yakni windows. (Compile any program run one system)
- **Konsepsi Kedua**, Merupakan konsep yang terbalik dari konsep pertama yakni apa yang ditawarkan Sun Microsystem melalui produknya Java, J2ME, JDK, yakni dicompile dengan satu bahasa pemrograman (java) dan berjalan dibanyak sistem operasi. (Compile one program running any system)

Selain itu periode ini juga merupakan jamannya CMS (Content Manajemen System), lompatan pengembangan PHP Script begitu cepat, dimana untuk membuat website atau portal telah tersedia banyak template, Banyak modul-modul yang siap pakai sehingga programmer atau webmaster tidak perlu lagi mempelajari semua script html dan bahasanya, tinggal merangkai

modul yang tersedia sehingga dalam beberapa hari saja sebuah web sudah dapat dibuat. Apa yang ditawarkan Mambo, PhkNuke dan Jomla saat ini sangat memudahkan para desainer web.
Beberapa bahasa pemrograman yang berkembang dalam periode ini termasuk

- Tcl/Tk,
- O'Caml,
- Ruby,
- Phyton 3.1,
- Java 6 JDK, JED, Java Beans, J2ME
- Microsoft Visual Net (VB Net, C++ Net, ASP NET) 2008
- Java Scrip Template oleh Mambo, PhpNuke, Jomla

Sejarah pandangan pertama tiga generasi

Istilah "generasi pertama" dan "generasi kedua" bahasa pemrograman tidak digunakan sebelum coining dari istilah "generasi ketiga." Bahkan, tak satupun dari ketiga istilah yang disebutkan dalam kompendium awal bahasa pemrograman.

Pengenalan generasi ketiga dari teknologi komputer bertepatan dengan penciptaan generasi baru bahasa pemrograman. Pemasaran untuk pergeseran generasi dalam mesin tidak berkorelasi dengan beberapa perubahan penting dalam apa yang disebut tingkat tinggi bahasa pemrograman, dibahas di bawah, memberikan konten teknis untuk perbedaan kedua / ketiga-generasi antara bahasa pemrograman tingkat tinggi juga, dan mengubah nama refleks bahasa assembler sebagai generasi pertama.

Generasi Kedua

Artikel utama: Generasi kedua bahasa pemrograman

Bahasa pemrograman generasi kedua, awalnya hanya disebut bahasa pemrograman tingkat tinggi , diciptakan untuk menyederhanakan beban pemrograman dengan membuat ekspresi yang lebih seperti modus normal ekspresi pemikiran yang digunakan oleh programmer. Mereka diperkenalkan pada akhir 1950-an, dengan FORTRAN mencerminkan kebutuhan programmer ilmiah, ALGOL mencerminkan upaya untuk menghasilkan tampilan standar Eropa / Amerika.

Masalah yang paling penting yang dihadapi oleh para pengembang dari tingkat kedua bahasa adalah pelanggan meyakinkan bahwa kode yang dihasilkan oleh para penyusun dilakukan

dengan baik-cukup untuk membenarkan meninggalkan pemrograman assembler. Dalam pandangan skeptisisme luas tentang kemungkinan memproduksi program efisien dengan sistem pemrograman otomatis dan fakta bahwa inefisiensi tidak bisa lagi disembunyikan, para pengembang yakin bahwa jenis sistem yang mereka ada dalam pikiran akan banyak digunakan hanya jika mereka bisa menunjukkan bahwa hal itu akan menghasilkan program hampir seefisien yang tangan kode dan melakukannya pada hampir setiap pekerjaan. Compiler FORTRAN dipandang sebagai tour de force-dalam produksi berkualitas tinggi kode, bahkan termasuk "… a Monte Carlo simulasi pelaksanaannya … sehingga dapat meminimalkan transfer barang antara toko dan indeks register."

Generasi ketiga

Artikel utama: generasi ketiga bahasa pemrograman

Pengenalan generasi ketiga dari teknologi komputer bertepatan dengan penciptaan generasi baru bahasa pemrograman. [1] Fitur penting dari generasi ketiga bahasa adalah hardware-kemerdekaan mereka, ekspresi yaitu algoritma dengan cara yang independen dari karakteristik mesin yang algoritma akan berjalan.

Beberapa atau semua dari sejumlah perkembangan lain yang terjadi pada saat yang sama dimasukkan dalam 3GLs.

Interpretasi diperkenalkan. 3GLs Beberapa disusun , proses analog dengan penciptaan dieksekusi kode mesin lengkap dari kode assembly, perbedaan adalah bahwa dalam bahasa tingkat tinggi tidak ada lagi hubungan, satu-ke-satu, atau bahkan linier antara petunjuk source code kode mesin instruksi dan. Compiler dapat menargetkan hardware yang berbeda dengan memproduksi terjemahan yang berbeda dari perintah kode sumber yang sama.

Penafsir, di sisi lain, pada dasarnya menjalankan instruksi kode sumber itu sendiri – jika seseorang menemukan sebuah "add" instruksi, ia melakukan tambahan itu sendiri, daripada keluaran instruksi tambahan akan dieksekusi kemudian. Mesin-kemerdekaan dicapai dengan memiliki interpreter yang berbeda dalam kode mesin dari platform yang ditargetkan, yaitu penafsir itu sendiri umumnya harus dikompilasi. Interpretasi bukanlah "muka" linear, tetapi model alternatif untuk kompilasi, yang terus ada di samping, dan lainnya, baru-baru ini dikembangkan, hibrida. Lisp adalah bahasa interprepreted awal.

Para 3GLs awal, seperti Fortran dan Cobol , adalah spaghetti kode , yaitu mereka memiliki gaya yang sama dari aliran kontrol sebagai assembler dan kode mesin , membuat penggunaan

berat dari goto pernyataan. pemrograman terstruktur [2] diperkenalkan model di mana program itu dilihat sebagai hirarki blok bersarang daripada daftar linear instruksi. Misalnya, programmer terstruktur adalah untuk memahami sebuah loop sebagai blok kode yang diulang, bukan perintah begitu banyak diikuti oleh melompat mundur atau goto. Pemrograman terstruktur kurang tentang *kekuasaan* – dalam arti satu tingkat yang lebih tinggi commannd ekspansi ke berbagai tingkat rendah yang – dari *keselamatan.* Pemrogram berikut ini jauh kurang rentan untuk membuat kesalahan. Pembagian kode ke blok, subrutin dan modul lainnya dengan antarmuka jelas-didefinisikan juga memiliki manfaat produktivitas dalam memungkinkan programmer banyak untuk bekerja pada satu proyek. Setelah diperkenalkan (dalam ALGOL bahasa), pemrograman terstruktur dimasukkan ke dalam hampir semua bahasa, dan dipasang dengan bahasa yang awalnya tidak memilikinya, seperti Fortran , dll

Struktur Blok juga dikaitkan dengan depresiasi variabel global , sumber kesalahan yang sama dengan goto . Sebaliknya, bahasa terstruktur diperkenalkan scoping leksikal dan manajemen penyimpanan otomatis dengan stack.

Fitur lain yang tingkat tinggi adalah pengembangan dari sistem tipe yang melampaui jenis data kode mesin yang mendasari, seperti string , array dan catatan .

Dimana 3GLs awal adalah tujuan khusus, (misalnya ilmu pengetahuan atau perdagangan) upaya telah dilakukan untuk menciptakan tujuan umum bahasa, seperti C dan Pascal . Sementara ini menikmati sukses besar, domain bahasa tertentu tidak menghilang.

Timeline Computer Aided Software Engineering (CASE)

Computer-aided software engineering (CASE) adalah domain perangkat lunak yang digunakan untuk merancang dan mengimplementasikan aplikasi. Alat KASUS serupa dan sebagian terinspirasi oleh alat bantu desain komputer (CAD) yang digunakan untuk merancang produk perangkat keras. Alat KASUS digunakan untuk mengembangkan perangkat lunak berkualitas tinggi, bebas cacat, dan mudah perawatan. Perangkat lunak KASUS sering dikaitkan dengan metode pengembangan sistem informasi bersama dengan alat otomatis yang dapat digunakan dalam proses pengembangan perangkat lunak.

Proyek Sistem Desain dan Optimasi Sistem Informasi (ISDOS), dimulai pada tahun 1968 di University of Michigan, memprakarsai banyak minat terhadap keseluruhan konsep penggunaan

sistem komputer untuk membantu analis dalam proses yang sangat sulit dalam menganalisis persyaratan dan pengembangan sistem. Beberapa makalah oleh Daniel Teichroew memecat seluruh generasi peminat dengan potensi pengembangan sistem otomatis. Pernyataan Soal-nya Alat Penguji Persamaan / Masalah Masalah Analyzer (PSL / PSA) adalah alat KASUS meskipun telah mendahului istilah tersebut. [3]

Thread utama lainnya muncul sebagai perpanjangan logis ke kamus data database. Dengan memperluas jangkauan metadata yang dimiliki, atribut aplikasi dapat disimpan dalam kamus dan digunakan saat runtime. "Kamus aktif" ini menjadi pendahulu kemampuan rekayasa model yang lebih modern. Namun, kamus aktif tidak menyediakan representasi grafis dari salah satu metadata. Itu adalah penghubung konsep kamus yang memegang metadata analis, yang berasal dari penggunaan seperangkat teknik terpadu, bersamaan dengan penggambaran grafis dari data semacam itu yang memunculkan versi KASUS sebelumnya.

Istilah ini awalnya diciptakan oleh perusahaan perangkat lunak Nastec Corporation di Southfield, Michigan pada tahun 1982 dengan grafis terintegrasi dan editor teks asli GraphiText, yang juga merupakan sistem berbasis komputer pertama yang menggunakan hyperlink untuk mengirim teks referensi silang dalam dokumen-bahkan awal pendahulu dari link halaman web hari ini. Produk pengganti GraphiText, DesignAid, adalah alat berbasis mikroprosesor pertama yang secara logis dan semantis mengevaluasi diagram perancangan perangkat lunak dan sistem dan membuat kamus data.

Di bawah arahan Albert F. Case, Jr. wakil presiden untuk manajemen dan konsultasi produk, dan Vaughn Frick, direktur manajemen produk, suite produk DesignAid diperluas untuk mendukung analisis berbagai metodologi analisis dan perancangan terstruktur, termasuk yang dari Ed Yourdon dan Tom DeMarco, Chris Gane & Trish Sarson, Ward-Mellor (real-time) SA / SD dan Warnier-Orr (data driven).

Peserta berikutnya ke pasar adalah Excelerator dari Index Technology di Cambridge, Mass. Sementara DesignAid berlari di Convergent Technologies dan kemudian mengembangkan mikrokomputer Burroughs Ngen, Index meluncurkan Excelerator pada platform IBM PC / AT. Sementara, pada saat peluncuran, dan selama beberapa tahun, platform IBM tidak mendukung jaringan atau basis data terpusat seperti halnya mesin Konvergensi atau Burroughs, daya tarik IBM kuat, dan Excelerator menjadi terkenal. Hot on the tumit dari Excelerator adalah ruam penawaran

dari perusahaan seperti Knowledgeware (James Martin, Fran Tarkenton dan Don Addington), alat bantu instrumen IEF dan Andersen Consulting dari Instrument Instrument Jepang (DESIGN / 1, INSTALL / 1, FCP).

Alat KASUS berada pada puncaknya di awal tahun 1990an. Pada saat IBM mengajukan AD / Cycle, yang merupakan aliansi vendor perangkat lunak yang berpusat pada repositori IBM's Software menggunakan IBM DB2 di mainframe dan OS / 2.

Dengan menurunnya mainframe, alat AD / Cycle dan Big CASE mati, membuka pasar untuk alat CASE mainstream saat ini. Banyak pemimpin pasar KASUS pada awal 1990an akhirnya dibeli oleh Computer Associates, termasuk IEW, IEF, ADW, Cayenne, dan Learmonth & Burchett Management Systems (LBMS). Kecenderungan lain yang menyebabkan evolusi alat KASUS adalah munculnya metode dan alat berorientasi objek. Sebagian besar vendor alat menambahkan beberapa dukungan untuk metode dan alat berorientasi objek. Selain itu produk baru muncul yang didesain dari bawah ke atas untuk menunjang pendekatan object-oriented. Andersen mengembangkan proyek Eagle sebagai alternatif Foundation.

Beberapa pemimpin pemikiran dalam pengembangan berorientasi objek masing-masing mengembangkan metodologi dan perangkat CASE mereka sendiri: Jacobsen, Rumbaugh, Booch, dll. Akhirnya, perangkat dan perangkat beragam ini digabungkan melalui standar yang dipimpin oleh Object Management Group (OMG). The OMG's Unified Modeling Language (UML) saat ini diterima secara luas sebagai standar industri untuk pemodelan berorientasi objek.

Sumber Referensi :

- ✓ https://id.wikipedia.org/wiki/Perangkat_lunakeloper
- ✓ https://infomaticexpo.wordpress.com/tugas-tugas-kuliah-dan-info-menarik/sejarah-dan-perkembangan-bahasa-pemograman/
- ✓ http://elib.unikom.ac.id/download.php?id=83957
- ✓ http://sulistriyani.blogspot.co.id/2010/05/proses-software.html