Etapy vývoja informačných technológií

1. Etapa klasickej automatizácie
2. Etapa typových aplikačných programových vybavení (typových APV)
3. Etapa interaktívnych systémov
4. Etapa využitia osobných počítačov
5. Etapa distribuovaných informačných systémov a globálnych počítačových sietí

Etapa klasickej automatizácie

- 60.-te a začiatok 70.-tych rokov

Charakteristické črty:

- IS sa vytvárali pre konkrétne objekty riadenia
- Tvorba IS sa uskutočňovala bez akejkoľvek automatizácie
- Čas návrhu a realizácie IS bol závislý od rozsahu projektu a pohyboval sa medzi 2-5 rokmi
- Informačná základňa IS bola usporiadaná ako množina súborov so sekvenčnou indexovo-sekvenčnou alebo priamou organizáciou
- Programovanie v strojovom jazyku a algoritmických programovacích jazykoch
- Technickými médiami boli dierne štítky a pásky, magnetické pásky, disky a iné
- Zložitá technológia spracovania dát vyžadujúca prísnu organizáciu (zhromaždenie a prenos dát do stredísk, prenos na technické média, kontrola a korekcia dát, programová kontrola vstupných údajov, vytvorenie a aktualizácia súborov dát, vyhotovenie kópií súborov, spracovanie dát PC, vyhotovenie výstupných zostáv alebo súborov dát, kontrola výstupných zostáv, odstránenie chýb a opakovanie výpočtov, vytlačenie zostáv)

Nedostatky:

- Zložitosť projektovania, programovania a spracovania dát bránili užívateľom na priamej účasti na ľubovoľnej etape ŽC IS
- Dlhodobé termíny tvorby IS
- Ručná práca a veľká prácnosť projektanta a programátora
- Kvalita IS závislá od odbornej kvalifikácie a tvorivej schopnosti projektanta a programátora, ktorí sa museli zúčastňovať implementácie aj prevádzky IS
- Rozdelenie informačnej základne na veľké množstvo samostatných súborov

Etapa typových aplikačných programových vybavení (typových APV)

- Rok 1969 – firma IBM zaviedla osobitný predaj SW a HW

Charakteristické črty:

- Na tvorbu IS sa používajú typové APV, vlastné aplikácie alebo kombinácie
- Typové APV automatizovali informačné úlohy organizácií a podnikov
- Popri súborovej sa používa aj databázová organizácia dát
- Technológia spracovania dát je zachovaná
- Režim spracovania dát je in-line a off-line

Nedostatky:

- V technológii prípravy dát spočívajúce z veľkej prácnosti a neefektívnosti
- V technológii spracovania dát
- Nekompatibilita jednotlivých typových APV a náročnosť prispôsobenia objektom
- Nemožnosť integrácie IS

- Nemožnosť tvorby kompletných IS, lebo typové APV automatizovali len niektoré ekonomické úlohy (napr. marketing, účtovníctvo, zásobovanie)

Etapa interaktívnych systémov

- Koniec 70-tych rokov a 80-te roky

Charakteristické črty:

- Skrátenie technologického cyklu spracovania dát
- Vstupné údaje sa bezprostredne zavádzajú od zdrojov informácií
- Skrátenie doby návrhu IS
- IS sa priblížili k systémom riadenia
- Používatelia vystačia s minimálnymi poznatkami z oblasti automatizovaného spracovania dát
- Interaktívny režim spracovania dát zväčšil škálu IS rozširujúcich možnosti OS

Nedostatky:

- Zvýšenie odborných požiadaviek na projektantov a programátorov dodatočnou aplikáciou určitých systémových SP a SAPP
- Pohodlnejšie dávkové spracovanie pre užívateľov
- Potreba značných znalostí používateľa v oblasti IT

Etapa využitia osobných počítačov

- 80.-te roky

Charakteristické črty:

- Prednostné využitie osobných PC ako samostatných pracovných miest alebo v lokálnych sieťach
- Prednostné využitie SP pre PC
- Vysoká kompatibilita HW a SW rôznych výrobcov
- Prednostné využitie relačných báz dát
- Filozofia automatizácie pracovného miesta
- Vznik CASE systémov

Nedostatky:

- Nevhodne aplikovaná filozofia automatizácie pracovného miesta zredukovala ŽC IS na fázy programovanie a prevádzka
- Vytvorenie neefektívnych IS, pretože sa úlohy integrovali až po automatizácii
- Ťažisko tvorby IS sa presunulo na programovanie

Etapa distribuovaných informačných systémov a globálnych počítačových sietí

Charakteristické črty:

- Masové využitie internetu
- Využitie databázových systémov umožňujúcich DSD
- Využitie SAPP na komplexnú automatizáciu
- Využitie objektovo orientovaných jazykov vyššej generácie
- Vývoj podľa štruktúrovaných alebo objektovo-orientovaných metodológií

Porovnanie etáp

Rysy	1	2	3	4	5
Riadenie spracovania dát	riadiacimi prog., neskôr OS	OS	OS s rozšírenými možnosťami a riadiacim prog. SP		sieťové OS
Prevládajúci HW	sálové počítače			PC	servery, PC
Prevládajúce metódy organizácie dát	súbory (sekvenčná, index-sek, priama organizácia)	súbory, BD (hierarchické, sieťové)	súbory, BD (hierarchické, sieťové, relačné)	relačné BD	relačné BD, dátové sklady
Režimy prenosu dát	In-line	In-line, off-line	On-line		
Režim spracovania dát	dávkový		interaktívny		
Účasť používateľa na príprave a spracovaní dát	malá		väčšia	veľmi veľká	
Prácnosť prípravy dát	veľmi veľká		malá		
Potrebnosť organizácie prípravy a spracovania dát	áno		nie		

Rysy	1	2	3	4	5
Prácnosť technológií	veľmi veľká		veľká	malá	
Prácnosť údržby informačnej základne IS	veľmi veľká		veľká	malá	
Potrebný operátorský personál	početný		malý		
Potrebný personál na prevádzku IS	početný		malý		
Prevládajúce metódy tvorby programov	assembler, cobol, fortran	typové APV a algor. p. j.	typové APV, SAPP, programovacie jazyky		typové APV, SAPP, OO prog. jazyky

Rysy	1	2	3	4	5
Doba projektovania	2-4 roky	1-2 roky	okolo roka	mesiace	
Doba programovania	2-3 roky	2-3 roky, typovo	1-2 roky	mesiace	
Dokumentovanie projektov a programov	ručné	typové, ručné	ručné, automatizované		
Možnosť automatizácie pri analýze, návrhu, programovania	nie	nie (TAPV áno)	Čiastočne áno	áno	
Prácnosť pri automatizácii programovania			veľká	malá	

SAPP

- Sú softvérové produkty, ktoré automatizujú jednu alebo viac rutinných činností životného cyklu IS a SP.
- Sú nástroje, ktoré projektanti a programátori môžu použiť pri analýze, návrhu, implementácii a prevádzke SP a uľahčenie, skvalitnenie a zrýchlenie práce

Delenie SAPP

- podľa využitia v etapách vývoja
- podľa funkčného určenia
- podľa podpory fáz ŽC IS
- podľa prostredia práce vyvíjaného IS

SAPP podľa využitia v etapách vývoja

- V etape typových APV a interaktívnych systémoch
  - Generátory vstupov
  - Generátory výstupov
  - Na automat. prác s informačnou základňou
  - SAPP na dokumentovanie IS
  - Generátory kódov
  - Komplexné SAPP
- V etape využitia osobných počítačov
  - SAPP z etáp 2. a 3.
  - Tabuľkové procesory (napr. Lotus, Excel, práca s tabuľkami, kalkulácie, generovanie výstupov – textové, grafické, súbory)
  - Kartotéky (tvorba, aktualizácia, generovanie výstupov - textové, grafické, zoradenie, kalkulácie)

- Textové editory (jazykovo orientované, používané na dokumentovanie AIS, poskytovali grafické výstupy)
- DBS na automatizáciu pracovného miesta
- CASE

Generátory vstupov

- Pracujúce v dávkovom režime
- Pracujúce v interaktívnom režime

Popis:
Automatizujú transfer údajov z dokladov do súborov, resp. bázy dát
Generujú programy na vytváranie súborov dát, resp. BD a na ich aktualizáciu zo vstupných dát
Generujú osobitý program pre každý vstup
Generátory vstupov umožňujú definovať:

- Kontrolu vstupných údajov
- Metódu organizácie súborov, resp. BD
- Technické médiá
- Štruktúru vstupov
- Štruktúru súborov, resp. BD

Program na záznam dát do vstupných súborov môže byť:

- Originálny – pre každý vstupný doklad
- Univerzálny – je potrebné zadať vstupné a výstupné parametre

Generátory výstupov
Na základe parametrov a popisu zadaných v riadiacom jazyku generujú programy, ktoré:

- Čítajú údaje zo súborov dát a BD
- Vykonávajú určité kalkulácie a operácie s týmito údajmi
- Vytvárajú výstupné súbory dát, resp. umožňujú tlačiť výstupné zostavy

Parametre generovania sa týkajú:

- titulnej časti výstupu
- štruktúry výstupu
- miesta načítania údajov
- požadovaných kalkulácií

Generované programy sú v algoritmickom jazyku (napr. assembler) a môžu byť neskôr upravené
Podľa príkazov riadiaceho programu sa vykonáva kompilácia aj linkovanie

Na automat. prác s informačnou základňou
Negenerujú programy, ale vykonávajú zadané funkcie
Automatizujú:

- Programovanie klasických technologických operácií so súbormi dát (premenovanie, kopírovanie, rozdelenie, zjednotenie, reorganizácia, zmazanie, aktualizácia súborov dát)
  Tieto systémy sa stali súčasťou OS
- Vykonanie aritmetických a logických operácií s údajmi BD a súbormi, zmeny štruktúry informačnej základne a zmeny špecifikácií údajov

- Rešerš v súboroch alebo BD po zadanom kľúči a display, resp. tlačenie
- Vytváranie a využívanie slovníkov dát a metabáz dát (tvorba, vyhľadávanie, zaradenie popisov dát do programu a linkovanie programov s DB)

Metódy realizácie systémov:

- Parametricko-interpretačná metóda
  - Efektívna pri úlohách s malým počtom technologických operácií a úlohách, ktoré v značnej miere využívajú systémové programy
  - Realizuje funkcie OS a funkcie vykonávané výkonnými modulmi (napr. aritmetické, logické, čítanie, zoradenie)
  - Príkazu zodpovedá osobitný výkonný modul, ktorého práca je podmienená zadanými parametrami
  - Určená pre SP s jednoduchou štruktúrou, a pre obmedzený počet technologických operácií
- Metóda generovania technológií
  - SAPP založené na metóde sú SP so zložitou štruktúrou – obsahujú interpretátor = translátor, riadiaci modul = generátor technológií, výkonné moduly
  - Každý výkonný modul vykonáva komplexnú technologickú operáciu ako samostatný program, ktorého funkcie možno regulovať parametrami

SAPP na dokumentovanie IS

- Používali sa len na dokumentovanie niektorých častí projektov AIS, najmä programových algoritmov
- Generátor vývojových diagramov (GVD) generoval z kódu programu vývojové diagramy, príkazy riadiaceho jazyka GVD sa zapisujú ako komentáre k programovým riadkom (assembler, cobol, PL/1, fortran)
- Nevýhoda: diagramy vznikajú z programu a nie naopak

Generátory kódov

- Automaticky zostavujú programy pre spracovanie dát podľa príkazov riadiaceho programu
- Vstup: procedurálny alebo neprocedurálny v riadiacom jazyku
- Výstup: program v strojovom kóde, resp. fáza

Riadiaci jazyk je orientovaný na:

- Filozofiu algoritmických jazykov
- Štandardné, štruktúrované, modulárne programovanie a ich kombináciu (štandardné a modulárne programovanie sú základom pre objektové programovanie a programovanie riadené udalosťami)

Štandardné programovanie

- Založené na idei rozpracovania typových štruktúr (programových častí) nazývaných bloky
- Každý blok sa môže skladať z častí vykonávajúcich sa zhora-nadol a majú jeden vstup a jeden výstup
- Ide o proces definovania hotových blokov a ich poradie v programoch

Štruktúrované programovanie

- Založené na metód Jacksona, t.j. idei, že všetky príkazy programu sa dajú znázorniť tromi schémami – sekvenciou, selekciou a iteráciou

Modulárne programovanie

- Základná idea je, že každý program možno stavať ako SP s jednoduchou štruktúrou
- Moduly sú viazané na vykonávané funkcie objektu a nie na možné operácie spracovania dát a majú jeden vstup a jeden výstup

DBS na automat. pracovného miesta

- d`Base, FoxPro
- automatizuje: Vytvorenie štruktúr BD, záznam údajov do BD, kontrolu vstupných údajov, tvorbu súborov výstupných správ, tvorbu ľubovoľných výstupov a aktualizáciu BD
- Riadiaci jazyk je určený na prácu s BD a tvorbu výstupov a je jednoduchší ako algoritmické jazyky, programovanie je v interaktívnom režime, ale na tvorbu výstupov stále zložité

SAPP podľa funkčného určenia

- Na podporu vývoja IS
  - Na tvorbu diagramov a na dokumentovanie
  - Na návrh vstupov a výstupov
  - Na generovanie kódov a testovanie programov
  - Na normalizáciu BD a tvorbu LMD
  - Na generovanie fyzického modelu dát
- Generátory vstupov
- Generátory výstupov
- Na automat. prác s informačnou základňou
- Iné

SAPP podľa podpory fáz ŽC IS

- Na podporu plánovania a riadenia návrhu IS
- Na automatizáciu prác vo fáze analýzy
- Na automatizáciu projektantských prác vo fáze návrhu IS
- Generátory kódov a SAPP na automatické testovanie programov
- Na priame vykonávanie technologických operácií spracovania dát

SAPP podľa prostredia práce vyvíjaného IS

- CASE (SDW, Rational Rose)
- DBS na automatizáciu pracovného miesta (d`Base, Fox Pro)
- Vývojové nadstavby DBS (Oracle DS)
- Iné SAPP (poskytujú aj aplikácie nespojené so žiadnou fázou ŽC IS)

CASE - Computer Aided Software Engineering

- Objavili sa koncom 80.-tych a začiatkom 90.-tych rokov
- Automatizujú úlohy jednej alebo viacerých fáz životného cyklu IS a SP a podporujú určité metodológie a metódy tvorby IS a SP
- Zjednodušujú a urýchľujú proces tvorby, znižujú náklady na tvorbu, formalizujú a štandardizujú proces tvorby a zvyšujú kvalitu navrhovaných IS a SP

CASE automatizujú:

- Tvorbu diagramov podľa rôznych metód
- Návrh vstupných a výstupných obrazoviek a simuláciu dialógu s nimi
- Návrh logického a fyzického modelu BD
- Generovanie programov pre navrhnuté vstupy, výstupy a BD
- Dokumentovanie jednotlivých fáz ŽC SP
- Testovanie programov
- Reverzné inžinierstvo
- Riadenie IS

Charakteristické rysy CASE (spoločné rysy):

- Centrálny riadiaci modul (menu, nastavenia)
- Modulová štruktúra
- Pracovná plocha a grafické prostriedky
- Technológia okien (typy okien)
- Technológia WYSIWYG – „What You See Is What You Get“
- Repozitory – metabáza dát
- Navigátor – nástroj na správu repository
- Hierarchizácia diagramov
- Technológia Export/Import
- Automatická kontrola (syntaktická, sémantická, úplnosti, konzistencie)
- Generátory dokumentácie
- Help systém
- Podpora metodológií, metód a diagramových techník
- Podpora tímovej spolupráce
- Editor a generátor V/V formulárov
- Objektové modelovanie
- Round-Trip Engineering - Generátory kódov, Reverzné inžinierstvo, Synchronizácia
- Automatické testovanie generovaných programov
- Modelovanie, generovanie a reverzné inžinierstvo dátového modelu
- Modelovanie ekonomických procesov
- Podpora a presadzovanie vývojových štandardov
- Znovupoužiteľnosť návrhov a vytvorených komponentov
- Porovnanie modelov a zistenie odlišností
- Správa požiadaviek používateľov

Centrálny riadiaci modul predstavuje modul, ktorý zabezpečuje plynulý prechod medzi jednotlivými modulmi a diagramami zastrešenými spoločným rozhraním. Komunikáciu a prezentáciu riadiaceho modulu plní väčšinou systém menu a jednotlivé ponuky a nastavenia v systéme.

Modulová štruktúra znamená, že sa systémy skladajú so samostatných a navzájom prepojených a nadväzujúcich modulov. Každý z modulov plní určitú úlohu alebo zabezpečuje nejakú funkcionalitu systému.

Metabáza dát (Repozitory) predstavuje spoločný priestor, databanku, kde sa uchovávajú informácie o všetkých diagramoch, ich prvkoch a vzťahoch, ktoré boli vytvorené v rámci modulov systémov CASE. Repozitory nedovolí zadávať redundantne nové objekty, ponúka už vytvorené, aby sme predišli chybe pri zadávaní už existujúceho názvu a podobne a je automaticky aktualizovaná pri práci s ľubovoľným modulom. Repozitory je priestor, ktorý rieši dátové prepojenie a nadväznosť jednotlivých modulov.
Nástroj na správu repozitory (navigátor, prehliadač - browser) je nástroj, ktorý poskytuje globálny pohľad na celý projekt v podobe stromovej štruktúry.

Grafické prostredie a prostriedky v sebe zahrňuje pracovnú plochu systému a grafickú podporu. Na pracovnej ploche sa zostrojujú a editujú diagramy podľa daných notácií a obrazcov, pretože sa jednoduchšie číta a chápe obrázok ako písané slovo. Do diagramov sa vkladajú prvky, ktoré sa vyberajú z ponuky preddefinovaných obrazcov v Toolbar alebo Toolbox. To znamená, že systém ponúka pomoc pri kreslení, kopírovaní, presúvaní, mazaní, usporiadaní prvkov a aj pri zadávaní potrebných parametrov.
Technológia WYSIWYG - „What You See Is What You Get“ čiastočne súvisí s grafickým prostredím a prostriedkami. Je to princíp verného prenosu vizuálnej informácie, resp. informácie modelovanej na počítači do reality tak, že zodpovedá presne modelovanému obrazu s čo najmenším, resp. nebadateľným skreslením.
Hierarchizácia diagramov umožňuje aj na dvojrozmernej ploche udržiavať globálny pohľad na modelovaný systém. Diagramy sa väčšinou zostrojujú zhora nadol, čiže sa hierarchicky dekomponujú z vyššej úrovne na diagramy nižšej úrovne až po elementárne prvky.

Technológia okien znamená, že systémy umožňujú prácu vo viacerých typoch okien. Okná možno rozdeliť na: diagramové, ktoré umožňujú kreslenie diagramov, zadávacie, ktoré umožňujú zadať stručný popis ľubovoľného prvku, výberové, ktoré umožňujú výber charakteristík ľubovoľného prvku, textové, ktoré umožňujú pridať komentár ľubovoľnému prvku, pomocné (help), ktoré poskytujú pomoc k zvolenému prvku alebo celému systému, dokumentačné, ktoré umožňujú vytvoriť dokumentáciu alebo zobraziť a editovať generovanú dokumentáciu, okno vlastností, ktoré umožňuje zadať hodnoty pre preddefinované vlastnosti ľubovoľného prvku a iné.

Automatická kontrola konzistencie, sémantiky, syntaxe, úplnosti. Kontrola konzistencie znamená kontrolu zosúladenosti jednotlivých úrovní diagramu. Kontrola sémantiky predstavuje logickú kontrolu modelov voči pravidlám tvorby diagramov, zásadám normalizovaného modelu a podobne. Kontrola syntaxe predstavuje fyzickú kontrolu modelu voči syntaxi cieľového systému riadenia bázy dát pri generovaní databázy alebo cieľového jazyka pri generovaní kódu. Kontrola úplnosti udáva, či sú v modely zahrnuté všetky prvky z modelu, na ktorý nadväzuje. Okrem spomenutých typov kontrol obsahujú niektoré CASE detekovanie a lokalizáciu (budúcej) run-time chyby, vrátane chýb typu zlej správy/alokácie pamäte (chyby neinicializovanej pamäte, neuvoľnenie alokovanej pamäte).

Export / Import systém, ktorý zabezpečuje export a import modelov alebo celých projektov medzi vývojármi pracujúcimi v odlišných prostrediach, alebo na prenos do nového OS, alebo novej verzie vhodnejšieho systému CASE a podobne.

Generátor dokumentácie, ktorý umožňuje vygenerovať komplexnú dokumentáciu celého projektu, vrátane diagramov, textových popisov diagramov a ich prvkov a komentárov vývojárov. Väčšinou je možné takto vygenerovanú dokumentáciu vyexportovať napríklad do HTML formátu na webovú stránku, aby bola dokumentácia voľne prístupná na internete. Navyše sa dá nastaviť aj hĺbka obsiahnutých informácií, takže je možné získať osobitne reporty pre manažérov, zákazníkov, ostatných členov vývojového tímu, administrátora databázového servera a podobne. Automaticky vytvorená dokumentácia má jednotný vzhľad a prispieva ku konzistencii tímu.

Systém pomoci (Help systém), ktorý obsahuje návod na ovládanie jednotlivých funkcionalít systému a poskytuje aj návod ako vytvoriť diagram, vysvetľuje významovo a obsahovo jednotlivé prvky a podobne.

Podpora metodológií, metód a diagramových techník (notácií). Príslušný systém CASE by mal podporovať a zabezpečovať dodržiavanie konkrétnych štruktúrovaných metodológií, metód a diagramových techník, ak ide o štruktúrovaný systém CASE, alebo objektovo orientovaných metodológií, metód a diagramových techník, ak ide o objektovo orientovaný systém CASE. Nejedná sa teda o ľubovoľné tvorenie modelov, keď sa práve hodia, ale o postupnosť na seba nadväzujúcich činností riadených konkrétne implementovaným metodologickým postupom.

Podpora tímovej spolupráce. Systém CASE musí dovoliť zostaviť tím, ktorý je zložený z jednotlivých členov, ktorí majú práva podľa príslušnosti do typu skupiny (napr. administrátor projektu, analytik, návrhár, programátor, tester). S podporou tímovej spolupráce súvisí aj tvorba verzií projektu, uzamykanie (locking) diagramov alebo podprojektov, aby sa predišlo kolíziám, zdieľanie rozpracovaných fragmentov, a aj spájanie (merge) čiastkových výsledkov do konečného tvaru.

Cyklický vývoj aplikácie (Round-Trip Engineering - RTE) je opakujúci sa proces, ktorý by mal zaisťovať väzbu medzi vygenerovaným kódom, návrhom a analytickým modelom systému, čiže synchronizuje analytické a návrhové modely s konkrétnou implementáciou. Ak analytik alebo návrhár urobí zmenu v modely, prenesie sa táto zmena automaticky do kódu aplikácie a do modelu nižších vývojových vrstiev. Tejto fáze sa hovorí forward engineering (FE, dopredné). Naopak, ak programátor urobí zmenu v kóde, tak sa táto zmena prejaví v príslušnom diagrame. Tejto fáze sa hovorí reverse engineering (RE, spätné). O udržanie RTE procesu sa stará tzv. synchronizátor kódu, ktorý umožňuje synchronizovať programový kód s modelom bez toho, aby sa musel realizovať celý RE alebo FE proces od začiatku. FE je jednosmerný proces a môžeme ho nazvať aj generovanie kódu, čo predstavuje generovanie zdrojového kódu v príslušnom programovacom jazyku z diagramov. RE je tiež jednosmerný proces, ktorý znamená spätné získanie modelu (diagramu) zo zdrojového kódu v nejakom programovacom jazyku. Podľa Eliota Chikofského a Jamese Crossa [KAD02] je reverzné inžinierstvo proces analyzovania systému za účelom identifikovania jeho súčastí (komponentov) a vzťahov medzi nimi. Výsledkom tejto analýzy je reprezentácia skúmanej aplikácie vytvorená za použitia určitého stupňa abstrakcie. RE je teda časť údržby a podpory softvéru, ktorá pomáha pochopiť systém a následne uľahčiť realizáciu prípadných zmien. Synchronizácia je proces obojsmerný a znamená zosúladenie modelu a kódu v určitom bode rovnováhy.

Modelovanie, generovanie a reverzné inžinierstvo dátového modelu. Pri modelovaní ide o tvorbu konceptuálneho, logického a fyzického dátového modelu. Konceptuálny dátový model je množina abstraktných dátových štruktúr (entít) a vzťahov medzi nimi. Je platformovo nezávislý a neobsahuje implementačné detaily. Logický dátový model vzniká z konceptuálneho tak, že z entít vznikajú tabuľky, z relácie 1:N cudzí kľúč a relácie M:N relačná tabuľka so zloženým primárnym kľúčom. Fyzický model vychádza z logického, ale je už spojený s predpokladanou cieľovou platformou, resp. databázovým serverom. Kvalitné nástroje podporujú desiatky DB serverov. Vo fyzickom modely sa analytik môže zaoberať špecifikami, zvláštnosťami a odlišnosťami jednotlivých platforiem. Samozrejmosťou nástroja je jednoduchý a priamočiari prechod z konceptuálneho modelu na fyzický. Nástroj sám vykoná preklad domén, relácií M:N, kľúčov a ďalších prvkov pre zvolenú cieľovú platformu. Konečný logický (štruktúrovaný prístup) alebo fyzický model (objektovo orientovaný prístup) potom možno vygenerovať. Spomenutý proces nazývame generovanie databázy alebo DDL súboru, pri ktorom ide o generovanie DDL súboru na pevný disk alebo databázy priamo na server z diagramu logického modelu dát. Pri generovaní je možné zvoliť, čo všetko sa má vygenerovať, či všetky tabuľky, alebo len niektoré, či aj indexy alebo pohľady a podobne. Možný je aj spätný chod od fyzického ku konceptuálnemu modelu nazývaný reverzné inžinierstvo, ale najčastejšie sa jedná len o spätné získanie logického modelu dát z existujúcej databázy alebo DDL súboru.

Objektové modelovanie. Úplný objektový prístup umožňuje definovať a modelovať používateľské scenáre a prípady použitia, rovnako ako hraničné, dátové, riadiace a doménové objekty. V rámci objektového modelovania sa nezaoberáme len definíciami statických objektov, ale aj modelovaním ich interakcií a najmä činností.

Automatické testovanie generovaných programov vychádza z vlastnosti generovania kódu. Táto funkcionalita môže zahŕňať automatické vytváranie množín údajov na testovanie, ich automatické otestovanie ako aj automatické korigovanie chýb.

Technológia neprocedurálneho vstupu vytvorená pomocou na seba nadväzujúcich menu, že v každej fáze môže menu a podmenu obsahovať len funkcie, ktoré v danej situácii má zmysel použiť.

Editor a generátor V/V formulárov ako editor dialógov, menu, zostáv a podobne.

Modelovanie podnikových procesov (BPM – Business Process Modelling) poskytujú najnovšie zo systémov CASE. Umožňuje definovať diagramy dátových tokov, diagramy závislosti medzi procesmi (resp. dekompozícia procesov), definíciu funkcií systému a podobne.

Systém existuje na viacerých platformách.

Podpora a presadzovanie vývojových štandardov. Jedná sa o štandardy ako aplikácie písať, aké môžu byť väzby medzi jednotlivými prvkami aplikácie a podobne. Tým, že celý tím používa rovnaký CASE nástroj, je dodržiavanie vývojových štandardov všetkými členmi tímu jednoduchšie.

Znovupoužiteľnosť návrhov a vytvorených komponentov je dôležitá vlastnosť, ktorá šetrí náklady pri vývoji.

Porovnanie modelov a zistenie odlišností podľa dôležitosti, prípadne zlučovanie modelov je tiež súčasť rady nástrojov.

Správa požiadaviek používateľov. Jasná a včasná definícia požiadaviek je základným stavebným prvkom úspešnej aplikácie, ktorá má spĺňať očakávanie ich

používateľov. Obvykle sa požiadavky v čase menia a vyvíjajú a nástroje CASE umožňujú efektívnu správu týchto zmien.

Výhody CASE

- Vyššia produktivita práce – vďaka automatizácii, ktorú CASE nástroje priniesli došlo k výraznému urýchleniu realizácie jednotlivých úloh, napr. zoberme si len kreslenie rôznych druhov diagramov.
- Menej chýb – využitie CASE so sebou prináša prehľadné grafické prostredie s množstvo priebežných alebo dodatočných kontrol, ktoré vedú k včasnému odhaľovaniu chýb. Čím sa chyby skorej odhalia a odstránia, tým sú aj náklady s tým spojené nižšie.
- Zjednodušenie komunikácie v tíme – CASE pomáhajú pri vysporiadaní sa s problémami pri riadení tímov, čo je veľmi časovo aj nákladovo náročná činnosť.
- Jednoduchšia údržba a ďalší vývoj produktu - výsledkom dokonalejšej analýzy, kvalitnejšieho návrhu, automatického generovania kódu, automatizovaného testovania a včasného odstraňovania chýb je vyššia kvalita celého systému. Vďaka nástrojom podporujúcim reinžinierstvo vyžaduje tento proces menej času a ďalších zdrojov, čo vedie k vyššej efektivite práce.
- Kvalitnejšia dokumentácia - k dispozícii je aj kvalitná dokumentácia, ktorá zabezpečuje jednoduchšiu a lacnejšiu údržbu počas celého životného cyklu systému. Dokumentácia je jednoducho vytváraná prostredníctvom automatického generovania. Väčšina nástrojov využíva revíziu poznámok k vývoju a údržbe systému.
- Umožnenie väčšej participácie používateľov na vývoji – použitie CASE nástrojov umožňuje hlavne vďaka zrozumiteľným diagramom a ich možnými popismi väčšiu participáciu používateľov na vývoji produktu, čo vedie k lepšiemu prijatiu nového systému. Zároveň dochádza aj k zníženiu doby potrebnej k zaučeniu používateľov na prácu v novom systéme.

Nevýhody CASE

- Možnosť nevhodného výberu nástroja alebo ich kombinácie – aby sa vďaka použitiu nástroja CASE ušetrili náklady, je dôležité dbať na vhodný výber nástroja. Ďalej sa musí venovať pozornosť integrácii nástrojov CASE a integrácii dát cez všetky platformy. Treba mať aj na zreteli možnosť prechodu z jedného nástroja CASE na iný.
- Vysoké náklady na ich zaobstaranie – nástroje CASE nie sú vo väčšine prípadov lacnou záležitosťou. Pri kalkulácii nákladov na ich zaobstaranie nesmieme zabudnúť zahnúť náklady na potrebný hardvér, softvér, školenie zamestnancov a náklady na podporu. Tieto celkové náklady potom možno porovnať s úsporami, ktoré si od zavedenia CASE nástroja sľubujeme.
- Vysoké nároky na znalosti používateľov CASE – nástroje CASE sú komplexné a komplikované nástroje s mnohými nadväznosťami ich jednotlivých častí a celou radou metód a techník, treba preto počítať s nákladmi a časom potrebným na zaškolenie a osvojenie si práce budúcich používateľov s novým nástrojom CASE. (Z tohto dôvodu vzniklo odvetvie poradenstva pre nástroje CASE. Konzultanti z príslušných spoločností sú pripravený poskytnúť zákazníkom školenia priamo

v priestoroch ich firmy. Tieto školenie sú významným faktorom urýchľujúcim proces oboznámenia sa s novým nástrojom CASE a učenia.)

Členenie CASE podľa ŽC IS

- Pre CASE – globálna stratégia
- Upper CASE – informačná stratégia a analýza (plánovanie, riadenie IS)
- Middle CASE – globálny a detailný návrh
- Lower CASE – implementácia (generovanie kódov, testovacích údajov, testovanie generovaných programov)
- Post CASE – zavedenie do prevádzky a prevádzka

Členenie CASE podľa stupňa integrácie

- Tools – automatizácia ľubovoľnej úlohy ŽC
- Toolkits – čiastočná alebo komplexná podpora v rámci jednej fázy ŽC IS
- Workbenches - čiastočná alebo komplexná podpora úloh minimálne v dvoch fázach ŽC IS
- I-CASE (Intergrated) - čiastočná alebo komplexná podpora celého ŽC IS

Smery integrácie

- V oblasti hardvéru a systémového softvéru
- V oblasti používateľského prostredia
- V oblasti riadenia (komunik. mechanizmus)
- V objektovej oblasti
- V oblasti meta BD (rovnaká)
- V oblasti formátu a záznamu dát (syntaktická integrácia)
- V oblasti riadiaceho jazyka (sémantická integrácia – rovnaký slovník príkazov alebo funkcií menu, syntax príkazových riadkov, štruktúru použitých riadiacich polí a hodnôt v nich)

Členenie CASE z hľadiska režimu práce

- CASE pre prácu na individuálnych pracovných miestach (decentralizované)
- CASE pre prácu v režime distribuovaného spracovania dát
- CASE pre prácu v režime centralizovaného spracovania dát

Metodológie a metódy podporované CASE

- Na podporu štruktúrovaných metodológií (SSADM, SDM, Yourdona, Merise) CASE: SDW, CASE 4/0, Excelerator, Adelia, Westmount I-CASE, Informix open CASE atď.
- Na podporu objektovo-orientovaných metodológií (OMT, RAD3, RUP) CASE: Paradigm Plus, Rational Rose
- Na podporu štruktúrovaných a objektovo-orientovaných metodológií

CABE - Computer Aided Business Engineering

- Vznikli podobne ako CASE v 80.-tych rokoch
- Značná pozornosť sa im venuje až od začiatku 21. storočia
- Výrobcovia ich označujú názvami:

- Business Process Modeling Tools,
- Enterprise Modeling Tools,
- Business Process Management Tools,
- Enterprise Architecture Tools.

- jeden nástroj môže spadať do obidvoch oblastí – CASE aj CABE

Definícia: Systémy CABE nie sú zamerané výhradne na vývoj informačných systémov, ale na systematickú podporu vecného (business) systému firmy, postavenú na jeho, dôkladnou analýzou vytvorenom, modeli, natoľko exaktným, aby bolo možné pomocou neho celý vecný systém riadiť.
[prof. Repa]

CABE nástroje by mali podporovať tieto oblasti business modelovania

- Modelovanie cieľov podniku a stratégie.
  Procesy, ktoré v podniku prebiehajú, by mali primárne vychádzať a podporovať ciele podniku, ktoré sú definované v stratégii firmy. Cieľom tejto oblasti by malo byť zachytenie týchto cieľov v nejakej zrozumiteľnej forme a zachytenie ich väzby na procesy, ktoré v podniku prebiehajú. Definovanie cieľov a ich naviazanie na procesy, spolu s metrikami procesu sú dôležitým predpokladom na meranie efektívnosti a optimalizácie procesu.
- Modelovanie organizačnej štruktúry.
  Za procesy, ktoré vo firme prebiehajú musí byť vždy niekto zodpovedný. Jednotlivé činnosti vykonávajú zamestnanci, ktorí majú v organizačnej štruktúre firmy definovanú určitú pozíciu. Definovanie a modelovanie organizačnej štruktúry je predpokladom pre jej prepojenie s procesmi. Organizačná štruktúra by nielen mala byť s procesmi prepojená ale aj navzájom konzistentná.
- Modelovanie topológie podniku.
  Toto modelovanie predstavuje zachytenie geografickej štruktúry podniku, prvkov okolia podniku a možnosti väzieb medzi prvkami okolia a procesmi.
- Procesné modelovanie.
  Procesné modelovanie predstavuje vlastnú analýzu a modelovanie procesu podniku. Tu je dôležité, zamerať sa na to, akú metódu a notáciu príslušný nástroj používateľom poskytuje. Podľa metódy a notácie sa potom nástroje môžu líšiť tým, aké prvky je možné v modely zachytiť, aká je možnosť zachytenia hierarchie procesov, zachytenie väzieb na organizačnú štruktúru a definovanie cieľov procesu a jeho metrík.
- Modelovanie okolia podniku.
  Cieľom modelovania okolia podniku je snaha zachytiť externé prvky podniku, ktoré vstupujú do procesu alebo vystupujú z procesu, a ktoré sú z hľadiska priebehu procesu alebo organizačnej štruktúry dôležité. Jednoducho povedané ide o zachytenie všetkých externých okolností, ktoré nejako chovanie podniku ovplyvňujú a podnik ich nemá plne pod svojou kontrolou.
- Modelovanie technologickej oblasti.
  Tento typ modelovania umožňuje zobraziť všetky technologické a technické prostriedky a ich prepojenia, ktoré podnik má a využíva.
- Modelovanie aplikácií a systémov (používaného SW).

Modelovanie aplikácií zahrňuje modelovanie všetkých v podniku používaných softvérových aplikácií a systémov a ich nadväzností.

- Modelovanie workflow.
  Modelovanie workflow predstavuje modelovanie postupnosti jednotlivých operácií, ktoré predstavujú činnosť osoby, mechanizmu, skupiny osôb, celej organizácie alebo stroja. Diagram, ktorým sa workflow zobrazuje je typ vývojového diagramu.

Oblasti business modelovania by mali byť podporované funkciami

- Ponuka vzorových šablón procesov, resp. referenčné modely alebo metodiky. Referenčné modely sa zaoberajú popisom funkcií alebo procesov podniku. Tento popis vychádza z „best practices“, ktoré sa premietajú do referenčného modelu a tak je možné tento model použiť ako vzor pre modelovanie v každom podniku. Špecifiká príslušných podnikov treba dodatočne domodelovať. Výhodné je, ak sú jednotlivé modely rozčlenené podľa predmetu podnikania (napr. bankovníctvo, automobilový priemysel, chemický priemysle, ...).
- Podpora tímovej práce, čo znamená, že na modely môže pracovať viacero analytikov, pričom je rôznymi spôsobmi riešené zamykanie a odomykanie modelov, alebo ich častí, pre jednotlivých analytikov modelujúcich v jednom časovom momente.
- Previazanosť modelov, t. j. jednotlivé modely a submodely je nutné dať do súladu. Nástroj by preto mal podporovať prepojenie jednotlivých modelov, napr. procesného modelu s organizačnou štruktúrou alebo cieľmi podniku.
- Kontrola konzistencie znamená, že v každom modely by sa nemali objaviť rozporné prvky a celý by mal byť súdržný vo všetkých svojich častiach.

CABE nástroje by mali firmám poskytnúť tieto výhody

- rýchla úprava existujúcich alebo návrh nových podnikových procesov v závislosti na zmene potrieb podniku,
- jasné a presné definovanie obchodných procesov a zjednodušenie ich zdieľania vnútri podniku,
- optimalizácia obchodných procesov a zvýšenie efektivity podávania informácií vo vnútri podniku,
- integrovanie procesov, informácií a aplikácií.

IDE - Integrated Development Environment

Definícia: IDE je programovacie prostredie integrované do softvérovej aplikácie, ktoré poskytuje tvorcu GUI, editor textu alebo kódu, kompilátor a /alebo interpreter a debugger [Webopedia].

Základnými prvkami každého IDE sú:

- Prehľadný editor zdrojového kódu, ktorý farebne odlišuje jednotlivé časti zdrojového kódu.
- Kompilátor a/alebo interpreter. Kompilátor prekladá zdrojový kód napísaný v niektorom programovacom jazyku do vykonateľnej formy, napríklad strojového kódu. Interpreter vykonáva interpretáciu zdrojového kódu v čase behu programu.

- Utility k zostavovaniu programu (linker), čiže programy, ktoré zoberú jeden alebo viac objektov generovaných kompilátorom alebo z knižníc a skombinujú ich do jedného vykonateľného programu.
- Debager (debugger) je program na odladťovanie a kontrolu programu. Pri debagovaní ide o proces nájdenia a zredukovania resp. odstránenia chýb z programu.
- Automatické dokončovanie písania kódu. Napr. IntelliSense, ktoré predstavuje Microsoft implementáciu automatického dokončovania písaných kľúčových slov ako názvov premenných, vyvolávanie metód a funkcií.
- Nástroj na upozornenie a odstraňovanie chýb v zdrojovom kóde v reálnom čase, ktoré je dôležité najmä pri dolaďovaní aplikácie.
- Tvorca grafického používateľského rozhrania (GUI), ktorý umožňuje jednoduché vytvorenie používateľského rozhrania, čiže jednotlivých obrazoviek (formulárov) pomocou vopred nadefinovaných elementov, ktoré reprezentujú určitú časť zdrojového kódu. Preddefinované elementy sú väčšinou prístupné v okne nástrojov (toolbox) alebo paneli nástrojov (toolbar).
- Systém pre rýchly vývoj aplikácií (Rapid Application Development – RAD), ktorý je hlavne určený a spojený s vyššie spomenutou tvorbou grafického používateľského rozhrania.
- Podpora refaktoringu, ktorý súvisí s modernými trendmi programovania. Refaktoring je zmena štruktúry resp. návrhu programu bez zmeny jeho správania s cieľom získať čistejší, čitateľnejší kód a kód s lepším návrhom, ľahšou údržbou a modifikovateľnosťou.
- Generovanie dokumentácie zo zdrojového kódu.

Členenie IDE

- Podľa počtu podporovaných jazykov
  - orientované na jeden programovací jazyk - ponúkajú množstvo zaujímavých a užitočných funkcií (napr. JDeveloper, Delphi)
  - podpora väčšieho počtu jazykov – univerzálnejšie (napr. Eclipse, MS Visual Studio)
- Podľa prostredia
  - Negrafické – hlavne vo svete open source
  - Grafické – vplyvom grafických OS ako MS Windows

Integrácia IDE s CASE/CABE

- IDE sú rozširované o CASE/CABE funkcionalitu. Dôvodom existencie tohto prístupu je skutočnosť, že je vhodné, aby mali vývojári prístup aj k modelovaniu.
- CASE/CABE nástroje ponúkajú pluginy pre IDE, resp. sú pluginy do IDE. Dôvodom existencie tohto prístupu je skutočnosť, že veľké projekty vyžadujú robustné CASE/CABE nástroje, ktoré je možné integrovať do IDE.