Metodika návrhu a realizace informačního systému – strukturální a objektová analýza.

Thursday, May 30, 2013 8:23 AM

Existuje v zásadě několik metodik, které popisují analýzu, návrh a realizaci informačních systémů

Jedná se o více méně podrobný popis postupu, který vede návrháře jasně definovanými fázemi krok po kroku při vytváření IS

Metodiky jsou často obecné a každá firma si je může přizpůsobit pro vlastní potřebu a prostředí

Mezi nejpoužívanější patří Strukturální analýza (ta je nejstarší), SSADM (vznik 80. léta ve Velké Británii) a Objektová analýza (konec 80. let, 90. léta)

Metodiky návrhu a realizace informačního systému

Fenomén úhlu pohledu
Chceme-li se vyhnout potížím s lokálním rozhodováním, musíme postupovat metodicky (ne chaoticky), strukturálně, dle „dobrých“ osvědčených vzorů.

Návod jak postupovat nám dávají ověřené postupy – vypracované metodiky.

Metodiky odrážejí určité **náhledy** na „realitu“, říkají „jaké“ kroky učinit v jakém pořadí a „jak“ je provádět. Dobré metodiky nám říkají i „proč“ to tak má být.

Metodiky jsou konservovanou zkušeností několika generací programátorů a projektantů. Zobecnění principů, zásad, které se osvědčily, viz historie UML.

Místo abychom se snažili popsat systém jako celek, vytváříme na něj jednotlivé pohledy – jeho jednotlivé, dílčí modely. Díváme se na systém postupně z jednotlivých „míst pozorování“, z jednotlivých perspektiv.

Díváme-li se na systém z jednoho místa, **opomíjíme** vlastnosti z tohoto místa „neviditelné“, nepodstatné a tím si práci zjednodušíme tak, že je mentálně zvládnutelná. Jednotlivé pohledy jsou jednodušší, zvládnutelné.

Opomíjené vlastnosti se neztratí, jsou hlavními vlastnostmi v jiných pohledech - modelech.

Pohledy musíme volit tak, že postupně popíšeme všechny **relevantní vlastnosti systému**. Postupně popíšeme vše, co potřebujeme k dosažení stanoveného cíle.

Z jednotlivých pohledů lze zpětně **zrekonstruovat celý systém** (počítačová tomografie).

Pro tvorbu různých pohledů jsou obvykle využity **diagramy – grafické objekty**, jejichž kombinací lze tyto pohledy vytvářet.

Diagram je graficky znázorněný model. Diagram popisuje jistou část modelu pomocí grafických symbolů.

Tento přístup lze přirovnat k **modelu stavby**, který je tvořen **syntézou dílčích stavebních plánů** odpovídajících specifickým pohledům na stavbu
– plán hrubé stavby, plánu rozvodů elektřiny, plánu rozvodů vody, … V každém z těchto plánů jsou zobrazeny pouze elementy modelu podstatné pro daný pohled, od ostatních elementů modelu je abstrahováno. Pohledy **nejsou nezávislé**, dohromady tvoří **konzistentní pohled na systém**, tedy konzistentní model.

Pro tvorbu modelu systému, respektive pro **tvorbu pohledů na s**ystém, jejichž syntézou bude

model, definuje např. UML **devět typů** diagramů.

Zkušenosti ukazují, že je účelné **tvorbu (návrh a realizaci) IS organizovat do posloupnosti etap**, mluvíme o tzv. **životním cyklu IS**.

Životní cyklus IS není statická sekvence činností. Popisuje dynamiku vývoje, měnící se vnější podmínky (změny legislativy), postup od obecného ke speciálnímu. Životní cyklus IS začíná prvotní představou o systému a končí vyřazením systému z provozu. Samozřejmě není znám přesný a úplný (matematický) model životního cyklu. Nepřesné modely ŽC jsou však nepostradatelné pro řízení projektů.

Základní fáze životního cyklu IS:

- Stanovení globálních cílů, specifikace požadavků, specifikace vlastností, které by měl budoucí systém realizovat (implementovat).
- Analýza systému, tvorba analytického, logického modelu systému, model požadovaných vlastností systému.
- Návrh (design), specifikace způsobu, jak požadované vlastnosti implementovat
- Implementace systému, převedení navrženého systému do spustitelného kódu
- Testování vytvořeného kódu
- Nasazení systému, provoz, údržba

V rámci životního cyklu IS řešíme i řadu organizačních a ekonomických otázek:
proč, pro koho, termíny, cena, pracovní tým, situace na trhu, návratnost investice, řízení pracovního týmu, hardwarové prostředky, dokumentace, reakce na změny.

Model vodopád
Pro řízení a vývoj IS by bylo ideální, kdyby po úplném ukončení jedné fáze, etapy ŽC následovala další a k předchozí etapě by nebylo nutné se již vracet.

Model vodopád je složen z posloupnosti vymezených činností.

Realita je však díky své složitosti jiná. Jednotlivé fáze, etapy ŽC se překrývají. Tak jak postupujeme v ŽC, postupně upřesňujeme výchozí poznatky a musíme se vracet k předchozím etapám. V průběhu práce se také mění výchozí požadavky uživatelů a vnější (legislativní, technologické prostředí).

Prototypový model
Tento model se začal prosazovat v 80.letech. Jeho hlavním cílem je urychlení vývoje IS využitím prototypů.

Prototyp můžeme chápat jako zjednodušenou implementaci celého systému nebo jako plnou implementaci části systému.

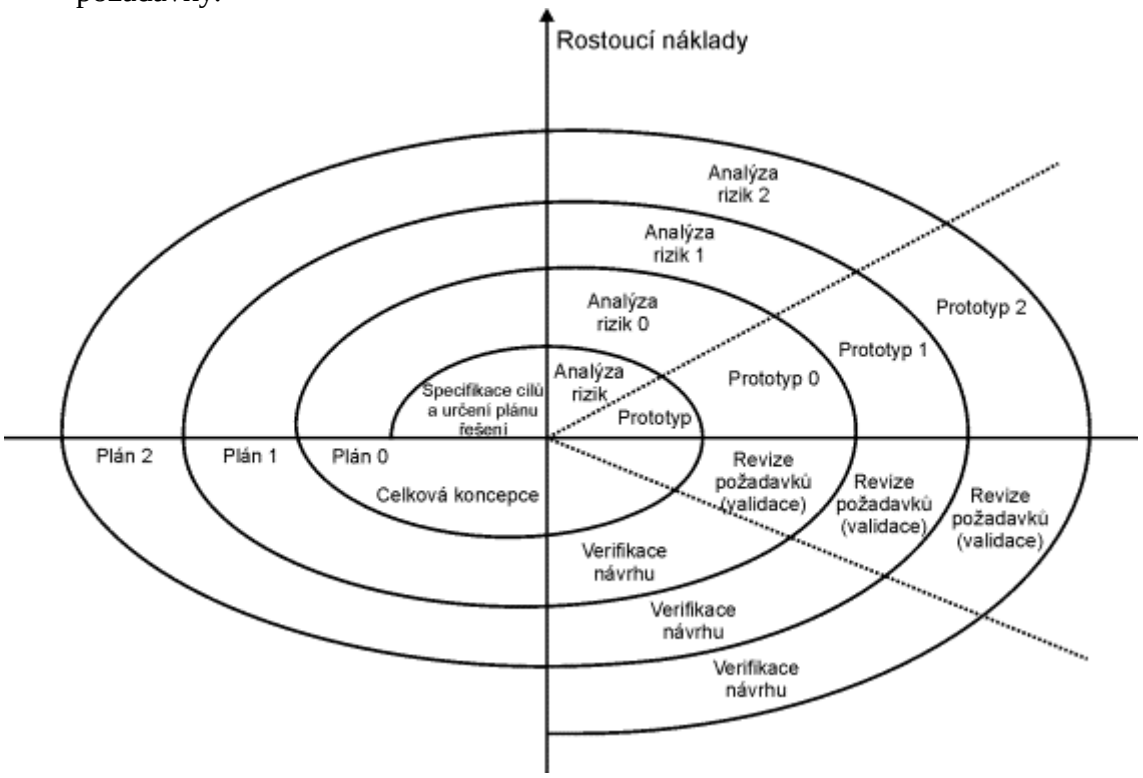
Prototyp je provedena v co nejkratším čase a v takové funkčnosti, která umožní ověřit, otestovat požadované vlastnosti (např. umožňuje zákazníkovi reagovat na výsledky). Na základě vyhodnocení vlastností prototypu jsou upřesňovány požadavky a modifikován další vývoj.

Spirálový model
Tento model vytvořil B.W.Boehm v roce 1988 a je kombinací prototypového přístupu a analýzy rizik.

Základem celého modelu je neustálé **opakování vývojových kroků** tak, že v každém dalším kroku se na již ověřenou část systému přibalují části na vyšší úrovni.

Postup vývoje v jednotlivých krocích se skládá z následujících částí.

- Specifikace cílů a určení plánu řešení.
- Vyhodnocení alternativ řešení a **analýza rizik s daným řešením souvisejících** (je daná alternativa vyhovující, únosná).
- Vývoj prototypu dané úrovně a jeho předvedení a vyhodnocení.
- Revize požadavků neboli validace (testování zda prototyp pracuje tak jak má).
- Verifikace, neboli ověření zda celkový výstup daného kroku je v souladu se zjištěnými požadavky.



Úhlová dimenze modelu reprezentuje postup prací v čase, radiální dimenze pak rostoucí náklady. Nevhodné prototypy jsou opuštěny. Každý nový průchod cyklem rozvíjí nejlepší prototyp.

Nevýhodou modelu je špatný odhad termínu dokončení projektu a jeho celková cena. Proces vývoje je jakoby otevřen.

Strukturální a objektová analýza

Objektový model – diagram tříd

Původní strukturální přístup k analýze IS spočíval v **rozdělení** systému na funkční a datovou část (např. DFD diagramy a ER model).
Přínosem byla funkční hierarchická dekompozice – psaní programu shora dolů a datové konceptuální modelování.

Rostoucí složitost systémů (magická hranice 1000 entit a 10000 funkcí) znemožňuje soudržnost datové a funkční vrstvy. Konstruovali se matice, kde řádky jsou datové entity a sloupce funkce systému. Koexistence se označovala křížkem – možnost dekompozice systém – diagonální matice.

Objektový přístup čelí kromě jiného složitosti systému tím, že třída - objekt jako nositel (funkční) odpovědnosti – dovednosti, plně **odpovídá** za svá data.

Objekt má svou identitu, vlastnosti, chování a **odpovědnost**. Síla odpovědnosti spočívá v tom, že je **nedělitelná** – žádný jiný objekt nemůže odpovědnost sdílet – dělit se o ni, plést se do ní.

Modelování tříd a objektů je **klíčová aktivita** objektově orientovaného vývoje.

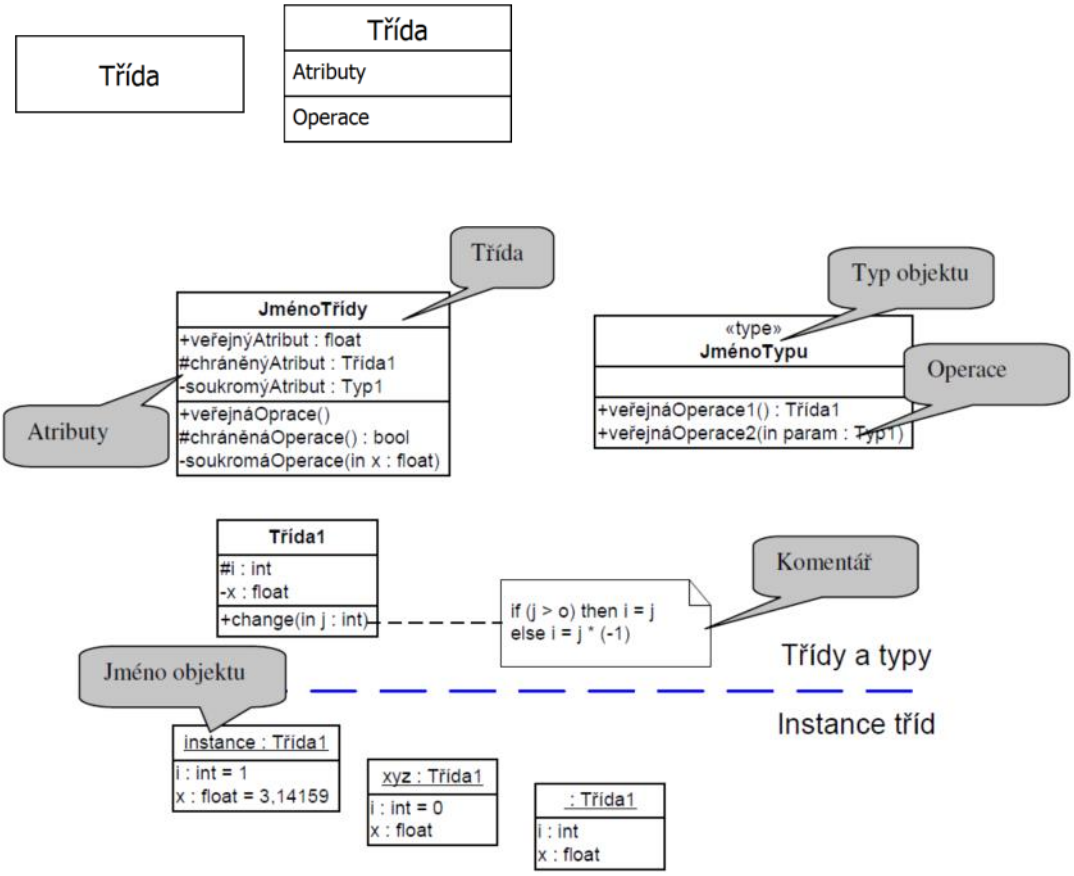
Třída je popisem **množiny objektů** sdílejících stejné vlastnosti - atributy, chování – operace (metody) a vztahy.
Objekt je instancí třídy (chybně se pojem třída a objekt volně zaměňují).

Definice – J. Rumbauh: objekt je diskrétní entita s jasně definovaným rozhraním, které zapouzdřuje stav a chování.

Třídu si můžeme představit jako razítko, objekty jsou pak otisky tohoto razítka, které vidíme na papíře.

Při návrhu třídy neuvažujeme o konkrétním naplnění atributů, pouze určíme jejich název a typ. Teprve při vzniku instance objektu se atributům přiřadí skutečné hodnoty.

Třída je jednoznačně určena svým názvem (v příslušném názvovém prostoru – balíčku). Pro třídu je možno definovat vlastnosti - atributy (Attribute) a chování - operace (Operation).
Vizuálním elementem:



Hledání tříd, jejich atributů a kompetencí - vyberme z reality objekty, kandidáty pro zobecnění na třídy a prověřme jejich vhodnosti:

- Potenciální třída je smysluplná, pokud je nezbytná pro funkci systému.
- Potenciální třída je dostatečně stabilní a invariantní vůči vnějším změnám např. technologie, legislativy apod.

Hledání tříd na základě analýzy podstatných jmen a sloves.

Analyzujeme jazyk problémové domény, např. text sebraných požadavků. Podstatná jména a jejich spojení mohou označovat třídy nebo atributy.

Slovesa mohou označovat odpovědnosti, chování tříd.

Pozor na skryté, utajené třídy, které nejsou v textu uvedeny.
From <https://d.docs.live.net/e3534876709763a3/Dokumenty/ZCU/Statnice/Statnice.docx>