Úvod do technologie programování a programovacích stylů, objektově orientovaný návrh, základní UML diagramy, psaní programů v Javě

Programování je proces vytváření software splňující určité funkce. K programování využíváme programovací jazyky, dříve procedurální (nestrukturované (asembler) - bez podprogramů, pouze skoky v programu (goto), nebo strukturované (basic, c, pascal) - podprogramy, skoky jen velmi vyjímečně), dnes nejčastěji objektově zaměřené (objekty jsou stavební kameny a obalují funkce, využití technik OOP).

Dříve bylo programování spojeno s testováním, nyní je proces rozdělen do několika fází:

Analýza - Sbírání informací od zákzníka či zadavatele

Návrh - Návrh SW tak, aby splňoval požadavky analýzy

Kódování - Implementace návrhu

Ladění - Testování a odstraňování chyb

Styly

Vodopád - fáze jdou za sebou, nevracíme se, ani neodbočujeme (nejstarší), týmy se zapojují do procesu pokud je na řadě jejich blok

Formální - návrh je matematicky zcela do detailu popsán a jednotlivé bloky modelu jsou poté 1:1 implementovány

Evoluční - začíná se malým funkčním SW, na který nabalujeme další funkce, ale nevíme, jaké funkce bude v budoucnu mít

Iterativní - proces je rozdělen na iterace, v každé iteraci je naplánováno několik funkčností, iterace stejně dlouhé, po iteraci je SW vždy v kompletní spustitelné podobě

Komponentový - využití hotových komponent, či implementace vlastních, po té stavění SW z komponent jako z kostek

Objektově orientovaný návrh

OOP umožňuje, a klade si za cíle:

Znovupoužitelnost - abstraktní části kódu jsou univerzální a je tedy možné je používat stále dokola

Robustnost - návrh předpokládá všechny vstup, nejen ty, které jsou očekávané

Přizpůsobivost - návrh lze pouze s minimálními zásahy do kódu použít na různých platformách

Pilíře OO návrhu

Dědičnost, **Zapouzdření**, **Polymorfismus** (viz PPA2)

Modularita - rozdělení programu na bloky - moduly

Abstrakce - Návrh na abstraktní úrovni je snadno implementovatelný

Jazyk UML

Jazyk pro abstraktní návrh OO SW, modelování SW před implementací, navrhujeme pouze strukturu, nikoliv funkčnost. Typy UML:

diagram tříd a objektů - popisují statickou strukturu systému

modely jednání (diagram případů užití) - dokumentují možné případy použití systému, události, na které musí systém reagovat

scénáře činností (diagramy posloupností) - popisují scénář průběhu určité činnosti, těchto činností je model jednání složen

diagramy spolupráce - zachycují komunikaci spolupracujících objektů

stavové diagramy - popisují dynamické chování objektu nebo systému

diagramy aktivit - průběh aktivit procesu nebo činností

diagramy komponent - popisují rozdělení systému na funkční celky a definují náplň jednotlivých komponent

diagramy nasazení - popisují umístění funkčních celků

Pro návrh diagramu tříd máme následující parametry:

Atributy a metody - definujeme viditelnost a vlastnosti (+ public, # protected, - private, / readonly, in/out směr)

Třída: obdélník, název tučně, pak seznam atributá a metod

Objekt: Název:Třída

Dědičnost - šipka od potomka k rodiči, může být i vícenásobná

Asociace - 0..1 1..1 0..N 1...N --> šipka od první třídy a ta obsahuje odkazy/využívá x..y objektů druhé třídy (např Podvozek 1 ---> 4 Kola), nad šipkou název činnosti (pokud možno lepší než "má")

Kompozice - je li objekt v kompozici s jiným, je na něj silně vázán (například seznam a položka seznamu. seznam může existovat i bez položek, ale položka bez seznamu ne), značka prázdný kosočtverec

Agregace - volná vazba, abstrakce "používá", objekty však mohou fungovat i nezávisle na sobě (například počítače v síti), značka plný kosočtverec

Java

Návrh - UML diagramy, CRC karty (karty, kde je pro každá blok sloupec Má za úkol a Vyžaduje). Je li toho na kartě příliš, je vhodné zvážit její rozdělení

Implementace - podle návrhu, karta odpovídá třídě.

Ladění a testování - kontrolní výpisy, logování, watch na proměnných, ..., po té test pro různá data (generovaná, "nepříjemná") a sledování správnosti, rychlosti, využití zdrojů, uvolnění

Dokumentace - štábní kultura, konvence pojmenovávání, komentáře, Javadoc komentáře a generovaná dokumentace