Metodika a životní cyklus vývoje softwaru

Životní cyklus softwaru je proces, během kterého je cyklus vyvíjen. Je několik různých způsobů a cest, podle kterých je možné postupovat při tvoření.

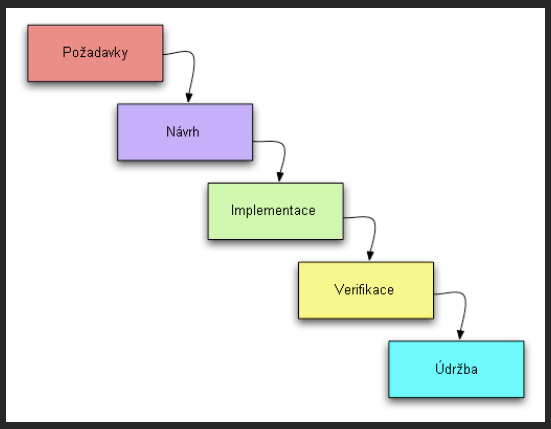
Modelů je opravdu mnoho, každá firma může mít například vlastní. Jsou ale nějaké obecně známé nebo nějaké, které se využívají.

Metodika je soubor postupů, kterými se vývoj řídí

Iterativní vývoj

Iterativní vývoj znamená předělávat, případně opakovat. Kroky směřují k tomu, aby systém byl opravený, případně vylepšený. Každá jedna iterace má svojí analýzu, návrh, testování, vývoj.

Vodopádový model



Protiklad agilního vývoje. Jeho hlavní problém je, že se nevracíme zpátky a proto se jedná o nevýhodnou metodiku. Jedná se o metodiku, při které je možné přijít k dalšímu kroku bezprostředně poté, co je perfektně dodržen aktuální krok.

Jedná se o model iterativní s pouze jedinou iterací.

Tento model je využíván v NASA a obecně v technologických gigantech.

Pozitiva: Často odhalení chyby v počátku vývoje je levnější, než její opravení v pozdější době, jednoduchý přístup, klade stejný důraz jak na kód, tak na dokumentaci.

Kritika: Prý není možné dovést část k dokonalosti, pokud klient změní požadavek, přijde veškerá práce vniveč.

Agilní metodika

Agilní metodika je typ vývoje software který je **iterativní** nekonečněkrát. Jeho specialitou a výhodou je, že je velmi přizpůsobivý na změnu požadavků klienta. Jedná se o protiklad vodopádového modelu, jelikož agilní se neustále opakuje. Umožňuje jednoduchou konkurovatelnost, jelikož vývoj se může kdykoliv jakkoliv upravit.

Spirálový model

Spirálový model je jeden z agilních způsobů řízení vývoje projektu. Jedná se o model, který se neustále točí dokola v procesech. V tomto případě ->

Analýza – Specifikujeme problém
Hodnocení – Zjistíme, jak problém vyřešit
Vývoj – Problém vyřešíme
Plánování – Plánování dalšího oběhu a domluva s klientem.

Samozřejmě je možné přidat další věci a kroky, jako například testování, případně získání feedbacku od zákazníka. V nejlepším případě je zákazník stále s námi a má připomínky.

Extrémní programování

Extrémním programováním se rozumí takové programování, které už od začátku přesně stanovuje práci všem pracovníkům. Proč extrémní? Protože všechny určité části jsou dotažené do extrému

Příklad: Pokud revize kódu proběhla úspěšně, revize kódu probíhá neustále, to samé platí pro testování. Pokud je testování úspěšné, pořád se testuje pomocí unit testů. Pokud je návrh úspěšný, navrhuje se neustále všemi programátory. Pokud je důležitá architektura, každý se bude podílet na její úpravě. Extérmně krátké iterační doby, z dnů a týdnů na minuty, hodiny, dny.

Extérmní programování má čtyři základní pilíře, kterýmiž jsou:

Komunikace – Nutná naprosto přesná komunikace mezi klientem a programátorem, programátorem a manažerem.
Jednoduchost – Vždy se programuje pouze to, co je potřeba, nikdy ani o řádek více.
Zpětná vazba – Neustále unit testy, které kontrolují, zda-li se něco nerozbilo.
Odvaha – Potřeba mít odvahu na řešení problémů, které na sebe navalí další a další problémy.

Je spousta dalších druhů metodik, třeba RAD – Rapid Application Development, prototypová metodika.