Disciplína verzus agilita

LENKA LITVOVÁ

Slovenská technická univerzita
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava
lenli.me@gmail.com

Abstrakt. V dnešnej dobe, keď sú softvérové systémy čoraz rozsiahlejšie a komplexnejšie, je plánovanie neoddeliteľnou súčasťou vývoja softvérového produktu. Plánovanie umožňuje lepšie pochopenie cieľov projektu, zefektívnenie procesu jeho vývoja, redukuje neurčitosť a vytvára základ pre riadenie projektu. Vytváranie plánu závisí od celkového prístupu k vývoju softvéru a jeho životného cyklu. Vopred je potrebné poznať aké činnosti treba naplánovať a aké je ich poradie. Esej je zameraná na dva najpoužívanejšie prístupy k vývoju softvéru: klasický plánovo riadený a novší agilný. Existencia nového prístupu bola vynútená meniacimi sa požiadavkami na softvér, ktoré vyplývajú z rýchleho vývoja technológie i celej spoločnosti. Otázkou ostáva možnosť aplikovania overených postupov pri zmenených podmienkach 21. storočia. Na úvod sú vysvetlené základné princípy oboch prístupov a následne na základe prieskumov, tak v komerčnej ako aj akademickej oblasti, sú porovnané oblasti ich nasadenia, výhody a nevýhody, ktoré so sebou prinášajú. Keďže ani jedno riešenie nie je povestným Brooksovým strieborným nábojom, je preskúmaná aj možnosť využitia hybridného prístupu, spájajúceho ich výhody a eliminujúceho ich nevýhody, so zameraním na školský tímový projekt.

Úvod

Počas krátkej histórie, ktorú má už softvérové inžinierstvo za sebou, bolo mnoho prístupov k vývoju softvérového systému na výslní a mnohé aj upadli do zabudnutia. V päťdesiatich rokoch minulého storočia bola väčšina projektov technicky orientovaných a tento prístup sa hodil na riešenie vtedajších nízkoúrovňových problémov. Rast projektov si vyžiadal aj zavedenie nových formálnejších postupov. Na konci sedemdesiatych rokov bol už uznávaný štruktúrovaný prístup a vodopádový model. Neúspešné projekty však opäť viedli k potrebe nového pohľadu. Ďalšie desaťročie so sebou prinieslo expertné systémy, vyššie programovacie jazyky, objektovo orientovaný prístup, vizuálne programovanie a znovupoužitie, vďaka čomu sa výrazne zvýšila produktivita.

Dvadsiate prvé storočie sa vyznačuje neustálym zdokonaľovaním softvéru i hardvéru a rastúcim záujmom oň. Softvérové systémy sú dokonalejšie, väčšie

Manažment v softvérovom inžinierstve, október 2007, s. 1-1.

a komplexnejšie. Chaotickým neriadeným prístupom by asi len ťažko dospel nejaký softvérový projekt do úspešného konca a to je predsa hlavný cieľ. Základom úspešného projektu je plánovanie, ktoré umožňuje lepšie pochopenie cieľov projektu, zefektívnenie procesu jeho vývoja, redukuje neurčitosť a vytvára základ pre riadenie projektu. Vytváranie plánu závisí od celkového prístupu k vývoju softvéru a jeho životného cyklu. Momentálne sú na výslní klasický plánovo riadený prístup a agilný. V tejto eseji sa snažím o ich porovnanie a nájdenie toho najlepšieho prístupu k tímovému projektu, ktorý by ho doviedol k jeho úspešnému ukončeniu.

Všetko treba naplánovať a zdokumentovať

Asi takto by sa dala v skratke vysvetliť myšlienka plánovo riadených metód. Sú podstatne staršie v porovnaní s agilnými. Kým synonymom pre agilné metódy je prispôsobivosť, pre plánovo riadené je to disciplína ako to vyplýva z ich spoločných charakteristík[3]:

- sústredenie sa na znovupoužitie a predvídateľnosť
- definovaný, štandardizovaný a inkrementálny proces vývoja
- dôkladná dokumentácia
- architektúra softvérového systému definovaná vopred
- podrobné plány, poradie činností, úlohy, zodpovednosti a popis výsledného produktu
- vytvorená základňa pre monitorovanie, kontrolu a vzdelávanie
- priebežný manažment rizík
- sústredenie sa na verifikáciu a validáciu

Jedným z predpokladov klasického prístupu je zostavenie plánu v prvej fáze vývoja softvéru a následný vývoj, dalo by sa povedať, je pokusom o jeho dodržanie. Aj keď na začiatku je zostavený len hrubý plán, ktorý sa postupne zjemňuje. Plán však môže byť úspešný len v prípade, že sa pri jeho tvorbe vychádzalo z pravdivých údajov. Ale v rámci tvorby softvérového systému existuje až príliš veľa premenných.

V tak rýchlo bežiacom svete ako je ten dnešný nie je možné vopred povedať, čo bude o rok či dva, rovnako ako nie je možné zafixovať požiadavky klienta. Keďže softvér sa rozšíril do všetkých oblastí nie je možné predpokladať, že klient z inej oblasti bude presne a úplne vedieť čo vlastne chce a potrebuje. Softvérový vývojár zas nevie čítať myšlienky. Tu nastáva hneď prvý problém – dajú sa na začiatku sformulovať všetky požiadavky na koncový produkt? Mohlo by sa povedať, že realizované budú iba dohodnuté požiadavky alebo už v zmluve dohodnúť dobu oneskorenia odovzdania pri dodatočných požiadavkách. Ale zákazník softvérovej firmy je ako každý iný zákazník – má svoju predstavu, ktorú chce naplniť, a pritom chce vedieť kedy a za koľko. A keďže konkurencia nikdy nespí, treba sa snažiť, pokiaľ je to možné, vyjsť zákazníkovi v ústrety.

Práve v tomto bode sa podľa môjho názoru ukazuje slabá stránka klasického prístupu. Aj keď počiatočný plán mohol byť akokoľvek dokonalý, po niekoľkých zásahoch doň spôsobených novými požiadavkami alebo nepredvídanými komplikáciami, stáva sa nezrealizovateľný. Ak je nutné dodržať konečný termín odovzdania a došlo k oneskoreniu vo fáze napríklad implementácie, vedie to ku skráteniu času venovaného testovaniu, a to vedie k zníženiu kvality softvéru. Netreba zabúdať, že ak aj vynecháme časový aspekt, ostáva tu otázka nákladov. Zabudovanie novej funkcie do už vyvíjaného systému je dosť nákladné, keďže takáto požiadavka prichádza väčšinou až v neskorších fázach vývoja a je teda potrebné vrátiť sa aj k tým predošlým.

Nech žije sloboda

Agilné metódy predstavujú v porovnaní s plánovo riadenými slobodu, ale čiastočne i chaos. Oficiálne ich možno datovať od februára 2001, kedy bol vydaný manifest agilného vývoja. Je to vlastne zhrnutie základných princípov viacerých už dovtedy existujúcich metód. Manifest zásadne uprednostňoval [6]:

- samostatnosť a vzájomnú súčinnosť pred procesmi a nástrojmi
- fungujúci softvér pred rozsiahlou dokumentáciou
- tesnú spoluprácu so zákazníkom pred obchodnými rokovaniami
- ústretové reagovanie na zmeny pred bezvýhradným dodržiavaním plánu

Na prvom mieste pri riadení sa princípmi agilného vývoja softvéru je úžitková hodnota pre zákazníka. Z tohto pohľadu sú zmeny pozitívne, keďže túto hodnotu zvyšujú. Nezanedbateľným princípom je aj jednoduchosť softvéru nie však na úkor technickej dokonalosti. Kritériom úspešnosti softvéru je jeho funkčnosť nie dokumentácia, preto je dôraz kladený na krátke intervaly dodania čiastočného riešenia. Dôležitá je aj komunikácia, či už medzi zákazníkom a vývojárom alebo v rámci vývojového tímu. Vychádza sa z myšlienky, že dôvera v kombinácii s komunikáciou a motiváciou vedie ku kreativite a teda k lepším výsledkom. Tímy pracujú samostatne na vlastnom zlepšovaní a zvyšovaní efektivity.

Plánovanie pri agilnom prístupe sa na rozdiel od plánovo riadeného prístupu sústredí na nadchádzajúci cyklus iterácie. V rámci každej iterácie sú získavané požiadavky od zákazníka, je upravovaná architektúra systému, je implementovaná, otestovaná aj odovzdaná aplikácia. Pre zákazníka vyplýva z tohto postupu jasná výhoda – pokiaľ výsledok prvej iterácie nenapĺňa jeho predstavu, vie už oveľa konkrétnejšie povedať, čo mu chýba, a keďže projektový tím ráta so zapracovaním nových požiadaviek, je možné nakoniec do maximálnej možnej miery naplniť požiadavky klienta. Ďalšou nezanedbateľnou výhodou z pohľadu klienta je, že veľmi skoro dostáva do rúk aplikáciu, ktorá síce nenapĺňa úplne jeho požiadavky, ale už ju môže využívať. V dnešnom svete, kde dvojnásobne platí, že čas sú peniaze, je toto veľká výhoda. Pri klasickom prístupe klient uvidí aplikáciu až po fáze implementácie

a pokiaľ nenapĺňa jeho predstavu, je veľmi komplikované aplikáciu výraznejšie pozmeniť.

Ale agilný prístup má aj svoje slabosti. Samostatnosť si vyžaduje, aby každý člen tímu mal prehľad o celom projekte, čo je pri veľkých projektoch problematické. Podobne je na tom komunikácia. Tá sa dá veľmi dobre realizovať v menších tímoch, ale v tímoch, ktoré majú 100 členov, je to už problém. Iteračný spôsob vývoja softvéru nie je vhodný pre systémy určené napríklad pre atómové elektrárne, kde každá chyba systému môže byť katastrofická. Uprednostnenie fungujúceho softvéru pred rozsiahlou dokumentáciou je na jednej strane dobré pre klienta, na strane druhej sa však pozabúda na princíp znovupoužiteľnosti. K tomuto prispieva aj problém architektúry navrhovaného systému. Neustále zmeny nie je potrebné len implementovať, ale aj zapracovať do už existujúcej architektúry. Návrh architektúry systému vyvíjaného agilným postupom je tým pádom náročnejší ako pri klasickom plánovo riadenom.

Samostatnú kapitolu tvoria ľudia – členovia agilného tímu. Nároky na nich sú oveľa vyššie. Pre úspech celého projektu je potrebné, aby každý člen bol odborníkom, čo vyplýva z požiadavky samostatnosti. Keďže dôraz je kladený na komunikáciu v tíme, vývojári musia byť komunikatívni. Táto vlastnosť je častou požiadavkou pracovných ponúk a človek väčšinou nad ňou prejde s názorom, že však hovoriť vie. Bohužiaľ, z vlastnej skúsenosti viem, že hovoriť a povedať niečo je veľký rozdiel. I keď vývojári hovoria tou istou rečou, len niektorí vedia efektívne komunikovať medzi sebou. Kritickou požiadavkou je tímová spolupráca, na ktorej je založený celý agilný prístup.

Každý má svoj domov

Keďže oba prístupy majú svoje výhody a aj nevýhody, je zjavné, že ani jeden nie je Brooksovým strieborným nábojom [5]. Pri výbere konkrétneho postupu treba zvážiť oba a vybrať si ten, ktorého nevýhody sú menšie ako jeho výhody. Celkovo by sa dali podmienky nasadenia jednotlivých prístupov rozdeliť podľa tabuľky 1.

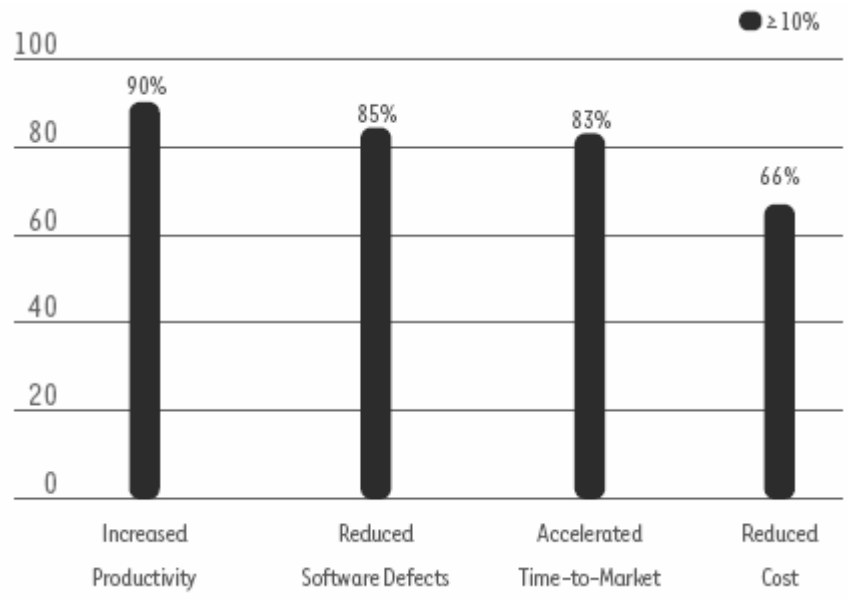
prístup	**agilný**	**plánovo riadený**
dôraz na	rýchlu hodnotu, zapracovanie zmien	stabilitu, predvídateľnosť, zabezpečenie
požiadavky	jednoduchší dizajn, zmeny lacné, skôr neformálny prístup	zložitejší dizajn, zmeny drahé, formalizovaný prístup
plánovanie	interné	zdokumentované
kontrola	kvalitatívna	kvantitatívna
zákazník	má záujem úzko spolupracovať a má na to schopnosti	stretnutia len v prípade potreby
vývojári	všetci odborníci	na rôznej úrovni
veľkosť tímu a projektu	menší	väčší

Tab. 1. Podmienky nasadenia jednotlivých prístupov.

Sloboda v reálnom svete

Agilný prístup síce zatiaľ v mojich očiach víťazí, ale keďže väčšina komerčných projektov, ktoré poznám, sú veľké, na prvý pohľad sa mi zdá problematické jeho aplikovanie pri ich vývoji. Posledné prieskumy však vyvracajú moje pochyby. Každoročne firma VersionOne v spolupráci s neziskovou organizáciou Agile Project Leadership Network (APLN) organizuje prieskum stavu agilného vývoja. Ich prieskum je jedným z najväčších v oblasti agilného vývoja softvéru, pričom sa ho zúčastňuje až 1700 respondentov zo 71 krajín [8]. Najdôležitejšie poznatky z tohtoročného prieskumu zverejneného v auguste 2007 sú:

- Tímy využívajúce agilný prístup sú čoraz väčšie. Až 31% respondentov pracuje v tímoch s viac ako 250 ľuďmi, 74% v tímoch nad 20 ľudí.
- Respondenti nahlásili viac ako 10% zlepšenie vo viacerých oblastiach (pozri Obr. 1):



Obr. 1. Zlepšenia pri využívaní agilných metód.[8]

- Dôvodom pre zavedenie agilného prístupu bola potreba zvládnuť meniace sa požiadavky (30% respondentov) a zrýchliť dodanie na trh (24% respondentov)

- Dve najväčšie prekážky pri zavádzaní agilného prístupu sú generálny odpor voči zmenám (36% respondentov, v roku 2006 len 20%) a deficit ľudí so skúsenosťami s agilným vývojom (34% respondentov, v roku 2006 len 21%).

Prieskumy ukazujú možnosť adaptácie agilného prístupu aj vo väčších tímoch a prakticky ukazujú jeho výhody. Namieste však je podľa mňa spomenúť aj najväčšie obavy pri jeho zavádzaní. Ich poradie v porovnaní s rokom 2006 zostalo zachované a je nasledovné:

1. nedostatočný plán
2. nedostatočná dokumentácia
3. strata manažérskej kontroly
4. chýbajúca možnosť predpovedania
5. nedostatok disciplíny vývojárov
6. nemožnosť škálovania

Klasický plánovo riadený prístupu disponuje väčšinou vecí, ktorých neprítomnosť pri agilnom prístupe spôsobuje obavy v organizáciách. Môj záver berúc do úvahy všetky výsledky prieskumu je jednoznačný. Napriek nevýhodám, ktoré nesie agilita so sebou, postupne víťazí v komerčnej sfére nad disciplínou klasického plánovo riadeného prístupu.

Agilita na akademickej pôde

Z môjho pohľadu sú zaujímavejšie možnosti nasadenia a prínosy agilného prístupu na akademickej pôde. Bolo vykonaných viacero experimentálnych pokusov na bakalárskom stupni štúdia na niektorých univerzitách [7]. V Reichlmayrovej štúdii organizovanej na Rochesterskom inštitúte technológie (RIT) išlo o vyskúšanie niektorých zásad agilného vývoja ako dôraz na komunikáciu a časté odovzdávanie výsledkov a nie o nasadenie konkrétnej metódy. Tímy boli zložené z 5 až 6 študentov na základe preferencií študentov. Dôvodom štúdie bol fakt, že pri využívaní klasického prístupu sa študenti nedostanú do kontaktu so štandardnými situáciami v rámci vývoja softvérového systému ako napríklad požiadavky na zmeny. Počas štúdie niektoré tímy neúspešne adaptovali princíp iteračného vývoja, keď implementovali aj časti budúcich iterácií a aj nepožadovanú funkcionalitu. Napriek týmto problémom boli celkové výsledky úspešné, keďže študenti získali skúsenosti so zapracovaním zmien a sebareflekciou.

Ďalšia štúdia bola realizovaná na univerzite v Montreali Germainom a Robillardom. Študenti dostali špecifikáciu, na základe ktorej mali vyvinúť softvérový systém do 45 dní. Porovnávané boli výsledky tímov pracujúcich pomocou RUP (väčšinou považovaná za plánovo riadenú metódu) a tímov využívajúcich extrémne programovanie XP (agilná metóda). XP tímy strávili viac času implementáciou, testovaním a integrovaním, kým RUP tímy viac času venovali predimplementačným aktivitám, ktoré však, ako ukázala štúdia, nezredukovali snahu vynaloženú na implementáciu. Iba malé rozdiely v konečnom výsledku možno vysvetliť

medzitímovou komunikáciou, kde RUP tímy získali informácie od XP tímov, ktoré začali implementovať skôr.

Na univerzite Heriot-Watt bola zorganizovaná štúdia na porovnanie funkcionality s používateľskými požiadavkami na strane jednej a vynaloženým úsilím na strane druhej. Pre minimalizovanie skreslenia údajov boli tímy zostavené rovnomerne podľa predchádzajúcich študijných výsledkov. Agilné skupiny pracovali v dvojtýždňových cykloch so zákazníckou interakciou, kým plánovo riadené tímy pracovali v 6-8 týždňových cykloch. Napriek snahe prísne dodržať zvolený prístup, nedalo sa zamedziť medzitímovým kontaktom. Organizátori preto zvolili ako pomocný nástroj dotazníky na zistenie koľko metód typických pre agilný prístup využili plánovo riadené tímy. Výsledky tejto štúdie boli pre organizátorov prekvapivé, pre mňa však nie. Mnohé plánovo riadené tímy sa nakoniec ukázali viac agilnými ako tie, ktoré agilne postupovať mali. Agilné tímy na začiatku víťazili v objeme odovzdanej funkcionality, čo ukazuje, že aj prvé iterácie majú význam. Na konci však neexistovali väčšie rozdiely medzi oboma skupinami. Jeden agilný tím dosiahol výborné výsledky v porovnaní získaných funkcií a vynaloženého úsilia, ale druhý dosiahol veľmi slabé výsledky aj v porovnaní s plánovo riadenými tímami. Organizátori si tieto výsledky vysvetľujú medzitímovou komunikáciou, ako i faktom, že všetci študenti boli oboznámení s agilnými metódami. Vysvetlením by mohli byť aj Hirschove slová[7]: „Na agilný vývoj sa najlepšie pozerá ako na evolúciu plánovo riadeného vývoja a nie ako na radikálny odchod od predchádzajúcich praktík.“

Agilita a tímový projekt

Môj prvotný pozitívny názor o výhodách agilného prístupu bol podporený nielen prieskumami v komerčnej oblasti ale aj štúdiami v oblasti akademickej. Keďže z môjho pohľadu je výhodné jeho použitie alebo aspoň využitie niektorých jeho princípov, treba sa zamyslieť nad agilným riešením školského tímového projektu.

Školský tímový projekt sa vyznačuje vopred danými termínmi odovzdania a nutnosťou rozsiahlej dokumentácie. Z tohto pohľadu je nutné aplikovať klasický plánovo riadený prístup. Ale, inšpirujúc sa štúdiami organizovanými na rôznych univerzitách, predsa len vidím priestor pre využitie princípov agilného vývoja. Tímy v rámci tímového projektu sú 5-6 členné, čím napĺňajú kritériá agilného prístupu, čo sa týka veľkosti tímu a tým pádom komunikácie. Existencia zdokumentovaného plánu skombinovaná s intenzívnou komunikáciou v rámci tímu by mohla napomôcť celkovému výsledku projektu. Zákazníka by mohol predstavovať vedúci projektu, takže časté konzultovanie namiesto len občasného oboznámenia s doterajšími výsledkami projektu je adaptáciou ďalšieho agilného princípu. Štruktúra tímového projektu predpokladá odovzdanie prototypu na konci zimného semestra a treba očakávať požiadavky na zmeny. Príprava na existenciu nových požiadaviek by tiež mohla uľahčiť ich zapracovanie a tak prispieť k hodnotnejšiemu výsledku.

Takýto prístup k vývoju softvéru by sa dal označiť ako hybridný. Snaží sa zobrať si to najlepšie z oboch existujúcich a zároveň eliminovať ich nevýhody. Otázna je jeho úspešnosť. Verím, že aj v prípade neúspechu by nemalo dôjsť k ohrozeniu

naplnenia projektu, keďže navrhnuté vylepšenia využívali aj tímy počas už zrealizovaných štúdií. Zároveň aj respondenti z komerčnej oblasti v prieskume priznali využívanie hybridných prístupov, i keď išlo len o relatívne malé percento z nich [8].

Záver

V eseji som sa zamerala na najčastejšie využívané prístupy pri vývoji softvérových systémov – klasický plánovo riadený a agilný. Na úvod som sa snažila oboznámiť čitateľa s ich základnými princípmi. Využila som svoje skúsenosti na určenie ich slabín, a zároveň som sa zamyslela nad ich možným odstránením. Keďže počiatočné podmienky agilného prístupu vo mne vyvolali pochyby ohľadom ich nasadenia a úspechu v komerčnej oblasti, preštudovala som aj prieskumy, ktoré moje pochybnosti vyvrátili. Naopak úspech v komerčnej oblasti ma donútil sa zamyslieť nad využitím agilných metód aj v školskom projekte. Nápomocné sa ukázali byť už zrealizované štúdie, ktoré mi pomohli pri vytvorení a popísaní predstavy konkrétnej aplikácie agilného prístupu pri riešení tímového projektu. Svoju predstavu plánujem aj aplikovať a na základe takto získaných praktických znalostí môžem predstavu rozvinúť alebo úplne zavrhnúť.

Použitá literatúra

1. Bieliková, Mária: Agilné metódy vývoja softvéru. SOFTECON 2004: Odborná konferencia o víziách a trendoch v moderných informačných technológiách, 24.2.2004. [citované 24.10.2007] Dostupné z < www.softec.sk/files/Softecon/Softecon2004/SOFTECON04-Bielik.pdf>.
2. Boehm, Barry: A View of 20th and 21st Century Software Engineering, In: Proceeding of the 28th international conference on Software engineering ICSE '06, ACM Press, máj 2006, s. 12 – 29.
3. Boehm, Barry – Turner, Richard: Balancing Agility and Discipline: A Guide for the Perplexed. Boston, MA, Addison Wesley. 2003. ISBN 0-321-18612-5
4. Boehm, Barry – Turner, Richard: Observations on Balancing Discipline and Agility, In: Proceedings of the Conference on Agile Development (ADC '03), jún 2003, s. 32.
5. Brooks, F.P.: The Mythical Man-Month: Essays on Software Engineering. Anniversary Edition. Addison Wesley. 1995.
6. Kolektív autorov: Manifesto for Agile Software Development.[citované 24.10.2007] Dostupné z < http://www.agilemanifesto.org/>.
7. Rundle, P.J. – Dewar, R.G.: Using Return on Investment to Compare Agile and Plan-Driven Practices in Undergraduate Group Projects. In: Proceeding of the 28th international conference on Software engineering, SESSION: Education papers: advanced topics in software engineering education, ACM Press, 2006, s. 649 – 654.

8. VersionOne: 2nd Annual Survey: "The State of Agile Development". Jún – júl 2007. Dostupné z <http://www.versionone.com/agilesurvey/>.
9. Víšek,Jozef – Kišoňová, Eva: Agilný vývoj software v distribuovaných projektoch, Konferencia Systémová integrácia 2007, Štrbské pleso, 27.9.2007. [citované 24.10.2007] Dostupné z < www.sssi.sk/download/si2007/prednasky/SI2007_**V**isek.pdf >.

Annotation

Discipline versus agility

Today's software systems are becoming bigger and more complex and therefore planning is an inseparable part of their development. The goal of planning is to better understand the goal of the project, make the development process more effective, reduce uncertainty and build a base for project management. In the process of preparation of a plan it is important to know the global approach to software development and his lifecycle. It is necessary to know ahead which tasks need to be planned and in which order. This essay is concentrated on the two most used approaches to software development: plan-driven and agile. Continuous changes in software requirements compelled the existence of a new approach. The question is how the old approach can be applied in changed environment of the 21. century. At the beginning the main principles of both approaches are explained. On the bottom of actual surveys in commercial and academic sphere the areas of their usage, their advantages and disadvantages are compared. As none of the approaches is the famous Brooks silver bullet, a hybrid approach is examined in concern to school team project. This hybrid approach should have all the advantages and eliminate their disadvantages.