Citace článku:
BUCHALCEVOVÁ, Alena. Jaký má být dnes vývoj softwaru – business driven, test driven, model driven, architecture driven nebo service oriented? Monínec 15.05.2005 – 18.05.2005. In: RUDOLF, Vladimír, FELBÁB, Jiří (ed.). XXVI. konference EurOpen. Plzeň : EurOpen, 2005, s. 5–16. ISBN 80-86583-08-2.

Jaký má být dnes vývoj softwaru – business driven, test driven, model driven, architecture driven nebo service oriented ?

Alena Buchalcevová

Abstrakt

Obrovsky rychlý vývoj hardwaru a síťových technologií jde ruku v ruce s vývojem základního softwaru, technologií middlewaru, programovacích jazyků a vývojových prostředí. Významné změny nastávají i v přístupech k procesům vývoje softwaru. Příspěvek se zaměřuje na nejvýznamnější trendy, přístupy a metodiky, které v dnešní době hýbou vývojem softwaru, a které se snaží najít cestu k větší úspěšnosti softwarových projektů a identifikovat přínosy zavedení IS/ICT v podnikání organizace.

1 Nejvýznamnější trendy v oblasti IS/ICT

Články, průzkumy a předpovědi společnosti Gartner patří mezi nejvýznamnější zdroje, které monitorují směry vývoje v různých oblastech, oblast IS/ICT nevyjímaje. Proto jsem při přípravě tohoto příspěvku vycházela z předpovědi společnosti Gartner pro rok 2005 „Predicts 2005: Deploy New Technology, Applications for Success“. Velmi potěšující bylo zjištění, že se tyto předpovědi nejdůležitějších trendů pro rok 2005 shodují s tématy jarní konference EurOpen. Společnost Gartner uvádí tyto nejdůležitější technologické trendy [Gartner,2004]:

- Linux a Open Source software bude stále více prorůstat do podnikové infrastruktury,
- osobní počítače se budou stále více virtualizovat, tj. počítačové zdroje se budou jak hardwarově, tak softwarově rozdělovat na části (partitioning), budou sdíleny a simulovány,
- další masový rozvoj mobilních a bezdrátových technologií,
- bezpečnost se zaměří zejména na prevenci průniků a útoků,
- ještě důraznější bude prosazování hesla „anytime anywhere“,
- boj mezi platformami J2EE a Microsoft .NET skončí provozováním obou platforem a důrazem na jejich integraci.

V oblasti aplikací Gartner předpovídá tyto přístupy k vývoji softwaru:

- pokračující snaha o snižování nákladů povede k růstu offshore outsourcingu a outsourcingu byznys procesů,
- webové služby, které se v mnoha odvětvích a organizacích dostávají za hranice pilotních projektů, budou v roce 2005 hlavním trendem,
- společnosti budou investovat do nástrojů na podporu sdílení znalostí.

1.1 Sladění IS/ICT s podnikovými procesy

Klíčovým trendem, který jde za rámec IS/ICT, je požadavek na sladění IS/ICT s podnikáním. Tento požadavek se objevuje již několik let mezi prioritami v oblasti IS/ICT. Přehled pěti nejdůležitějších priorit v oblasti IS/ICT podle 2003 Worldwide IT Benchmark Report společnosti Meta Group [Metagroup,2002] pro respondenty z USA a z ostatních zemí světa uvádí tabulka 1.

5 nejvyšších priorit pro USA	**5 nejvyšších priorit pro ostatní svět**
1. snížení nákladů	1. zvýšení produktivity
2. business alignment	2. snížení nákladů
3. zvýšení produktivity	**3. business alignment**
4. řízení projektu	4. zlepšení SW procesů
5. zlepšení kvality SW	5. zlepšení kvality SW

tabulka 1: Přehled pěti nejdůležitějších priorit v oblasti IS/ICT

Business alignment je odrazem současné ekonomické recese, která významně ovlivňuje i oblast IS/ICT. Výdaje na informatiku se snižují, požaduje se vyšší kvalita, kratší čas a přínosy IS/ICT v podnikání. Právě sladění IS/ICT s podnikovými procesy má být řešením tohoto problému. Jde o to, aby čas a prostředky vynaložené na projekty v oblasti IS/ICT měly přímý vliv na podnikání. IS/ICT je třeba spojit s podnikovými procesy a hodnotit úspěch zavedení IS/ICT podle toho, jak se projeví v podnikání organizace. Nástrojem, jak toho dosáhnout, se zdá být koncept služeb. Služby vystupují jako spojovací článek mezi podnikovými procesy a IS/ICT. Společnost Meta Group je přesvědčena, že jedním z nejdůležitějších trendů v informatice v následující dekádě bude právě orientace na služby [Metagroup,2003]. Otázka orientace na služby při vývoji IS/ICT není jen otázkou technologie, ale podstatně ovlivňuje i metodiky vývoje a nasazení IS/ICT. Ačkoli se technologie i ekonomické prostředí za poslední desetiletí dramaticky změnily, metodiky

většinou s těmito změnami nepočítají. Většina metodik byla navržena pro prostředí, kde se aplikace vyvíjejí od začátku, ne pro prostředí založené na službách, kdy je třeba řešení poskládat z různých zdrojů.

Architektura orientovaná na služby je postavena na těchto klíčových principech [Bloomberg,2003]:

- podnikové procesy jsou určující pro služby, které jsou zase určující pro technologii
- služby musí umožňovat agilitu podnikových procesů
- úspěšná Architektura orientovaná na služby se stále vyvíjí.

Atraktivnost služeb spočívá ve zvýšení produktivity IS/ICT řešení, snížení nákladů vývoje a nasazení a zkrácení času uvedení na trh. Dalším cílem je vyšší přínos z IS/ICT řešení. Architektura orientovaná na služby je důležitá pro podniky, protože představuje rámec, který sjednocuje byznys model s technologiemi a realizuje funkcionalitu zajišťující efektivní podnikání.

2 Současný stav v oblasti vývoje IS/ICT

Úspěšnost projektů vývoje informačních systémů není uspokojivá. Podle výsledků výzkumu společnosti Standish Group splňovalo kritéria úspěšnosti v roce 2000 jen 28% všech projektů na vývoj aplikací. Tato kritéria byla definována jako dokončení projektu včas, se všemi specifikovanými funkcemi a za daných nákladů [Johnson,2001]. Příčin této skutečnosti je celá řada. Mezi nejvýznamnější bych zařadila:

- turbulentní změny ve společnosti,
- velmi rychlý vývoj informačních a komunikačních technologií,
- složitost vývoje softwaru,
- tlak na rychlost vývoje,
- železný trojúhelník,
- potřeba integrace se stávajícími systémy.

2.1 Turbulentní změny ve společnosti

Současná ekonomika je stále více orientovaná na znalosti, je charakterizována kratším životním cyklem produktů, důrazem na inovace a rychlým technologickým pokrokem. Svět kolem nás se mění a je třeba na změny nejen reagovat, ale dokonce změny vyvolávat, a tak získat náskok před konkurencí. Z pohledu IS/ICT to znamená, že jak při vývoji IS, tak při jeho provozu je třeba do systému promítat změny.

2.2 Velmi rychlý vývoj informačních a komunikačních technologií

Změny, které probíhají v celé společnosti, jsou ještě výraznější v oblasti informačních a komunikačních technologií, neboť právě tato oblast patří dnes k nejdynamičtějším. Obrovsky rychlý vývoj hardwaru a síťových technologií jde ruku v ruce s vývojem základního softwaru, technologií middlewaru, programovacích jazyků a vývojových prostředí. Tento rychlý vývoj nám na jedné straně poskytuje stále lepší prostředky, na druhé straně ale představuje závažný problém, pokud se zabýváme otázkami jako je uchování investic vložených do IS/ICT, kvalifikace vývojářů softwaru a podobnými. Právě na řešení těchto otázek je zaměřena iniciativa organizace OMG Modelem řízená architektura (Model Driven Architecture) MDA. MDA vychází ze skutečnosti, že množství změn v systému klesá s postupem na vyšší úrovně abstrakce. Dopady neustálých změn technologií je možné omezit jen na část modelu – na jeho nižší vrstvy. Při MDA vývoji se nejprve vytvoří Platformově nezávislý model (Platform Independent Model – PIM), který reprezentuje věcnou funkcionalitu a chování systému. Pomocí MDA nástrojů se PIM mapuje na zvolenou platformu (například Corba, Java/EJB, XML/SOAP) a generuje se Platformově specifický model (Platform Specific Model – PSM). Nakonec se generuje implementační kód pro příslušnou technologii. MDA nástroje umožňují také zpětné inženýrství (reverse engineering), a tak je možné vytvořit modely stávajících systémů pro účely integrace aplikací. MDA generátory aplikují současně i návrhové vzory [Buchalcevová,2003].

2.3 Složitost vývoje softwaru

Na celou historii vývoje softwaru, která není ve srovnání s ostatními odvětvími dlouhá, můžeme pohlížet jako na boj se složitostí. Na jedné straně se do tohoto boje nasazují stále výkonnější nástroje, na druhé straně rostou požadavky na software (rozsah, kvalita, rychlost vývoje, flexibilita, přívětivost a další). Hlavním atributem softwaru je tedy stále složitost jeho vývoje, která je také jednou z příčin velkého počtu neúspěšných softwarových projektů. Na vývoj softwaru má vliv jak prostředí vývoje, tak cílové prostředí. Proměnnými veličinami při vývoji softwaru jsou dle [Scrum,1995]:

- dostupnost kvalifikovaných specialistů (pro nové technologie, nástroje, metody a domény je malý počet kvalifikovaných odborníků),
- stabilita technologie pro implementaci (nové technologie jsou méně stabilní),
- stabilita a schopnosti nástrojů,
- efektivnost používaných metod,
- dostupnost expertů na věcnou oblast i technologii,

- nová funkcionalita a její vztah k existující funkcionalitě,
- metodika a její flexibilita,
- konkurence,
- čas,
- zdroje,
- další proměnné.

Celková složitost vývoje softwaru je funkcí těchto proměnných, přičemž tyto proměnné se v průběhu projektu mění.

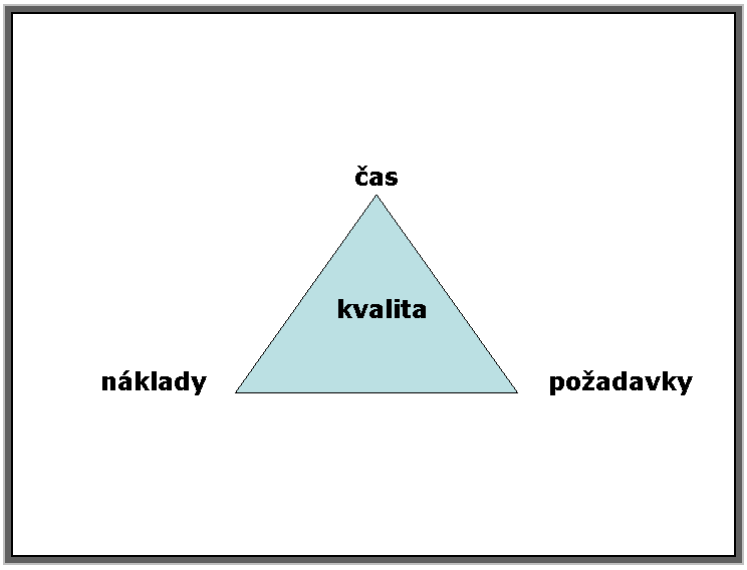
Software má mnoho aspektů, které jej odlišují od jiných produktů, a proto je i proces jeho vývoje odlišný. Tradiční přístupy předpokládají, že procesy při vývoji softwaru je možné plně definovat a konzistentně opakovat. To předpokládá, že je možné definovat a opakovat: problém, řešení, nositele řešení (vývojáře) a prostředí. Tyto předpoklady však podle zastánců agilních přístupů při vývoji softwaru neplatí. V mnoha případech není možné definovat problém na začátku projektu, protože požadavky nejsou přesně specifikovány a nebo se mění. Opakovatelnost řešení předpokládá, že je možné plně specifikovat architekturu a návrh. Také vývojáři nejsou stejní, ale liší se svými schopnostmi a znalostmi. Liší se i prostředí, ve kterém vývoj probíhá. Vývoj softwaru tak probíhá v podmínkách chaosu a je to velmi složitý proces, který nelze předem plně popsat, ale je nutné jej průběžně monitorovat a přizpůsobovat se změnám.

2.4 Tlak na rychlost vývoje

Turbulentní ekonomické prostředí a vysoká konkurence v odvětvích vedou k tomu, že je třeba realizovat změny rychle. A jestliže je software klíčovým faktorem fungování organizací, pak je třeba také software vytvořit a zavést velmi rychle, dříve než to udělá konkurence. To vystihuje termín „time to market“, který je klíčovým požadavkem při dnešním vývoji a nasazování softwaru.

2.5 Železný trojúhelník

Softwarový projekt je omezen železným trojúhelníkem (obrázek 2). Pokud je přesně definován plán (čas), rozpočet (náklady) a rozsah vytvářeného softwarového produktu (požadavky), nemá řešitelský tým žádný manévrovací prostor, a tak spěje k selhání. Jediné, co se pak může měnit, je kvalita produktu.



Obrázek 1: Železný trojúhelník

První bod železného trojúhelníku odpovídá na otázku „Jak dlouho bude vývoj trvat ?“ Nerealisticky těsný plán je známým problémem v řadě projektů. Stejně problematický je ale i několikaletý plán bez interních dodávek, který negarantuje prakticky žádný pokrok. Řešením je realizovat projekt v krátkých iteracích. Primárním produktem musí být fungující software, který splňuje v daném čase nejdůležitější požadavky. Druhý bod trojúhelníku odpovídá na otázku „Co bude vývoj stát ?“ Zdroje jsou v projektech většinou špatně řízeny, velmi brzy jsou zařazeny do určitých kategorií a jejich výše bývá na začátku často stanovována uměle. Třetí bod trojúhelníku odpovídá na otázku „Co dostaneme?“ Při tradičním způsobu vývoje se tým snaží definovat většinu požadavků brzy. Tento přístup ale nezohledňuje fakt, že požadavky se mění s tím, jak se vyvíjí znalosti a prostředí. Ve středu železného trojúhelníku je kvalita. Ta řeší otázku „Bude výsledek dost dobrý (kvalitní)?“ Kontroly, které mají odhalovat defekty, jsou tradičním způsobem garance kvality. Ale je to nejlepší přístup? Místo vyhledávání chyb po dodání, by bylo lepší vytvořit bezchybný produkt. Standardy kódování a pravidla modelování pomáhají v dosahování kvality stejně jako vývoj s testováním na začátku a aktivní účast investorů.

Jak se tedy vypořádat se železným trojúhelníkem? Řešením je asi dosáhnout rozumného kompromisu. Nejlepší manažeři si uvědomují, že pro zajištění úspěchu projektu je třeba vzít v úvahu železný trojúhelník a realizovat změny, pokud nastanou. Watts Humprey, jeden z tvůrců CMM řekl: „To, co lidé skutečně chtějí, je vysoce kvalitní systém, který implementuje vše, co chtějí, při nulových nákladech a hned. Vše ostatní je jen dohoda, kompromis.“

2.6 Potřeba integrace se stávajícími systémy

Informační systémy dnes nevznikají na zelené louce. Většina organizací má již automatizovány hlavní oblasti své činnosti. Požadavkem dnešní doby je integrace těchto „ostrůvků“ automatizace do jediného systému. Zatímco dříve byly informační systémy nahrazovány novými, dnes se začíná prosazovat názor, že je třeba stávající systémy, které dobře pracují, propojit s ostatními a zpřístupnit je například přes webové rozhraní. To je jeden z úkolů systémové integrace. Systémová integrace představuje způsob, jak umožnit technologiím ve stále se měnícím informatickém prostředí spolu komunikovat. Jak řekl Richard Soley, předseda a CEO OMG[1], ve své úvodní přednášce na OMG konferenci Integrate 2003: „Integrace je problém, který nemůžeme odsunout.“ [Soley,2000]. Integrace není jen technologický problém, ale je to také obchodní problém. S požadavkem integrace systémů úzce souvisí nutnost celopodnikového (Enterprise) pohledu.

3 Metodiky

Složitost tvorby informačních systémů, jejíž příčiny jsem nastínila v předcházející kapitole, se již dlouhou dobu snaží řešit metodiky vývoje informačních systémů. Význam metodik pro vývoj informačního systému dokumentují například výsledky prezentované ve zprávě „2003 Worldwide IT Benchmark Report“ společnosti META Group [Metagroup,2003], v níž se uvádí, že 51,6% všech respondentů používá při vývoji informačních systémů metodiku.

3.1 Pojem metodika

Metodika (methodology) představuje v obecném smyslu souhrn metod a postupů pro realizaci určitého úkolu. **Metodika vývoje a údržby IS/ICT definuje principy, procesy, praktiky, role, techniky, nástroje a produkty používané při vývoji, údržbě a provozu informačního systému, a to jak z hlediska softwarově inženýrského, tak z hlediska řízení**.[Buchalcevová,2005]

Kromě pojmu metodika se můžeme setkat s pojmy proces a softwarový proces. Mnohé metodiky mají slovo „proces“ přímo ve svém názvu. Příkladem jsou metodiky Rational Unified Process, Open Process, Object-Oriented Software Process a další. Existují metodiky hodnocení softwarových procesů (například Capability Maturity Model), mluví se o zlepšování softwarových procesů. Softwarový proces je v kontextu těchto přístupů definován jako sada činností, metod, praktik a transformací, které lidé používají pro vývoj

[1] standardizační organizace Object Management Group

a údržbu softwaru a dalších s tím spojených produktů (projektových plánů, návrhů, testovacích případů apod.) [Paulk,1993].

3.2 Stav v oblasti metodik v ČR a ve světě

Metodik, které se zabývají vývojem informačních systémů, je velké množství. Tato skutečnost má objektivní důvody, kterými jsou mimo jiné:

- různé technologie a paradigmata vyžadují různé techniky (strukturované , objektové),
- organizace se liší firemní kulturou,
- každý jedinec je jedinečný á má jiný styl uvažování, práce**,**
- každý tým je jedinečný,
- projekty se liší velikostí týmu,
- projekty se liší důležitostí.

Důležité je, aby metodiky pro vývoj a údržbu IS/ICT byly jednotně popsány a kategorizovány. Charakteristiku předních současných metodik, jejich popis v jednotné struktuře a kategorizaci je možné nalézt v [Buchalcevová,2005]. Zařazení metodik do dobře definovaných kategorií je velmi důležité, neboť metodiky nejsou srovnatelné v řadě hledisek. Většinou se jedná o metodiky zaměřené jen na určitou fázi vývoje informačního systému, na určitou věcnou oblast, na určitý typ projektu a podobně. Uvedená publikace se snaží také definovat kritéria, jak vybrat vhodnou metodiku pro určitý typ projektu, a postupy pro její přizpůsobení na konkrétní podmínky firmy a projektu.

Problém většiny metodik je také v tom, že se zaměřují na vývoj nového informačního systému. Přitom v dnešní době je hlavním úkolem zejména rozvoj stávajících systémů, implementace typových programových řešení, integrace dílčích řešení do celopodnikového systému.

Podíl firem, které používají při vývoji software formální metodiku, je v České republice nižší než ve světě. Tato skutečnost může mít řadu důvodů, ale jsem přesvědčena, že mezi hlavní důvody patří:

- nedostatek českých metodik[2], neboť většina metodik je v angličtině a nejsou lokalizovány do češtiny,
- metodiky se zpravidla šíří na komerční bázi a české firmy nechtějí či nemohou vydávat prostředky na nákup metodik,

[2] z českých metodik můžeme uvést například metodiky *Objektově orientované metodiky a technologie* (OOMT) viz [Drbal,1997], *Multidimensional Management and Development of Information System* (MMDIS) [Vorisek,1997], *Business Object Relation Modeling* (BORM) viz [Polák,Merunka,Carda,2003]

3.3 Rigorózní metodiky a agilní metodiky

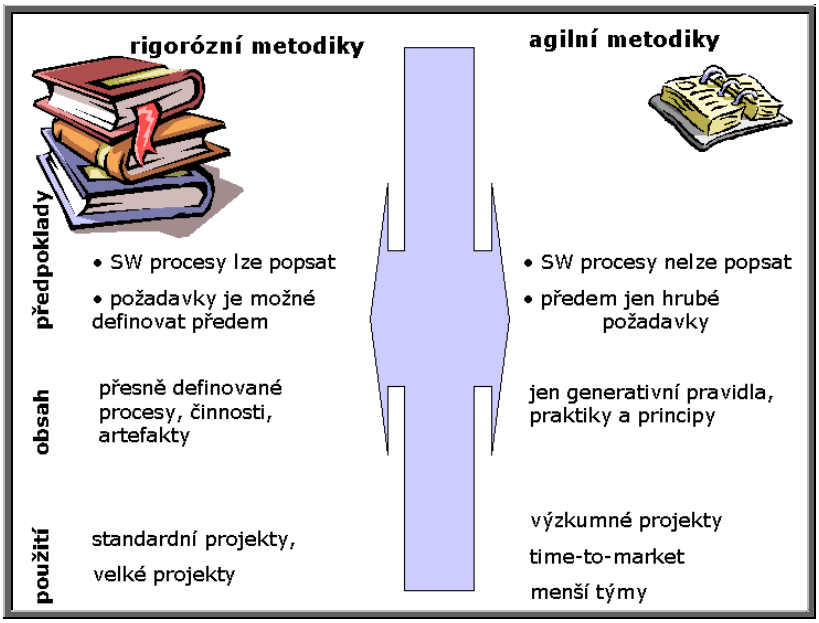
V současnosti můžeme sledovat dva hlavní proudy v metodických přístupech, které jsou označovány jako rigorózní metodiky a agilní metodiky. Rigorózní metodiky vycházejí z přesvědčení, že budování IS/ICT lze popsat, plánovat, řídit a měřit. Snaží se podrobně a přesně definovat procesy, činnosti a vytvářené produkty, a proto bývají často velmi objemné. Rigorózní metodiky jsou zpravidla založeny na sériovém (vodopádovém) vývoji. Při tomto způsobu vývoje probíhají jednotlivé fáze jako plánování, analýza, návrh, implementace, zavedení sekvenčně za sebou. Existují ale také rigorózní metodiky založené na iterativním a inkrementálním vývoji3 Příkladem těchto metodik jsou OPEN, Rational Unified Process (RUP), Enterprise Unified Process (EUP). V rámci rigorózních metodik tvoří samostatnou kategorii metodiky pro hodnocení softwarových procesů (Software Process Assesment). Jsou realizovány zejména v rámci projektu SPICE, který představuje hlavní mezinárodní iniciativu pro podporu vývoje mezinárodního standardu pro hodnocení softwarových procesů. Metodiky hodnocení softwarových procesů jsou založeny na přesvědčení, že kvalita procesu určuje kvalitu produktu, a proto popisují postupy, které umožňují hodnotit úroveň zralosti procesů při vývoji software. Nejznámější z těchto metodik je Model zralosti (Capability Maturity Model)

Změny technologií a ekonomického prostředí, ke kterým v současnosti dochází, a požadavky na rychlé zavedení IS/ICT vyžadují změny v metodikách. Tradiční rigorózní metodiky přestávají v takových podmínkách vyhovovat a začínají se prosazovat metodiky, které umožňují vytvořit řešení velmi rychle a pružně jej přizpůsobovat měnícím se požadavkům. Tyto metodiky jsou označovány jako agilní. Jedná se o různé metodiky, které vznikaly od druhé poloviny 90. let a které prosazují myšlenku, že jedinou cestou, jak prověřit správnost navrženého systému, je vyvinout produkt, nebo jeho část, co nejrychleji, předložit ho zákazníkovi a na základě zpětné vazby upravit. Každá z agilních metodik je svým způsobem specifická, ale všechny jsou postaveny na stejných principech a hodnotách. Proto se sešli představitelé těchto přístupů v únoru 2001, podepsali „Manifest agilního vývoje software“ [Fowler,Highsmith,2001] a vytvořili „Alianci pro agilní vývoj software“[AgileAlliance,2003].

[3] Iterativní vývoj představuje opakované (iterativní) provádění jednotlivých fází při vývoji IS. Výsledkem každé iterace je funkční verze systému. Současné metodiky doporučují velmi krátké iterace (dny). Iterativní vývoj může probíhat buď pro celý systém, jehož funkčnost se v jednotlivých iteracích rozšiřuje, a nebo ve spojení s inkrementálním vývojem (systém se vyvíjí po přírůstcích). [KIT,2003]

Agilní metodiky představují ve své podstatě reengineering procesů při budování IS/ICT. Když byl v 90. letech prosazován reengineering v podnikových procesech, dostala jedna osoba (vlastník procesu) plnou kontrolu nad celým procesem. Podobně agilní přístup k vývoji softwaru dává jednomu vývojáři plnou kontrolu nad všemi fázemi procesu vývoje – od přímé komunikace se zákazníkem při sběru požadavků až k realizaci.

Rigorózní a agilní metodiky představují dvě skupiny metodik, které vycházejí z odlišných předpokladů a odlišného pohledu na vývoj software. Výsledkem je jiný obsah a zaměření každé kategorie metodik a jiný okruh projektů, na které je vhodné tyto metodiky aplikovat. Odlišnosti obou přístupů jsou přehledně zachyceny na obrázku 2.



Obrázek 2: Srovnání rigorózních a agilních metodik [Buchalcevová,2005]

I když jsou východiska, obsah, přístupy i použití rigorózních a agilních metodik na první pohled velmi rozdílné a jejich zastánci vystupují zpravidla antagonisticky, je možné oba přístupy určitým způsobem kombinovat. Rigorózní metodiky je možné odlehčit a aplikovat v jejich rámci některý z agilních přístupů. Velmi zdařilý popis aplikace základních principů agilních metodik v metodice RUP je možné nalézt v [Kroll,2001]. Dalším příkladem propojování rigorózních a agilních metodik je aplikace metodiky Agilní modelování v RUP, která je popsána v [Ambler,2001]. Na druhé straně, pokud potřebujeme použít agilní metodiky na větší projekty či projekty větší důležitosti, je třeba je více formalizovat, zařadit více dokumentace apod. Agilní metodiky jsou v převážné většině zaměřeny na vývoj nového řešení. V poslední době se objevuje snaha aplikovat agilní přístupy i na úpravy řešení a integraci řešení a stejně tak na některé metodiky patřící do kategorie globálních metodik.

4 Závěr

Příspěvek podává obraz současného stavu v oblasti IS/ICT a nejdůležitějších trendů. Je příznačné, že všem těmto tématům jsou věnovány příspěvky na jarní konferenci EurOpen 2005.

Literatura

[AgileAlliance, 2003] http://www.agilealliance.org/articles/index

[Allen,2002] Allen, P.: *The OMG'S Model Driven Architecture, component development strategies*, Cutter Information Corp., January 2002. Dostupný z WWW: http://www.cutter.com/articles.html

[Allen,2003] Allen, P.: *Service–Oriented Architecture Concepts*, Expertś corner LogOn, 6/2003. Dostupný z WWW: http://www.ltt.de/cgi-bin/down/download.pl?download/experts/allen-07.03.pdf

[Ambler,2001] Ambler, S.: *Agile Modeling and the Unified Process*, Agile Modeling, 2001. Dostupný z WWW: http://www.agilemodeling.com/ essays/agileModelingRUP.htm

[Ambler,2002] Ambler, S.W.: *Agile Software Development*, 2002. Dostupný z WWW: http://www.agilemodeling.com/essays/agileSoftwareDevelopment.htm

[Beck,2002] Beck, K.: Extrémní programování, Grada, 2002, ISBN 80-247-0300-9

[Bloomberg, 2003] Bloomberg, J.: *Principles of SOA*, Application Development Trends March 2003. Dostupný z WWW: http://www.adtmag.com/article.asp?id=7345

[Buchalcevová, 2002] Buchalcevová, A.: *Agilní metodiky*, In: Objekty 2002, ČZU Praha, 2002, ISBN 80-213-0947-4.

[Buchalcevová, 2003] Buchalcevová, A.: *Model Driven Architecture jako nový přístup k vývoji i integraci aplikací*, In: Systémová integrace 2003, s.469-476, ISBN 80-245-0522-3.

[Buchalcevová, 2005] Buchalcevová, A.: *Metodiky vývoje a údržby informačních systémů*, Grada publishing, 2005, ISBN 80-247-1075-7

[CMMI,2002] *Capability Maturity Model® Integration (CMMISM) – Version 1.1 – Staged Representation*, Technical Report CMU/SEI-2002-TR-012, The Software Engineering Institute, 2002. Dostupný z WWW: http://www.sei.cmu.edu/cmmi/

[Cockburn,1998] Cockburn, A.: *The Methodology Space*, 1998. Dostupný z WWW: http://alistair.cockburn.us/crystal/articles/ms/methodologyspace.htm

[Cockburn,1999] Cockburn, A.: *A Methodology Per Project*, Humans and Technology Technical Report, TR 99.04,1999 Dostupný z WWW: http:// crystalmethodologies.org/articles/mpp/methodologyperproject.html

[Drbal,1997] Drbal, P. a spolupracovníci: *Objektově orientované metodiky a metodologie*, skripta VŠE, 1997, ISBN: 80-7079-740-1.

[Fowler, Highsmith, 2001] Fowler, M– Highsmith, J.: *The Agile Manifesto*, Software Development, August 2001. Dostupný z WWW: http://www.sdmagazine.com/ documents/sdm0108a/

[Gamma,2003] Gamma, E.– Helm, R.– Johnson, R.– Vlissides, J.: *Návrh programů pomocí vzorů, Stavební kameny objektově orientovaných programů*, překlad anglického originálu, Grada, Praha 2003, ISBN 80-247-0302-5.

[Gartner,2004] Predicts 2005: Deploy New Technology, Applications for Success

Dostupný z WWW: http://www.gartner.com/resources/124700/124735/predicts_2005_d.pdf

[Highsmith,2002] Highsmith, J.: *Agile Software Development Ecosystems*, Addison-Wesley, 2002, ISBN 0-201-76043-6.

[Humprey,1999] Humprey, W.: *Pathways to Process Maturity: The Personal Software Process and Team Software Process.* Dostupný z WWW: http://interactive.sei.cmu.edu/Features/1999/June/Background/Background.jun99.htm

[Jacobson,Booch, Rumbaugh,1999] Jacobson, I.– Booch, G.– Rumbaugh, J.: *The Unified Software Development Process*, Addison-Wesley,1999, ISBN: 0201571692.

[Johnson,2001] Johnson, J.–Boucher K.D– Connors K.– Robinson J.: *Collaborating on Project Success*, Software Magazine, February/March 2001

[KIT,2003] *Terminologický slovník KIT*, Vysoká škola ekonomická, Katedra informačních technologií, 2002. Dostupný z WWW: http://www.cssi.cz [12.11.2003]

[Kroll,2001] Kroll, P.: *The Spirit of the RUP*, the Rational edge, 2001. Dostupný z WWW: http://www.therationaledge.com/content/dec_01/f_spiritOfTheRUP_pk.html

[Metagroup, 2003] *Summary of Results 2003 Worldwide IT Benchmark Report*, 2003. Dostupný z WWW: http://www.metagroup.com

[Paulk,1993] Paulk, M.C. – Curtis, B. – Chrissis, M. B. – Weber, Ch. V.: *Capability Maturity Model for Software*, Version 1.1, Technical Report CMU/SEI-93-TR-024. Dostupný z WWW: http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.024.html

[Polák,Merunka, Carda,2003] Polák, J.– Merunka, V.– Carda, A.: *Umění systémového návrhu, Objektově orientovaná tvorba informačních systémů pomocí původní metody BORM*, Grada, Praha 2003, ISBN 80-247-0424-2

[RUP,2001] *Rational Unified Process: Best Practices for Software Development Teams*, Rational Software White Paper TP026B, Rev 11/01.Dostupný z WWW: http://www-140.ibm.com/developerworks/rational/library/253.html

[Řepa,1999] Řepa, V.: *Analýza a návrh informačních systémů*, Ekopress, 1999, ISBN 80-86119-13-0

[Scrum,1995] *SCRUM Software Development Process, Building The Best Possible Software*. Dostupný z WWW: http://www.controlchaos.com/scrumwp.htm

[Soley,2000] Soley, P. : *Model Driven Architecture*, white paper OMG Group, 2000. Dostupný z: ftp://ftp.omg.org/pub/docs/omg/00-11-05.pdf

[SPICE,1998] ISO/IEC TR 15504: *Information technology - Software process assessment*, International Standards Organization, 1998 Dostupný z WWW: http://www.sqi.gu.edu.au/spice/

[Voříšek,1997] Voříšek, J.: *Strategické řízení informačních systémů a systémová integrace*, Management Press, Praha 1997, ISBN 80-85943-40-9.