

IT_572

„Prostředky CASE a jejich využití při tvorbě IS“

Téma práce:

Použití CASE pro řízení IS/ICT firmy

(CASE study + obecné závěry + trendy a možnosti)

2006

Autoři:

David Maleček

David Měrka

István Gyorfy

1 Úvod

Tato práce se zaměřuje popis a zhodnocení možností použití CASE nástrojů pro řízení IS/ICT. Danou otázkou se budeme zabývat ve dvou rovinách a to jednak na praktickém příkladu existující firmy (case study) a jednak teoreticky.

V praktické části budeme klást důraz zejména na popis předmětné oblasti v konkrétní firmě s diskuzí možných problémových oblastí event. možností využití CASE pro podporu řízení v identifikované oblasti.

V teoretické části se budeme snažit zejména uvést metodiky, best practices pro předmětnou oblast, diskutovat jejich obsah a vztah k předmětné oblasti. V současné době neexistuje (alespoň jsme žádnou neobjevili) konkrétní metodika pro řízení IS/ICT potažmo popisující a stanovující jakým způsobem toto řízení podporovat CASE nástroji. Budeme proto vycházet zejména z již existujících metodik, které jsou obvykle primárně zaměřeny na vývoj a implementaci IS/ICT resp. na řízení projektu IS/ICT a budeme se snažit ukázat jakým způsobem dané metodiky podporují řízení IS/ICT.

Ze všeho nejdříve ovšem definujeme co si pod pojmem řízení IS/ICT budeme v rámci této práce představovat.

2 Teoretická část

2.1 Definice řízení IS/ICT

Pod řízením IS a ICT budeme v rámci této práce považovat zejména následující činnosti:

2.1.1 Strategické řízení IS/ICT

Oblast činností, které jsou realizovány především na úrovni managementu společnosti. Obvykle se jedná o soubor dokumentů, příp. vizí, který definuje dlouhodobější budování IS/ICT ve firmě, základní pravidla, principy a cíle. Tyto aspekty by se pak měli promítnout a realizovat v rámci jednotlivých IS/ICT projektů. Jedná se zejména o:

- Informační strategie
- Bezpečnostní strategie a politika
- Organizace řízení IS/ICT

2.1.2 Taktické řízení IS/ICT

Zahrnuje činnosti, které souvisí přímo s vývojem a zaváděním nových prostředků IS/ICT. Jedná se zejména o:

- Řízení služeb IS/ICT
- Plánování projektu
- Řízení ekonomiky IS/ICT a metrik
- Řízení personálních a datových zdrojů

2.1.3 Operativní řízení IS/ICT

Do této oblasti patří jednak běžný provoz a podpora IS/ICT v každodenním běhu firmy. Zadruhé se pak jedná o řízení jednotlivých projektů, jejich vzájemných závislostí, kontrola jejich průběhů a plnění cílů.

2.1.4 Tvorba, správa a řízení dokumentace IS a ICT.

Jako specifickou oblast, která se prolíná všemi jednotlivými typy a úrovněmi řízení pak považujeme dokumentaci všech aspektů IS/ICT a to zejména s důrazem na její tvorbu, správu, perzistenci a dostupnost.

Další text dokumentu je veden zejména s ohledem na uvedené typy řízení a to v kontextu, jak daný typ je (či není) podporován dostupnými CASE nástroji event. metodikami.

2.2 Proč řídit IS/ICT

V současné době je činnost podniků stále více závislá na IS/ICT podniku. Jde tedy o podporu hlavních (core) podnikových procesů procesy informatiky podniku. Důsledkem tedy je, že flexibilita organizace je přímo závislá právě na flexibilitě podnikové informatiky.

S rostoucím rozsahem IT podpory pro různé oblasti činnosti organizace zároveň roste množství vazeb a závislostí mezi jednotlivými částmi IS/ICT podniku. Současně neustále rostou požadavky uživatelů podnikových systémů na rychlost implementace změn a nové funkcionality.

S přibývajícimi změnami v informačních systémech roste komplexita těchto systémů. Postupem času se vedle sebe kupí mnoho dílčích částí informačního systému od různých dodavatelů. Čím větší je komplexita těchto systémů, tím náročnější je IS řídit a tím vyšší a hůře odhadnutelné jsou náklady na integraci jednotlivých částí do správně jednotně pracujícího systému, na údržbu IS a na drobné změny za provozu. V důsledku komplexity se IS/ICT může snadno stát neovladatelným a nepředvídatelným systémem s fatálními riziky pro chod podniku. Takové změny a rozvoj informačních systémů je nutné nějakým způsobem řídit.

Pokud rozvoj podnikových systémů není řízen, dochází k neustálému zvyšování nákladů na IS/ICT, neboť špatně řízené promítání změn do informačních systémů je ve výsledku velmi nákladné. Prvotní promítnutí změny nebo přidání funkcionality se sice zpočátku při takovémto přístupu jako příliš nákladné nejeví, ale velmi záhy se bohužel ukáže potřeba provést ještě další změny, na které se původně zapomnělo, což ve výsledku znamená další úsilí a další náklady.

2.3 Proč využívat CASE nástroje při řízení IS/ICT

Spíše než proč využívat CASE nástroje při řízení IS/ICT by na úvod byla vhodnější otázka proč *modelovat* podnikové IS/ICT. Odpověď je nasnadě. Modelování IS nemá své opodstatnění pouze při vytváření a implementaci těchto systémů. Modely hrají důležitou roli i v případě řízení informatiky, tedy jejím provozu a rozvoji. Jak již bylo uvedeno výše, v dnešní době se díky vývoji IT technologií nesetkáme v případě rozsáhlejších IS se systémem, který by sestával pouze z jedné aplikace od jednoho dodavatele. Současné systémy v sobě zahrnují několik dílčích aplikací od různých dodavatelů, a tak je nutné toto nějakým způsobem řídit. Významným podkladem proto jsou modely informačních systémů.

Důležitým faktorem, proč modelování využívat je také další rozvoj a změny IS. Abychom mohli mluvit o řízeném promítání změn do IS, musíme mít jasno, jaké jsou dopady těchto změn. Pokud však máme podnikové systémy "namodelovány" pouze v podobě zdrojového kódu, je taková analýza dopadů velmi obtížná a nespolehlivá.

Protože modely procházejí při iterativním návrhu IS a při jeho následném řízení úpravami, je vhodné je vytvářet a upravovat pomocí CASE nástrojů, které je umožňují udržovat v aktuálním stavu. Tyto modely dokumentují stav návrhu v příslušném stádiu a po dokončení systému se stávají součástí finální dokumentace. Vidíme tedy, že řada zdrojů pro dokumentaci je uložena právě v CASE nástrojích.

Modelování IS může probíhat na různých úrovních detailnosti, přičemž úplně na vrchol bychom mohli postavit modelování procesů.

2.3.1 BPM (Business Process Modeling)

Pro celkový pohled na podnikový IS z hlediska dynamiky slouží procesní model. V tomto modelu je především zachycena vazba mezi firemními procesy a aplikacemi, které je podporují. Většina změn v systémech je totiž vyvolána změnami firemních procesů, takže model procesů je výchozím místem zkoumání při analýze dopadů změn. Na základě tohoto modelu jsme tak schopni odvozovat, které systémy budou změnou ovlivněny a v jakém rozsahu.

2.3.2 Model závislosti mezi jednotlivými aplikacemi (moduly, subsystémy) a jejich rozhraní

Vzhledem k tomu, že většina podnikových systémů je složena z jednotlivých vzájemně

provázaných aplikací od různých dodavatelů, jsou pro údržbu celého systému klíčovými informace o rozhraních aplikací. Základním modelem systému z hlediska jeho struktury je tedy model závislostí mezi jednotlivými aplikacemi (moduly, subsystemy) a jejich rozhraní.

Model rozhraní umožňuje provádět analýzu dopadů toho, jak změna jednoho systému ovlivní další systémy. Závislosti mezi aplikacemi podnikového IS jsou na tomto modelu zobecněním komunikace, která probíhá na úrovni rozhraní systémů.

2.3.3 Detailní modely aplikací

Jednotlivé systémy je pak možné detailněji modelovat pomocí tradičních UML modelů, jako je model tříd nebo model interakcí. Vytvoření a údržba těchto detailních modelů je samozřejmě podstatně nákladnější a vyplatí se u systémů, které jsou buď intenzivně rozvíjeny nebo při novém vývoji.

2.3.4 Dokumentace

Dokumentace hraje důležitou roli již v procesu vývoje programového systému. Jednotlivé fáze vývoje je třeba dokumentovat a každá fáze má většinou definovány určité výstupní dokumenty (záleží na zvolené metodice vývoje IS). Díky dokumentaci tak máme dokonalý přehled o tom, kde, co a jak je implementováno.

Právě zde se naskytuje možnost využití dokumentačních nástrojů zvoleného CASE nástroje. Jednou z hlavních předností modelování IS/ICT pomocí CASE nástrojů a jejich následného využití při řízení informatiky, je schopnost těchto nástrojů generovat a udržovat aktuální dokumentace systému, které jsou vždy (nebo by tomu tak alespoň mělo být) nedílnou součástí aplikací. Nejen že tyto systémy jsou schopné dokumentaci jednorázově vygenerovat, ale jsou ji rovněž schopny udržovat v aktuálním stavu dle provedených změn v modelech systému.

Většina současných strukturovaných i objektově orientovaných CASE prostředků má v sobě totiž integrované prostředky pro tvorbu všech typů dokumentace vznikající v různých fázích životního cyklu. Nejedná se zpravidla jen o pouhý tisk diagramů vyskytujících se na obrazovce, nýbrž o dokonale hierarchicky zpracovanou dokumentaci ve formě přehledových sestav, detailních popisů entit, atributů, relačních a dědičných vztahů a toto vše teprve doplněno jednotlivými diagramy.

2.3.5 Model jako komunikační nástroj

Protože na vývoji i na následném řízení IS pracuje několik lidí, jsou modely nezbytné i

jako komunikační prostředek mezi nimi. Jednak jsou součástí finální dokumentace systému a jednak jsou přístupné pomocí CASE nástrojů.

Celkový model podnikového IS je také velmi užitečným podkladem pro komunikaci s dodavateli jednotlivých aplikací při zadávání požadavků na funkcionalitu dodávaných aplikací.

2.3.6 Audit IS/IT

CASE nástroje, potažmo dokumentace a vytvořené modely jsou také důležitým podkladem pro audit informačních systémů.

2.4 Výsledný přínos pro organizaci

Nasazení systematictějšího přístupu nastíněného výše, který je navíc podpořen nástrojem CASE usnadňujícím vytváření a údržbu příslušných modelů, vede zpravidla ke znatelným úsporám nákladů na rozvoj a údržbu podnikového IS/ICT. Tyto úspory jsou však nejen finančního charakteru, který vyplývá z toho, že změny se daří úspěšně realizovat na první pokus, ale i charakteru praktického, neboť používání CASE nástrojů a modelování všeobecně vede k přehlednosti a transparentnosti informačního systému.

3 Rational Unified Process (RUP)

Protože se v naší firmě přistupuje k řízení informatiky dle metodiky RUP (nebo se této metodice alespoň přibližují), bude jistě na místě si tuto metodiku alespoň stručně představit.

Rational Unified Process (RUP) je metodika pro vývoj software, která byla vytvořena společností Rational Software, jež se stala divizí společnosti IBM. RUP není nějaký konkrétní normativní (předpisující) proces, jedná se spíše o jakýsi adaptabilní procesní rámec. Jako takový, RUP popisuje způsob, jak vyvíjet software efektivně s využitím osvědčených technik. Zatímco RUP v sobě zahrnuje velké množství různých aktivit, je rovněž určený (zamýšlený) k tomu, aby mohl být přizpůsobený na míru, a to ve smyslu výběru (vývojového) procesu odpovídajícímu konkrétnímu projektu či vývojářské společnosti. RUP je vhodný při aplikaci v týmech pracujících na rozsáhlejších projektech vývoji sw.

3.1 Čtyři nejdůležitější elementy metodiky RUP

V metodice Rational Unified Process jsou definovány čtyři základní pojmy, na nichž je celý RUP postaven a z nichž sestává celý vývoj podle této metodiky. Pro RUP jsou to

nejdůležitější elementy modelování, na nichž staví nejen celé modelování, ale přeneseně i celý vývojový proces (pomocí těchto elementů se modelují základní případy použití a pracovní procesy, z nichž se celý RUP skládá). Jedná se o:

- pracovníky (Workers),
- činnosti (Activities),
- pracovní procesy (Workflows)
- a meziprodukty (Artifacts).

RUP pokrývá celý životní cyklus vývoje softwaru – a skoro vše, co RUP obsahuje, lze označit jedním z těchto základních pojmů.

3.2 Best practices metodiky RUP

Kromě čtyř základních elementů je RUP založen také na šesti best practices, což znamená, českým jazykem řečeno, šesti základních pravidlech, které byly ozkoušeny jako nejlépe fungující. Zde je jejich výčet:

1. iterativní vývoj softwaru,
2. správa a řízení požadavků,
3. použití komponentové architektury,
4. vizuální modelování softwaru,
5. ověřování kvality softwaru,
6. správa a řízení změn.

Protože tato semestrální práce se zaměřuje na řízení IS/ICT (a nikoliv na samotný vývoj SW), budeme se v dalším textu zabývat pouze best practice, která se řízení IS/ICT bezprostředně týká.

3.2.1 Správa a řízení změn a konfigurací

V každém softwarovém projektu jsou změny nevyhnutelné. Neřízené změny (ať již při vývoji nebo i po předání) vedou k chaosu. RUP proto definuje metody ke kontrolování a monitorování změn. To je důležité pro zachování konzistence systému. Hlavním úkolem je řídit verzování software, vytvořit systém předávání posledních verzí veškerých výstupů a

koordinovat implementaci změn.

Protože jedním ze základních pilířů RUPu je iterativní přístup, je potřeba řízení změn o to důležitější. Změnový management v rámci RUPu počítá se třemi specifickými oblastmi:

- Řízení konfigurací (Configuration management)
- Řízení změnových požadavků (Change request management)
- Řízení stavů a měření (Status and measurement management)

3.2.2 Řízení a správa změn a konfigurace – postupy:

- Plánování konfigurace projektu a řízení změn

Do plánování konfigurace a řízení změn spadá především tvorba plánu konfigurace, tvorba pravidel řízení konfigurace a tvorba pravidel změnového řízení.

- Vytvoření prostředí pro řízení konfigurace

Tento krok spočívá ve vytvoření jednak společného repozitáře (ten slouží pro sdílení dat a kódu při vývoji sw), jednak repozitáře pro integraci (do kterého se ukládají hotové a řádně otestované komponenty ze společného adresáře).

- Změna a předávání částí konfigurace

Jedná se o postup vymezující přístup pracovníka v libovolné roli k projektovým datům, provádění a publikaci změn.

- Správa verzí

Zahrnuje uložení verze komponenty do společného repozitáře a vytvoření modulu pro nasazení.

- Řízení změn

V průběhu životního cyklu IS prochází většina artefaktů několika verzemi. Proto je nutné sledovat a řídit jednotlivé požadavky na změny. Řízení změn (Change management) je standardizovaný, centrálně řízený postup řízení a dokumentace změn, který umožňuje udržení systému v konzistentním stavu i v případě úprav. Jedná se o následující procesy:

- Podání požadavku na změnu
- Zhodnocení požadavku na změnu

- Oznámení duplicity nebo zamítnutí požadavku
- Zadání řešení požadavku

4 Model Driven Architecture

Model Driven Architecture byla zveřejněna v roce 2001 a již nyní získala velký vliv na metody vývoje aplikací. V současné době existuje přinejmenším 40 nástrojů, které podporují alespoň jeden z hlavních aspektů MDA. Jedná se o koncept standardizující modely, které jsou v průběhu vývoje aplikace vytvářeny a definující způsoby mapování mezi nimi. Proto také název, který by se dal přeložit jako „Modelem řízená architektura“, zkráceně MDA. MDA není ve své podstatě převratnou novinkou, nakonec otázkou členění modelů se zabývá každý rozumná metodika softwarového vývoje, nicméně podstatné je to, že toto členění standardizuje a dále definuje způsob transformace jednoho modelu v druhý.

Koncept MDA sice úzce souvisí s UML, avšak není ne tento modelovací standard bezprostředně vázaný, neboť se dá aplikovat i jiným způsobem modelování než je UML.

4.1 Princip MDA

Princip MDA je velmi jednoduchý, spočívá v oddělení business logiky od technologie platformy. Platformou se rozumí sada subsystémů nebo technologií, které zajišťují logickou sadu rozhraní a vzorů, které může jakákoliv aplikace využívat bez potřeby vědět, jak je která funkce, poskytována platformou, implementována. Toto umožňuje realizovat aplikace vytvořené pomocí MDA v celé řadě otevřených platform jako např. CORBA, J2EE, .NET a jiných webových platformách. Nezávislost na platformě nabývá významu tím více, čím rychleji jsou vytvářena nové a nové technologie.

4.2 Modely dle MDA

MDA vidí modely aplikací ve čtyřech úrovních:

- CIM – model nezávislý na počítačovém zpracování
- PIM – platformově nezávislý model řešení
- PSM – platformově specifický model řešení
- Code – kód aplikace, tj. výsledná realizace řešení

4.2.1 CIM – Computation Independent Model

Model nezávislý na počítačovém zpracování je de facto modelem vlastního „businessu“, čili činností, které musí vykonávat pracovníci firmy aby tato mohla fungovat. Model CIM tedy znázorňuje tyto činnosti a obvykle se také nazývá model firemních procesů. Celkem pikantní je, že notace modelu CIM není v UML plně standardizována (a to ani v chystané verzi UML 2.0). Tento model vytvářejí buď sami uživatelé nebo business analytici.

Model CIM je při vývoji a údržbě důležitý především pro to, aby vývojáři pochopili činnosti uživatele a chápali příslušné souvislosti. Navíc je to jediný model, který lze bez obav předložit běžnému uživateli a ten je schopen ho po krátkém vysvětlení pochopit a především se k němu vyjadřovat. Další modely totiž nejsou pro uživatele snadno pochopitelné a jsou určeny především vývojářům.

4.2.2 PIM – Platform Independent Model

Platformově nezávislý model reprezentuje koncepci řešení dané problémové oblasti na základě konkrétních požadavků. Jeho hodnota je především ve vyřešení koncepčních otázek, jako jsou třeba algoritmy zaskladňování a vyskladňování v případě skladů, nebo způsob párování saldokontních položek v účetnictví. Tento model však neobsahuje informace spojené s konkrétní technologií realizace a vytvářejí ho IT analytici.

Platformově nezávislý model umožňuje vyřešit určitou oblast na obecné úrovni a teprve pak zvažovat konkrétní technologii pro vlastní realizaci. Dalším důležitým aspektem pro vytváření PIM modelu je to, že je cca 3x jednodušší než model, který již specifické technologické informace obsahuje a tudíž se v tomto modelu snadno zorientuje analytik, který obvykle nemá (ani to není žádoucí) detailní technologické znalosti.

4.2.3 PSM – Platform Specific Model

Model řešení na dané platformě (např. J2EE nebo C# a ASP.NET) je podkladem pro vlastní implementaci. Z hlediska vztahu ke kódu je důležité to, že PSM má stejnou strukturu jako kód aplikace. Tento model vytvářejí návrháři.

4.2.4 Code

Z hlediska MDA je zdrojový kód chápán také jako model konkrétní realizace na dané platformě. Konec konců určitě znáte ze svého okolí aplikace, kde jedině tento model skutečně odpovídá realitě toho, co aplikace dělá.

4.3 Nač tolik modelů?

MDA tedy doporučuje při tvorbě a údržbě aplikací vytvořit a udržovat tyto čtyři modely, neboť se tím výrazně zjednoduší údržba a rozvoj aplikace. Podívejme se nyní proč.

Udržování CIM modelu umožní vývojářům analyzovat dopady změn v činnostech uživatele na příslušnou aplikaci. Vezměme si jako příklad jarní změnu zákona o DPH. V tomto zákoně je například příchod platby považován za zdanitelné plnění a je nutné k ní vystavit daňový doklad. Implementovat do stávající ekonomické aplikace automatické vystavování daňového dokladu při příchodu platby se může zdát triviální záležitostí. Pokud však není jasné, kdo bude tento doklad odesílat zákazníkovi, zda ho bude odesílat vždy nebo jenom na vyžádání, může velmi snadno dojít k tomu, že změna je sice implementována zdánlivě správně, ale není možné jí použít. Proto je potřeba nejprve na modelu CIM analyzovat jak se změní, či rozšíří činnosti uživatele a pak teprve můžeme uvažovat, jak tyto změny ovlivní naši ekonomickou aplikaci.

Udržování aktuálního modelu PIM je pak klíčové pro analytiku, kteří musí definovat příslušnou změnu na koncepční úrovni. Velice častým problémem je, že aplikace je dokumentována pouze platformově specifickým modelem, který obsahuje množství informací souvisejících s technologickým řešením. Tyto informace však vlastní koncepční řešení značně zatemňují a výrazně ztěžují analytikovi práci. Důsledkem jsou pak časté chyby způsobené opomenutím nebo tím, že analytik zasahuje do technologických záležitostí, kterým až tak dobře nerozumí. V našem příkladu s daňovým dokladem k platbě musí analytik vyřešit především způsob jeho číslování, vazby na platby a další související doklady a v neposlední řadě například způsob archivace. Pokud se však musí probírat modely plnými session kontrolérů, object brokerů a podobných technologických objektů, jeho produktivita a kvalita práce samozřejmě značně klesá.

Modely CIM a PIM teda potřebujeme, ale na co nám je platformově specifický model – PSM? Jak již bylo řečeno, PSM model znázorňuje technologický způsob řešení a má stejnou strukturu jako zdrojový kód aplikace. Právě posledně uvedené je velmi důležité – PSM je totiž v podstatě vizualizací kódu. Pokud jste měli možnost programovat rozsáhlejší aplikaci, jistě jste narazili na problém jak se vyznat ve spoustě programových tříd a množství kódu. A právě k tomu slouží PSM model – abychom se vyznali v kódu a to zpětně, jako forma dokumentace, ale i dopředu, jako zadání pro vytvoření kódu. Některá vývojová prostředí (Integrated Development Environment – IDE) dnes již obsahují takovéto „vizualizéry“ kódu ve formě

UML modelu, obvykle však jen ve svých Enterprise verzích.

4.4 Transformace – aneb snadno z modelu do modelu

V čem spočívá nejdůležitější přínos MDA je to, že definuje kromě shora uvedených modelů také způsob a postup jejich transformace. Opět se však nejedná o zcela nový přístup, ale spíše o standardizovanou aplikaci osvědčených praktik, především zkušeností z použití návrhových vzorů (Design Patterns).

Z praktického hlediska jsou zajímavé transformace CIM do PIM a PIM do PSM.

4.4.1 Transformace CIM <-> PSM

Tato transformace vyjadřuje vztah mezi činnostmi uživatele a funkcionalitou aplikace. MDA se sice touto transformací příliš nezabývá, ale obvyklý postup je transformace firemních procesů do akcí uživatele v aplikaci (v UML tzv. Use Cases).

4.4.2 Transformace PIM <-> PSM

Z hlediska MDA je nejzajímavější transformace platformově nezávislého modelu (PIM) do platformově specifického modelu (PSM).

Postup transformace dle MDA je zhruba následující:

1. Návrhář doplní PIM model tzv. mapovacími značkami, které definují, jaká obecná transformační pravidla budou použita.
2. Pro PIM model (resp. jeho části) je zvolena implementační platforma.
3. Na základě mapovacích značek jsou provedeny odpovídající transformace již s ohledem na zvolenou platformu.

4.5 K čemu tedy MDA slouží?

Přínos koncepce Model Driven Architecture je tedy především v tom, že jasně definuje, co je analytický model, co je návrhový model a jak provádět jejich transformaci. Cílem je, aby aplikace byla popsána na různých úrovních abstrakce a tím pádem je značně usnadněno konzistentní promítání změn v aplikaci. Další pozitivní důsledek je standardizace návrhu, která umožňuje zvýšit kvalitu aplikací. Důsledkem toho všeho je především snížení nákladů na údržbu aplikací, tedy peníze, o které jde jako obvykle až v první řadě.

Jak již bylo zmíněno na začátku, MDA je především o zjednodušení a tím i zlevnění

údržby a rozvoje aplikací. Vzhledem k tomu, že právě sem směřuje větší část investic související s vývojem softwaru (Gartner Group uvádí že až 70%), je tato koncepce nakonec zajímavá i po finanční stránce. MDA se bude zcela jistě dále rozvíjet. Budou vytvořeny mocnější nástroje, které budou tuto technologii využívat. S těmito nástroji bude zřejmě ubývat nutnost upravovat PSM a vše se již bude definovat v PIM, který bude spustitelný a testovatelný (bude tedy převládat přístup translationist). Potom budou tyto aplikace opravdu Model Driven.

5 Další přístupy k řízení IS/ICT

Dalšími přístupy (zde bude lepší využít tento pojem než pojem „metodika“) jsou ITIL a COBIT. Jedná se spíše o manažersky pojaté přístupy k řízení IS/ICT a proto se jimi nebudeme příliš zabývat, ale pouze zmíníme jejich stručnou charakteristiku.

5.1 ITIL (*Information Technology Infrastructure Library*)

První z nich je metodika ITIL. „ITIL je rozsáhlý, konzistentní a procesně orientovaný rámec pro oblast IT Service Managementu.“¹ Jedná se o sadu 8 knižních svazků obsahujících best practices pro efektivní dodávku, správu a řízení IS/ICT. Poskytuje tedy komplexní na zkušenostech založenou sadu postupů, které byly v praxi ověřené jako spolehlivé a fungující.

Charakteristickým znakem ITILu je procesně orientovaný přístup k řízení IT služeb. Zavádí též mezinárodně používanou terminologii v oblasti řízení IS/ICT. Vzniká již od konce 80.let, vyvíjen je v rámci organizace OGC (British Office of Government Commerce).

Dělení správy ICT infrastruktury dle ITILu:

- Design and planning - návrh ICT infrastruktury,
- Deployment - implementace a nasazení plánovaného ICT řešení,
- Operations - provoz stávající ICT infrastruktury a její běžná údržba (monitoring a řízení),
- Technical support - analýza dat z dohledových systémů a zpětná vazba.

¹ www.itil.cz

5.2 COBIT (Control Objectives for Information and Related Technology)

COBIT byl vyvinut jako všeobecně použitelný a přijímaný přístup k řízení, kontroly a auditu ICT. Původně jej vyvinula auditorská organizace ISACA v roce 1996. Na dalším vývoji se od roku 1998 podílí institut IT Governance. COBIT představuje standard pro řízení informatiky a kontrolní nástroj pro auditory. Přístup je však odlišný než v případě ITILu: COBIT je soupisem určitých stavů, ve kterých by se firemní IS/ICT měla nacházet.

Základní COBIT definuje 34 procesů seskupených do čtyř následujících procesních domén:

- plánování a organizace
- akvizice a implementace
- poskytování a podpora
- monitorování.

6 Případová studie firmy A

6.1 Představení firmy.

Zkoumaná firma působí v oblasti obchodně a manažersky zaměřených školení a vzdělávání zejména zaměstnanců farmaceutických společností. V současné době ve společnosti pracuje cca 30 zaměstnanců – vesměs lektorů. Vzhledem k tomu, že školení se pořádají na nejrůznějších místech republiky, popř. v zahraničí, vystupují zde nároky zejména na:

1. koordinaci jednotlivých lektorů a jejich účasti na jednotlivých školeních
2. koordinace a najímání školící techniky a prostor
3. zajišťování ubytování a s tím spojených služeb (strava, program) pro účastníky vícedenních školení.
4. marketingová a obchodní činnost – vytváření nabídek, aktivní řízení vztahu se zákazníky, telemarketing
5. vedení dlouhodobých vzdělávacích programů a projektů pro významné zákazníky
6. řízení účasti ve výběrových řízeních
7. řízení vzdělávání vlastních zaměstnanců, získávání akreditací a certifikace vzdělávacích programů

Z pohledu informačního systému a jeho vývoje prošla společnost několika stádii.

V prvním stádiu, vzhledem k malému rozsahu činnosti, neexistovala potřeba zavádění specializovaných sw produktů. Existující požadavky byly řešeny implementací existujících komerčních produktů a to zejména v oblasti komunikace, tvorby dokumentů a finančního řízení a účetnictví.

S rozvojem rozsahu působnosti společnosti vyvstal požadavek na vznik nebo pořízení specifického sw produktu, který by zajišťoval jak koordinaci a plánování jednotlivých školení, tak umožňoval evidenci školení jednotlivých zákazníků, podporoval různé marketingové akce (slevy za objem zakázek apod.), řízení projektů a dlouhodobých programů

pro jednotlivé zákazníky, aktivní oslovování a evidenci vztahu s novými subjekty.

Nastalá situace byla řešena poptávkou po vývoji proprietárního nového řešení založeného na třívrstvé architektuře a webových technologiích. V důsledku se jednalo o komplexní IS složený z jednotlivých modulů, které byly vyvíjeny separátně – postupně, v rámci jednoho připraveného prostředí jak na úrovni db tak na úrovni prezentační a business logiky. Vývoj probíhal s využitím externích zdrojů a celková doba dodávky všech požadovaných modulů byla nakonec cca 15 měsíců – od analýzy na první modul a celou koncepci řešení až po dodání posledního smlouveného modulu.

Na vývoji se celkem podílelo pět analytiků/programátorů a to v různých fázích projektu různí lidé. Z tohoto důvodu bylo potřeba již na začátku projektu vydefinovat způsob a zajištění časové kontinuity analýz a designů a dokumentace jednotlivých programových modulů tak, aby bylo možno v relativně krátkém čase zaučit i nového programátora do již vyvinutých částí systému. V následujících jednotlivých podkapitolách bude řečeno jestli a jakým způsobem jsou realizovány jednotlivé činnosti řízení IS/ICT v popisované firmě a zda je využíváno CASE nástroje.

6.2 Vývoj IS/ICT - metodika

Na začátku uvedeného projektu neexistovala v popisované firmě žádná metodika (ani formální ani neformální) pro vývoj IS/ICT a zejména pro podporu zhodnocení, do jaké míry byl ten který projekt úspěšný či ne. Jakási metodika se tedy tvořila v průběhu popisovaného projektu a vyústila v podstatě v metodiku velmi podobné metodice RUP, ovšem v mnohem zjednodušenější formě.

Z pohledu řízení IS/ICT spadá spíše do oblasti operativního řízení – popisuje totiž průběh infromatického projektu jeho jednotlivé fáze a jejich výstupy, používané nástroje. Pro názornost uvedu její stručný popis.

Metodika dělí vývoj a implementaci IS/ICT do dvou oddělených fází. První fáze je analytická a vzniká při ní analytický model budoucího IS/ICT. Vstupem do této fáze bývá seznam požadavků zadavatele na budoucí aplikaci. U složitějších nebo komplexnějších funkcionalit následně (v podstatě samovolně) začaly vznikat i jakési jednoduché procesní modely – bylo to vyvoláno nutností vyjasnit si se zadavatelem průběh nebo konkrétní aspekt určité činnosti, kterou měl požadovaný programový modul podporovat. V tomto ohledu se celkem dobře

osvědčil CASE nástroj a procesní diagramy byly velmi často využívány pro přímou komunikaci se zadavatelem. Nikdy ovšem nebyl vytvářen (ani nevznikl požadavek ze strany zadavatele) komplexní BPM firmy a jejích činností.

Analytický model jako takový je pouze logický, jeho fyzická podstata je vyjádřena zejména návrhem obrazovek budoucího systému, popisem scénářů chování systému a popisem jeho vnitřní logické struktury, který je realizován class diagramem. V této fázi je plně využíván modelovací CASE nástroj – v daném případě Enterprise Architect – v němž je vytvořen celý model. Ten je pak exportován do dokumentace s předem předepsanou základní strukturou. První fáze končí schválením zadání (modelu) pro programování zadavatelem a přechází ve fázi druhou a tou je programování, testování, implementace.

Obecně tato fáze obsahuje vlastní kódování, případně nákup potřebné ICT, tvorbu fyzických modelů databází. CASE nástroj je v dané firmě v této fázi projektu využíván pouze k vytvoření entitního (databázového) modelu – pokud je danou aplikací vyžadována databáze – a vygenerování skriptů pro tvorbu databáze z daného modelu. Analytický model se zpravidla již nemění. Designový model se obvykle nedělá – aplikace nebývají tak implementačně složité, že by bylo nutné analytický model rozpracovávat do designu.

V rámci této, druhé, fáze se též provádí testování naprogramovaných modulů uživateli – jakési akceptační testy. Pro jejich podporu se v případě složitější logiky modulů využívá schopnost CASE nástroje pro tvorbu a generování testovacích scénářů a podpora řízení testů – k jednotlivým scénářům je možné evidovat jejich stav pro testování, event. evidovat požadavek na změnu a přetestování apod. Nicméně toto nebývá realizováno v každém projektu, ale pouze v případě, že logika modulu je již natolik složitá nebo jeho funkcionalita je vyhodnocena jako kritická (např. se pracuje s fin. ukazateli), že je nezbytné provést komplexní a úplné testování modulu.

U složitějších implementací je též doporučeno vytvořit tzv. model nasazení – tj. diagram popisující nasazení jednotlivých komponent na jednotlivé systémy. Nicméně s ohledem na jednoduchou architekturu IS/ICT ve zkoumané firmě, tato možnost nebyla nikdy využita.

Obecně vzato slouží v dané firmě CASE nástroj pouze pro potřeby analytiků a event. programátorů v rámci daného projektu na vývoj IS/ICT. Pro potřeby dokumentace systému, jeho následné správy a provozu je pak z nástroje generována příslušná dokumentace, která

bývá doplněna o dokumentaci programového kódu a tedy pro další potřeby bývá využívána tato dokumentace.

6.3 Provoz IS/ICT

Jak již bylo popsáno výše, k podpoře provozu IS/ICT ve firmě je používána dokumentace, která je sice generována z CASE nástroje, ale nástroj není při provozu používán přímo. Samotný proces provozu IS/ICT není podporován CASE nástrojem, resp. tento není v jeho průběhu nijak využíván a dokonce lze říci, že proces provozu IS/ICT není ani formálně více popsán a definován. Provoz je ve firmě realizován outsourcingem – obvykle tím, kdo daný IS nebo ICT dodal – a bývá více či méně smluvně definován. Jestli jednotlivé subjekty používají CASE nástroj pro podporu poskytování provozního outsourcingu není známo, ale předpokládáme, že nikoliv.

Jediným specifickým případem, kdy bývá CASE nástroj využit v provozu je situace, kdy je potřeba provádět změny datového modelu běžící databáze. Jak bylo popisováno výše, databáze bývá generována z modelu v CASE nástroji a i případné změny ve struktuře bývají nejdříve modelovány a pak za využití generovaných skriptů realizovány přímo v db.

6.4 Řízení služeb IS/ICT

Z rozhovoru se zástupci firmy vyplynulo, že tato oblast není (za současných podmínek – velikost firmy, její obrat, složitost IS/ICT) v danou chvíli aktuální a tedy není sledována, řízena ani jinak zpracovávána. Veškerá funkcionalita systémů je popsána v příslušné dokumentaci a v současné době není natolik rozsáhlá, aby vyžadovala komplexnější přístup – jakým hlubší řízení služeb IS/ICT je.

6.5 Plánování projektu IS/ICT

Vzhledem k tomu, že v podstatě všechny projekty v této oblasti jsou outsourcované je obvykle zodpovědností dodavatele vytvořit a předložit plán projektu. V doposud realizovaných projektech ovšem bylo toto plánování prováděno mimo CASE nástroje – ostatně většina v současnosti používaných nástrojů nepodporuje své využití pro plné plánování a následné řízení projektu – nejčastěji pomocí produktu MS Project, nebo i jen formou obyčejných dokumentů.

Firma není natolik velká aby v ní existovalo (v oblasti IS/ICT) multiprojektové prostředí. V podstatě vždy se v jednom čase provádí pouze jeden projekt.

6.6 Řízení ekonomiky IS/ICT a metrik

Firma opět neprovádí důsledné řízení ekonomiky IS/ICT v závislosti na hodnocení veličin dosahovaných určenými metrikami. Vzhledem k outsourcingu většiny IS/ICT prostředků, jsou náklady na jednotlivé prostředky v daném období relativně dobře vyčíslitelné pomocí přijatých faktur od jednotlivých dodavatelů.

Rozhodně pro tuto oblast není využíván CASE nástroj a tato oblast je spíše předmětem finančního řízení firmy a tedy jejích finančních a účetních systémů. Zodpovědnými osobami za tuto oblast jsou ve firmě pouze její majitelé, kteří též přijímají rozhodnutí o uzavírání resp. rozvazování smluvních vztahů s jednotlivými dodavateli.

6.7 Řízení personálních a datových zdrojů

Řízením personálních zdrojů se firma nezabývá. V oblasti IS/ICT nemá žádné interní zdroje a tedy není co řídit. Oblast řízení datových zdrojů je realizována částečně pomocí CASE nástroje (jak bylo uvedeno výše). Nicméně vzhledem k relativně jednoduché architektuře a v podstatě jednoho integrovaného datového zdroje v rámci většiny kritických IS firmy není toto řízení komplikované a je opět v kompetenci dodavatelské firmy.

6.8 Informační strategie

Firma má vypracovanou jednoduchou formu informační strategie. Reálně se jedná o dokument, ve kterém je stanoven střednědobý plán rozvoje IS firmy a hrubý odhad potřebných financí. V této oblasti není využíván žádný CASE nástroj. Vzhledem k faktu, že daný dokument je vypracováván v přímé spolupráci se zástupci dodavatelských firem – zde působí jako konzultanti – je zvolena forma obyčejného dokumentu s předem stanovenou strukturou a rozsahem obsahu.

6.9 Bezpečnostní strategie a politika

Je ve firmě součástí informační strategie a vzhledem k velikosti firmy nebývají tyto oblasti specificky řešeny ani spravovány. Realizovány bývají na úrovni jednotlivých projektů resp.

programových modulů a tedy bývají (ve své realizaci) součástí modelů v CASE nástroji nikoliv ale explicitně.

6.10 Organizace řízení IS/ICT

Jak již bylo zmíněno výše, oblast řízení IS/ICT ve firmě zajišťují majitelé firmy. Vzhledem k velikosti firmy opět není nutné provádět explicitní organizaci řízení IS/ICT ani pro tuto činnost definovat a provozovat nějaké specifické postupy a nástroje.

6.11 Tvorba, správa a řízení dokumentace IS a ICT.

Skrze všechny výše uvedené činnosti při řízení IS/ICT se prolíná problematika tvorby, správy a řízení dokumentace. Ve zkoumané firmě je kladen relativně velký důraz na tvorbu kvalitní, dostupné a úplné dokumentace k dodávaným IS/ICT – vedení firmy si uvědomuje (jak se ostatně již několikrát stalo), že dokumentace je jediné místo, ve kterém jsou obsaženy informace o jejich IS/ICT.

Jak již bylo řečeno v rámci metodiky vývoje IS/ICT je definován soubor dokumentů spolu s jejich obsahem a strukturou a pro vytváření těchto dokumentů jsou používány nástroje CASE. Výhodou je, že dokumentace tak plně popisuje to co je skutečně modelováno a po přidání dokumentace programových kódů i toho co je reálně implementováno.

Vlastní správa (uložení, sdílení a distribuce) dokumentace probíhá mimo CASE nástroje s využitím standardních prostředků jako jsou sdílené disky, přístupová práva, emailová komunikace atd.

Jako výchozí šablony pro jednotlivé dokumenty byly využity výstupy z CASE Enterprise Architect. Firma taktéž požaduje (v souladu s metodikou) vytváření a používání diagramů a modelování v notaci UML. Výhodou stanovení takového dlouhodobě vyžadovaného pravidla je fakt, že zodpovědné osoby na straně businessu firmy se postupem času naučily číst a rozumět základním diagramům UML a tedy se podstatně (oboustranně) zlepšila, zrychlila a zjednodušila i komunikace mezi firmou jako zadavatelem a příslušným dodavatelem. Zároveň též jsou dodavatelé nuceni dokumentovat svá řešení, což snižuje (pro popisovanou firmu) riziko při budoucím eventuálním rozvázání smluvního vztahu s daným dodavatelem.

7 Shrnutí

Na závěr této práce uvedu shrnutí základních aspektů využívání CASE nástroje v popisované firmě pro řízení IS/ICT:

1. CASE nástroj je využíván zejména v oblasti operativního řízení a to hlavně v průběhu vývoje a implementace nového IS/ICT
2. CASE nástroj je v omezené míře používán v oblasti podpory provozu existujících IS/ICT. Zde by mohlo být realizováno širší užití.
3. V oblasti strategického řízení v podstatě není realizováno použití CASE nástroje. Částečně toto lze vysvětlit i tím, že firma je malá a tedy obecně oblast strategického řízení není natolik složitá jako u firem větších (co do rozsahu).
4. V oblasti taktického řízení je CASE nástroj využíván jen ve velmi omezené míře nebo vůbec. Zde bychom viděli největší příležitost pro větší využití CASE nástroje a to zejména v souvislosti s definováním a popisem obchodních procesů – oblast, která v této chvíli není nijak řešena – a následně definicí a popisem služeb IS/ICT a jejich vazby na obchodní procesy – tedy definicí co je čím podporováno. Využití CASE nástroje v této oblasti se jeví jako velmi vhodné – současné CASE nástroje celkem dobře podporují modelování a popis této oblasti. Zároveň by pak takový popis mohl sloužit jako jeden ze vstupů při definování požadavků na budoucí aplikace.

Závěrem lze říci, že dílčí činnosti a potřeby při řízení IS/ICT v této firmě byly podporovány CASE nástrojem, ale zejména jako důsledek metodiky pro vývoj a implementaci IS.

Velké mezery jsou zejména v oblasti modelování BP z oblasti IS/ICT, kde by CASE nástroj poskytoval poměrně slušné možnosti. Naopak v jiných oblastech jsou schopnosti CASE nástrojů obecně omezené popř. vůbec žádné – hodnocení, měření a podpora provozu, jen částečně podpora administrace (spíše ve smyslu dokumentačního nástroje).