CRITICAL REQUIREMENTS OF THE INFORMATION SYSTEM DESIGN

Zdeněk FIEDLER, Master Degree Programme (4)
Dept. of Information Systems, FIT, BUT
E-mail: xfiedl03@stud.fit.vutbr.cz

Supervised by: Dr. Jaroslav Zendulka

ABSTRACT

The contribution is aimed to the process of the information system creation. It summarizes methods for managing the process and compares them mutually. The goal is to identify parameters that influence the process the most. The concern is laid on the information – requirements, which are necessary to obtain for satisfying time and budget constraints. It deals with the requirements analysis and its connection to the methods for the process management. It also identifies the typical troubles of the requirement analysis and shows possibilities for their solution. On the bases of those possibilities, the rules for determination of the critical requirements and the principles for handling them are defined.

1 ÚVOD

Jedním z problémů, se kterým se v dnešní době potýkat řada softwarových projektů, je nedodržení rozpočtu a plánovaného termínu dokončení. Naší snahou je odhalit klíčové příčiny tohoto jevu a pokusit se navrhnout možné postupy řešení.

2 METODY TVORBY INFORMAČNÍCH SYSTÉMŮ

Chceme-li pochopit problémy, musíme se seznámit s používanými metodami. Rozlišujeme tzv. metody řízení, jejichž cílem je řídit samotný proces tvorby systému, a dále metody praktické tvorby, které obsahují jednotlivé fáze analýzy, návrhu, vývoje, testování, nasazení a údržby. Metody řízení jsou obecně použitelné pro řízení libovolných procesů. Jejich použití v informačních technologiích má některá úskalí a drobné odlišnosti, nicméně jejich vlastnosti se nemění. K těmto metodám jsme zařadili:

- Teorie omezení (Theory of Constraints)
- Balanced Scorecards

Jejich podrobnému popisu se nebudeme dále věnovat. Je také třeba podotknout, že některé z metod praktické tvorby obsahují i vlastní postupy řízení a stojí tak na rozhraní těchto dvou skupin.

Metody praktické tvorby jsou úzce vázány na použitý model životního cyklu. Ten také do značné míry ovlivňuje jejich chování. Zjednodušeně řečeno jsou si metody, vystavěné nad stejným modelem životního cyklu, velmi podobné. Zařadili jsme sem následující metody:

- Yourdonova moderní strukturovaná analýza
- Unified Process
- Extreme Programming
- Model Driven Architecture

Tyto metod představují průřez několika obdobími praktické tvorby a podrobně jsou popsány v [3]. Yourdonova analýza používá jako svůj model životního cyklu tzv. vodopád. Představuje vrchol strukturovaných analýz konce 80. let dvacátého století. Vývoj programovacích technik a především nástup objektové orientace si vyžádal nové přístupy. V dnešní době je široce používán Unified Process, který obsahuje i postupy řízení a je podporován programovými prostředky. Klade důraz na analýzu a modelování (s použitím UML). Jeho modelem životního cyklu je inkrementální vývoj. Přesto nereflektuje zcela přesně na postupy označované jako rychlé prototypování. V reakci na tuto skutečnost vznikl koncem 90. let v akademickém prostředí Extreme Programming. Modelem životního cyklu je mu evoluční prototypování a používá postupy, které boří některé zažité představy. Nejmladší metodou, jež byla poprvé publikována v roce 2000, je Model Driven Architecture. Je založena na tvrzení, že i samotný programový kód je modelem. Všechny modely, které v průběhu celé tvorby systému vznikají, modelují totéž. Cílem proto je, místo tvorby nového modelu, použít model původní a pouze ho transformovat – z analytického modelu až do programového kódu. Samozřejmě bez zásahu člověka nebo pouze s jeho minimálním přispěním. Převážná většina postupů používaných touto metodou zatím neopustila akademické prostředí a její široké nasazení nelze v brzké době očekávat. Jedná se však o slibný koncept do budoucna.

Samotným prvkem, který je všem metodám společný (až na přístup) a který z nich lze vyčlenit, je analýza požadavků. Jejím cílem je shromáždit požadavky zákazníka na systém a předat je do dalších fází tvorby, jak definuje Kovitz v [1]. Zde také vznikají chyby s nejhorším dopadem. Plynou z nepochopení požadavků zákazníka. Výsledkem je, že implementovaný systém neodpovídá jeho představě. Tento druh chyb se nesnadně odstraňuje – jedná se o změnu funkčnosti systému – a většinou způsobuje právě překročení rozpočtu nebo nedodržení termínu, v horším případě obojí. Práce se proto orientuje na zjištění těchto kritických požadavků.

3 KRITICKÉ POŽADAVKY NÁVRHU

Při hledání kritických požadavků návrhu jsme se nejdříve zaměřili na některé z typických problémů vývoje softwaru v České republice. Informace byly získány během rozhovorů s vývojovými týmy a inspiraci poskytl i Paleta v [2]. Jednotlivé problémy jsem shrnuli do následujícího seznamu:

- syndrom jedné místnosti
- tvorba nepoužitelných modelů
- fluktuace pracovníků
- odhad pracnosti a plánování

- neznalost požadavků zákazníka

V pozadí většiny problémů stojí velikost vývojových týmů v České republice. Většina metod řízení a praktické tvorby předpokládá tým o cca 35 členech, typický český tým má méně než 10 členů. Další překážkou je předávání informací uvnitř týmu a především mezi týmem a zákazníkem. Zlepšení komunikace by měl přinést produkt na bázi Knowledge Managementu. Jeho hlavním přínosem by mělo být odstranění redundance dotazů a schopnost využít informace získané z jednoho projektu v projektu jiném.

Zkoumali jsme též přínosy nasazení informačního systému z pohledu zákazníka. Souhrnem 19 projektů implementace 6 různých informačních systémů typu ERP byly vytvořeny následující oblasti, v nichž jsou přínosy nejmarkantnější a finančně měřitelné:

- snížení zásob
- zlepšení logistiky (dosažení Just-in-time)
- efektivní plánování
- zjednodušení administrativy

Na základě těchto průzkumů označujeme jako kritické požadavky návrhu takové požadavky, které zákazníkovi brání v dosažení výše uvedených přínosů. Zároveň jsou to ty požadavky, které zákazník nesdělí vývojovému týmu. Klíčovou se tedy stává schopnost získání těchto kritických požadavků od zákazníka. Jednou z možností jak je získat, je metoda dotaz-odpověď. Rozhodujícím faktorem tak je kladení správných otázek.

4 ZÁVĚR

Cílem práce je vytvoření metodologii pro nalezení těchto správných otázek vedoucích k odhalení kritických požadavků návrhu. A dále její kompozice s uvažovaným systémem na bázi Knowledge Managementu. Budeme se též snažit definovat nároky kladené na takovýto systém a funkčnost, jež by měl poskytovat. Na jejich základě pak posoudíme vhodnost existujících systémů pro námi uvažované použití, případně navrhneme jejich rozšíření či vzájemné propojení. Následně provedeme hodnocení z hlediska realizační náročnosti.

PODĚKOVÁNÍ

Tento příspěvek vznikl za podpory grantového projektu Výzkum a implementace metod znalostního managementu pro vývoj a údržbu software (1ET409980417) programu Informační společnost AV ČR a inženýra Zdeňka Opršala.

LITERATURA

[1] Kovitz, L.: Practical Software Requirements. Greenwich, Manning Publications, 1, 1999, ISBN: 1-884777-59-7.

[2] Paleta, P.: Co programátory ve škole neučí, aneb softwarové inženýrství v reálné praxi. Brno, Computer Press, 1, 2004, ISBN: 80-251-0073-1.

[3] Maciaszek, L., Liong, B.: Practical Software Engineering, A Case Study Approach. Harlow, Addison-Wesley, 1, 2005, ISBN: 0-321-20465-4.