4IT421 – Zlepšování procesů budování IS

Semestrální práce

Implementace metodiky Rational Unified Process v systému Unicorn Universe

12. 12. 2012

Jméno a příjmení	Mészáros Maroš, Oldřich Hanuš
Xname	xmesm07, xhano07
Obor	Informační systémy a technologie
Studijní plán (C,D,E)	E

Obsah

1. Úvod

Vývoj software je poměrně složitým procesem, který v dnešní době vyžaduje velice širokou škálu dovedností. Projekty se stávají čím dál více komplexnějšími, takže již nestačí jen pouhá znalost programování. Pro zajištění úspěchu je důležité si zvolit vhodnou metodiku, což je předpis osvědčených pravidel a postupů, které by nás měly daným procesem provést. Existuje celá řada nejrůznějších metodik zabývajících se vývojem software. Některé z nich jsou poměrně odlišné a vhodné pro zcela rozdílné typy projektů, některé se naopak do značné míry překrývají. Jednou z takových metodik je Rational Unified Process, kterou úspěšně implementovala společnost Unicorn a. s., a kterou do značné míry používá při řízení ve svém informačním systému Unicorn Universe. Tato práce má za cíl představit a demonstrovat aplikaci metodiky Rational Unified Processes v informačním systému Unicorn Universe a tím zlepšit orientaci možných budoucích pracovníků, kteří s tím to systémem přijdou do styku. V závěru práce vzniklo také porovnání informačního systému Unicorn Universe z hlediska řízení projektů oproti řízení projektů ve vývojovém nástroji Rational Team Concert.

2. Metodiky vývoje software – tradiční a agilní

Jak již bylo řečeno, metodika je předpis osvědčených pravidel a postupů (best practices), které nás provází celým procesem vývoje software. Metodiky lze rozdělit do dvou kategorií - agilní a tradiční.

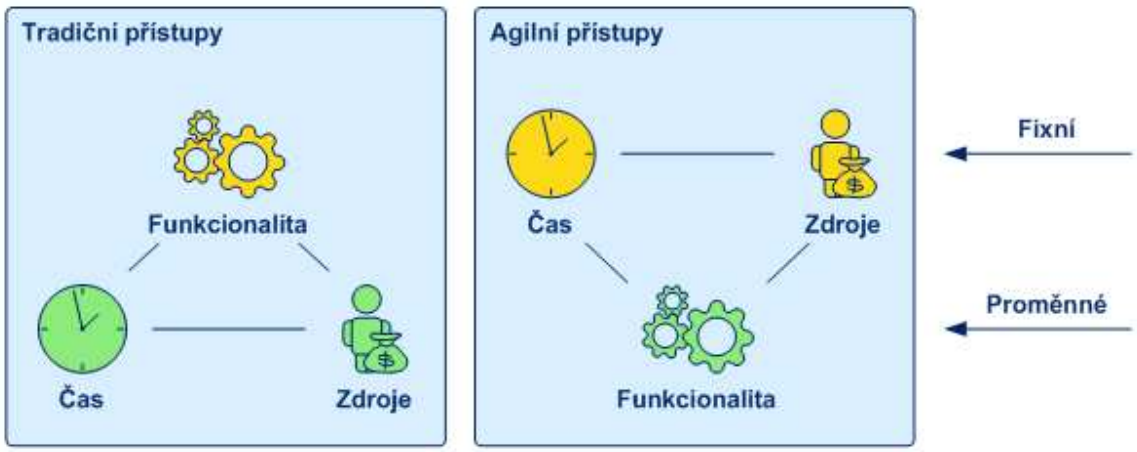
2.1 Agilní přístup

Agilní metodiky jsou založeny na skutečnosti, že nejlepší možný způsob k dosažení cíle je co nejrychleji vyvinout a dodat software zákazníkovi (i s omezenou funkčností). Délky jednotlivých iterací jsou proto plánovány na co nejkratší dobu (může být i jeden den). S tím je spojeno také průběžné a neustálé testování, které produkují velký počet nových verzí. Produkt se pak následně bude dle připomínek zákazníka upravovat. Velký důraz je kladen na interakci se zákazníkem. Zákazník proto vstupuje do celého procesu jako klíčová role, se kterou je potřeba úzce komunikovat a případně ho do procesu aktivně zapojit. Zákazník tak má vývoj více pod kontrolou a v případě potřeby může pružněji reagovat. Právě špatná komunikace bývá častou příčinou neúspěchu mnoha projektů. Proto agilní metodiky považují komunikaci za součást procesu vývoje. Mezi zástupce agilního přístupu můžeme jmenovat metodiku SCRUM, Extreme Programming, či Lean Development.

2.2 Tradiční přístup

Tradiční přístupy zastávají filosofii neměnných požadavků od zákazníka na funkčnost. Je potřeba jen splnit zadanou specifikaci. Procesy se snaží pokrýt celý životní cyklus softwarového projektu (komplexnost), přičemž iterativní cyklus převažuje nad postupným (vodopádovým). Je zde patrná vysoká organizovanost a systematičnost v přístupu k tvorbě software (na rozdíl od agilních metodik). Terminologie je objektově orientovaná a proces je zaměřen na trvalé zlepšování. Mezi zástupce tradičního přístupu můžeme jmenovat metodiku Rational Unified Processes, starší a zdarma Unified Processes, či Microsoft Solutions Framework.

Níže lze vidět obrázek s porovnáním jednotlivých přístupů z hlediska času, zdrojů a funkcionality. Tyto základní proměnné hrají důležitou roli při vývoji software.

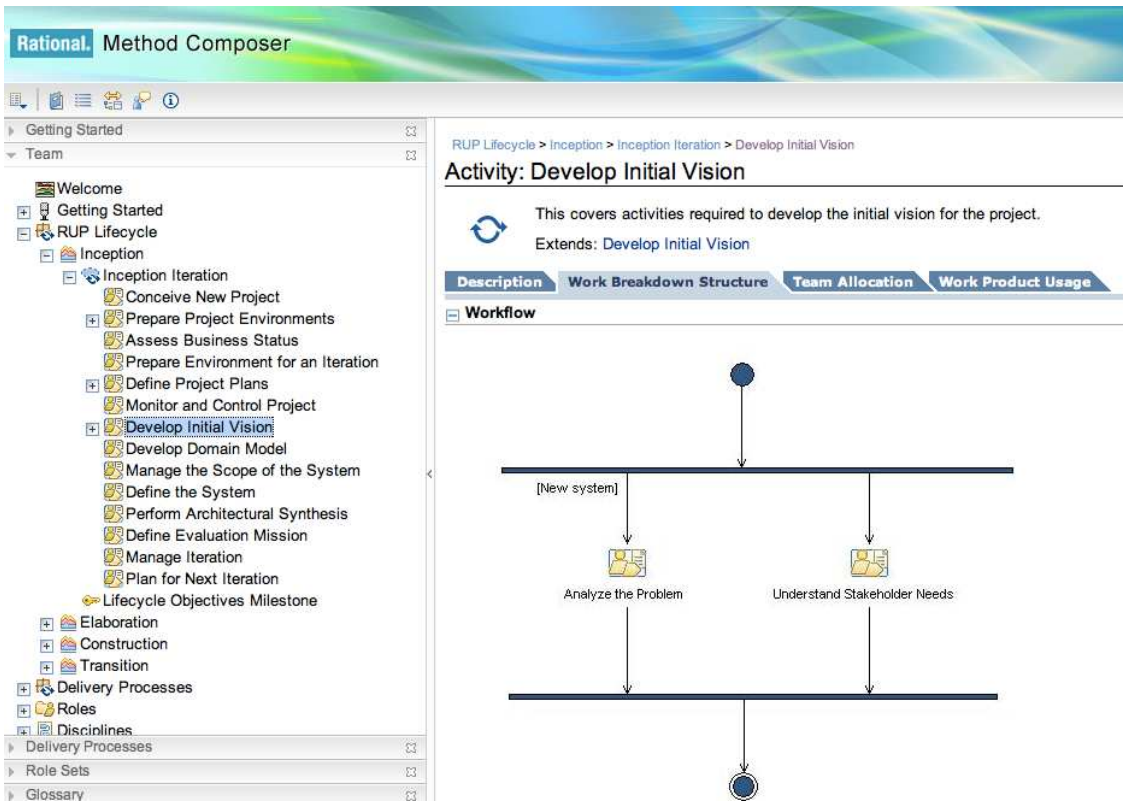


Obrázek 1: Porovnání metodik
Zdroj: Interní materiál Unicorn a.s.

Klasické přístupy většinou považují za neměnné (fixní) požadavky zákazníka na funkčnost. Naopak jako proměnné zde vystupuje čas a použité zdroje. Podle potřeby lze například zapojit do projektu větší či menší počet vývojářů, atd.
Naopak agilní přístupy považují čas a zdroje za neměnné – do vývoje nelze zapojit více zdrojů, čí prodloužit termín. Termín je nutné stanovit na začátku projektu, stejně jako zdroje (například počet vývojářů). Jako proměnná zde vystupuje funkcionalita – ta se v průběhu vývoje mění a přizpůsobuje. Některé agilní metodiky tyto tři základní proměnné modifikují, případně dále rozšiřují.

3. O metodice Rational Unified Process

Metodika RUP je zástupcem tradicích metodik. V případě agilních metodik se může zdát, že proces vývoje software je do jisté míry chaotický. U tradičních metodik, zvláště pak u RUP, je tomu naopak – zde je proces vývoje velmi systematický a organizovaný. Rational Unified Process představuje detailně propracovaný a zdokumentovaný přístup k vývoji software. RUP je poměrně rozsáhlá metodika založená na principu iterativního vývoje. Metodika bývá nejčastěji dodávaná jako rozsáhlá nápověda (znalostní báze) v podobě webových stránek.

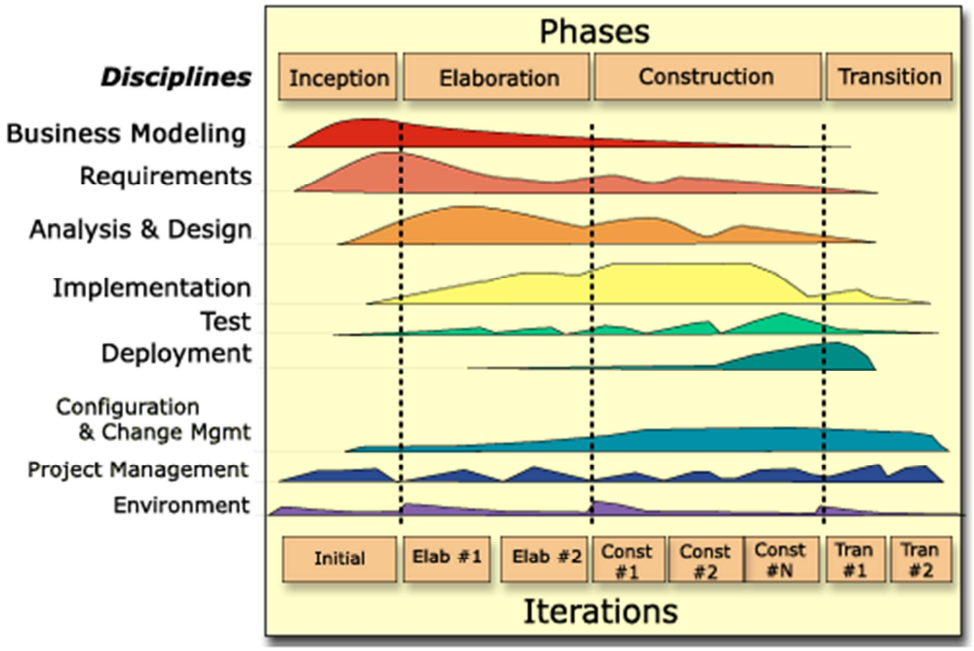


Obrázek 2: Ukázka znalostní báze RUP
Zdroj: Ukázka znalostní báze RUP (rup.unicorncollege.cz)

Tato online nápověda představuje rozsáhlou znalostní bázi, kde lze kdykoliv, v průběhu životního cyklu vývoje software, získat potřebné informace a zjistit, jaké kroky budou následovat.

RUP podrobně odpovídá na všechny otázky související s procesem tvorby software. Řeší čtyři základní problémy: **kdo**, **co**, **kdy** a **jak** je potřeba dělat v průběhu celého životního cyklu. Cílem této metodiky je vývoj kvalitního software, který splňuje požadavky zákazníka (kvalita a kvantita), má predikovatelný harmonogram (termíny) a nepřekročí stanovený rozpočet.

Metodika rozděluje životní cyklus procesu vývoje software do dimenzí, které se dělí na věcné (disciplíny) a časové (fáze). Každá věcná dimenze musí k závěru jedné časové fáze dodat odpovídající výstupy.

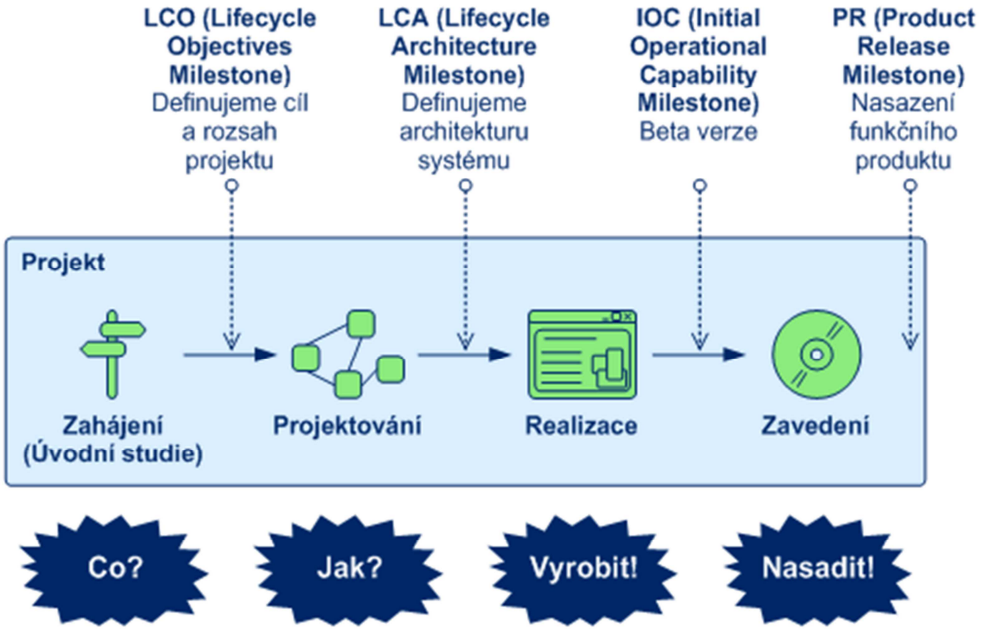


Obrázek 3: Schéma časových a věcných dimenzí RUP
Zdroj: http://era.nih.gov/docs/rup_fundamentals.htm

Každá disciplína se zaobírá jiným pohledem na software a řeší svou specifickou oblast.

Obchodní modelování se zaměřuje na pochopení prostředí a obchodu zákazníka.
Požadavky se zaměřují na potřeby zákazníka, které je nutné zanést do IS.
Analýza a design rozkreslují podobu IS do modelů pro programátory.
Implementace se zaobírá fyzickým vývojem software.
Testování hledá chyby a testuje schopnost IS.
Nasazení připravuje umístění systému do plného provozu

Konfigurace řídí konzistenci projektu a zanášení změnových požadavků a tvorbu dokumentace.
Projektový management zabezpečuje řízení projektu z hlediska lidských zdrojů, fází, iterací, rizik a podobně.
Prostředí je disciplína zabírající se zajišťováním potřebných nástrojů, vybavení, informačních zdrojů a definic vnitropodnikových standardů pro vývoj.
Časové dimenze, kterými postupně všechny věcné disciplíny procházejí, řeší jiný hlavní cíl vývoje.



Obrázek 4: Schéma časových dimenzí RUP a jejich milníky.
Zdroj: Interní materiál Unicorn a.s.

V první fázi zvané “Zahájení” je základem především zodpovězení na otázku, co se bude vyrábět a jaká jsou hlavní rizika. Ve fázi “Projektování” se primárně řeší, jak bude vyvíjený software fungovat po stránce architektury. Další fáze “Realizace” dle zpracovaných analýz vyrobí software, který se v poslední fáze “Zavedení” nasadí.

4. Představení IS Unicorn Universe

Unicorn Universe je informační systém, který je postaven na technologiích Java Enterprise Edition. Poskytuje moderní třívrstvovou architekturu. Datová vrstva je postavena na technologiích Oracle. Aplikační server je implementován za použití Apache Tomacat. K službám systému lze přistupovat přes vystavované API veřejných služeb.
Systém původně vznikl pro vnitropodnikové účely společnosti Unicorn a.s. v oblasti řízení a správy. Jeho filozofie návrhu však již od počátku umožňovala vysokou modularitu a customizovatelnost. Základ systému je tak ve všech modulech a implementacích pořád stejný. Jednotlivé distribuce systému jsou pak pomocí metodiků společnosti přizpůsobovány business procesům zákazníka.

Obrázek 5: Hierarchie platformy Unicorn Universe
Zdroj: Interní materiál Unicorn a.s.

Universe

Pojmem Universe lze chápat platformu pro provoz mnoha informačních systémů (osobních i podnikových) umožňujících ukládání a řízení informací. Universe je dělen na jednotlivé sektory (instalace Unicorn ES do oddělených databází), které obsahují teritoria. Velikost Universe tak není limitována velikostí jedné databáze.

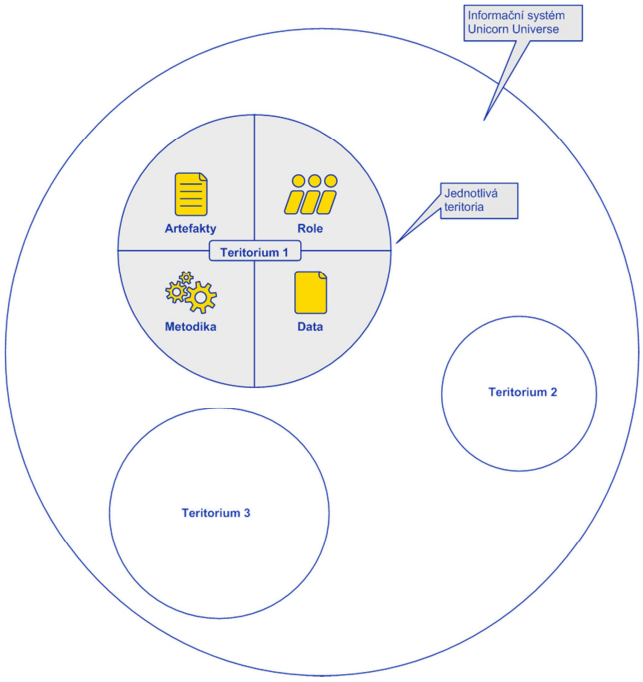
Sektor

Je logická jednotka reprezentující jedno nasazení systému Unicorn ES s následujícími vlastnostmi: běží na něm jedna verze systému a má jednotně definované SLA. Většinou ho tvoří primární DB a sada aplikačních serverů (na všech běží jedna verze systému). Sektory mohou mezi sebou komunikovat a jeden sektor obsahuje 0, až N teritorií.

S pojmem sektor se normální uživatel sice nesetká, ale tento pojem je důležitý zejména kvůli faktu, že některá nastavení systému jsou platná v rámci celého sektoru.

Teritorium

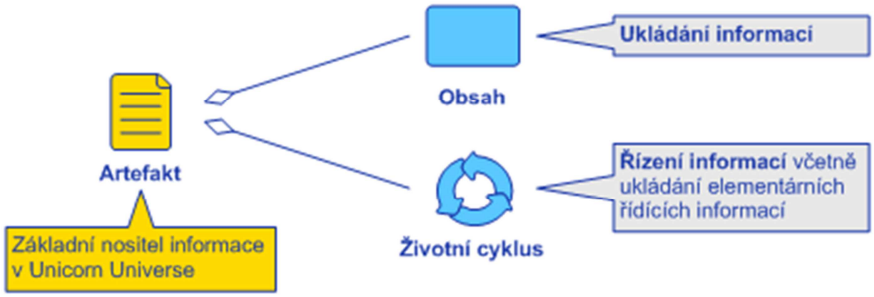
Je nezávislá logická oblast pro provoz podnikových nebo osobních informačních systémů. Uživatel pracuje s pojmem teritorium - ví, v jaké teritoriu se nachází, do jakých teritorií má přístup a podobně. Konkrétní uspořádání teritorií v rámci Unicorn Universe zobrazuje následující obrázek. Každý uživatel Unicorn Universe má My Territory, což je osobní systém uživatele, do kterého má přístup pouze on a jeho přátelé, kterým to dovolí.
Vedle privátních teritorií jsou zmiňována v rámci Unicorn Universe tzv. Business Territory jednotlivých podniků. Do nich má přístup pouze ten, komu to kompetentní osoba za teritorium dovolí.



Obrázek 6: Ukázka rozdělení systému
Zdroj: Vlastní tvorba: za použití šablon UUBML

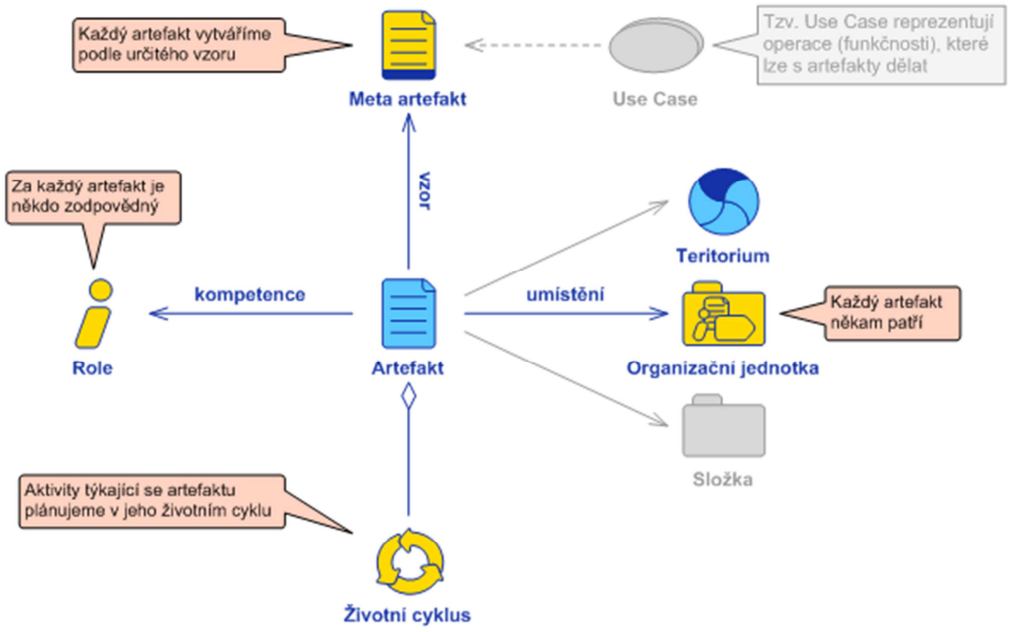
Artefakt

Základním nositelem informace je v Unicorn Universe artefakt. Jeho klíčovou vlastností je schopnost v sobě jakoukoliv informaci nejen ukládat, ale současně i řídit.



Obrázek 7: Dvě základní funkce artefaktu
Zdroj: *KOVÁŘ, Vladimír. Unicorn ES Powered Company: Management, 2009*

Artefakt má vedle obsahu a životního cyklu i některé další vlastnosti, které pomáhají informace řídit, a které obyčejné soubory nemají.



Obrázek 8: Řídící model artefaktu
Zdroj: *KOVÁŘ, Vladimír. Unicorn ES Powered Company: Management, 2009*

Meta artefakt – Každý artefakt je vytvořen podle šablony (podle meta artefaktu), která definuje vzor struktury, obsahu a životního cyklu – tzn., definuje typ informace, kterou artefakt obsahuje. Tak je zajištěno, že výstupy stejného typu budou mít stejný věcný obsah, formu a životní cyklus.

Kompetence – Za každý artefakt je vždy kompetentní (zodpovědná) právě jedna role. Artefakt, který nemá definovanou zodpovědnou roli, neexistuje. Kompetentní role je zodpovědná za správu artefaktu, správu přístupových práv a správu životního cyklu.

Životní cyklus – Informace, které se týkají konkrétního artefaktu, jsou řízeny přímo v životním cyklu daného artefaktu. Tím je zajištěno, že informace, které jsou uvedeny v artefaktu, jsou na stejném místě i řízeny.

Umístění artefaktu – Každý artefakt je vždy umístěn do organizační jednotky, ke které náleží (v ní může být umístěn do složek).

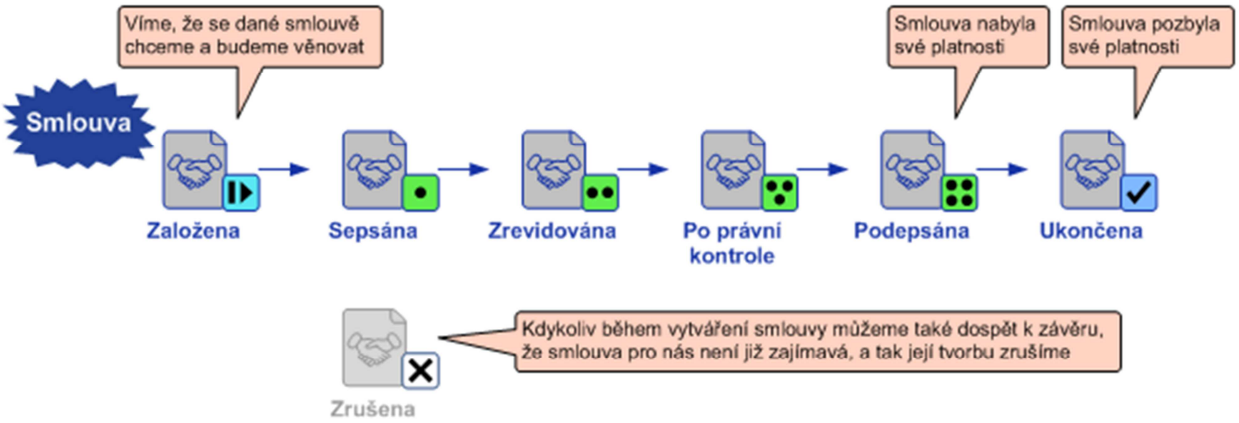
Přístupová práva – Na artefakt mají přístup pouze takové role, které na ně mají ze své pozice právo a potřebují ho pro svou práci (toto vychází z implicitních, popř. explicitních práv). Přístupová práva k artefaktům jsou definována vzhledem k rolím a funkčnostem, které je možné spouštět na daném artefaktu.

Životní cyklus

Každá informace se v čase vyvíjí, prochází různými stavy. Stejně tak každý artefakt, když prochází svým životním cyklem, prochází různými stavy. Některé trvají krátkou dobu, některé stavy jsou dlouhodobější. Je vhodné, aby dlouhodobější stavy informace odrážely stav artefaktu. Tím je zajištěno, že každý uživatel, který s artefaktem pracuje, ví, v jakém stavu se daná informace nachází.

Každý artefakt má nastaven svůj určitý stav. Stav artefaktu odráží stav informací, které jsou v něm uloženy. Vyjadřuje míru rozpracovanosti jeho obsahu. Pokud artefakt reprezentuje nějaký reálný objekt, informuje stav artefaktu o stavu tohoto objektu.

Například při práci se smlouvami se sledují tyto stavy: zda se o obsahu smlouvy teprve jedná, zda je smlouva sepsána, zda byla zrevidována, zda prošla právní kontrolou, zda je podepsána (a tedy vstoupila v platnost) nebo zda je ukončena, zrušena (a tedy neplatí) apod.



Obrázek 9: Důležité stavy, které se sledují při tvorbě smluv
Zdroj: *KOVÁŘ, Vladimír. Unicorn ES Powered Company: Management, 2009*

Aktivita

Nad každým artefaktem lze vytvořit aktivitu, která má svého zadavatele a řešitele. Aktivitu lze založit pouze nad artefaktem. Aktivita je elementární činnost, u které nepředpokládáme další rozpad, tj. jejich řešení trvá nanejvýš v řádu hodin. Tyto aktivity mají několik výchozích typů:

Udělej - Používá se pro řízení jednoduchého úkolu v řádově v rámci několika hodin a je určen jednomu řešiteli. Vhodný pro jakoukoliv běžnou činnost

Zpráva - Jednorázově notifikuje řešitele o dění nad artefaktem a nevyžaduje po něm žádnou činnost, ani rozhodnutí.

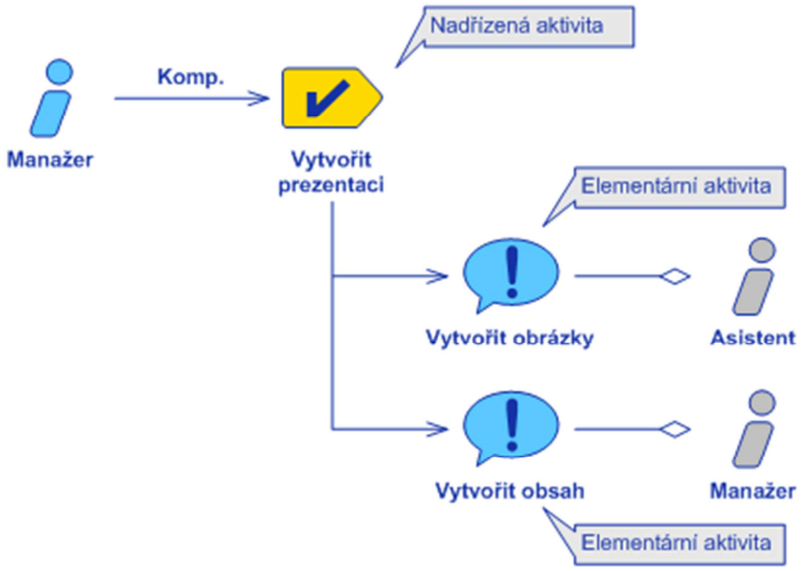
Odpověz - Vyžaduje po řešiteli dodání informace.

Rozhodni - Vyžaduje se rozhodnutí se zadavatelem předem definovaných variant. Existují již předdefinované varianty pro běžné úkoly jako schvalování plateb a podobně.

Rezervuj si čas - Vyžaduje se rezervace času v spojení s artefaktem, například na nějakou schůzi.

Nadřízená aktivita

Nadřízené artefakty se skládají z jednoduchých aktivit popsaných výše. Agregují je tím do složitějších úkolů, které můžou mít i více řešitelů. Vyřešením všech dílčích úkolů se vyřeší nadřízená aktivita.



Obrázek 10: Nadřízená aktivita
Zdroj: Interní materiál Unicorn a.s.

Nadřízení aktivity je možné do sebe vnořovat, ale v tomto bodě se ze strany uživatele vyžaduje soudnost, protože příliš strmý hierarchický strom by řízení velmi komplikoval. Také nadřízené aktivity, podobně jako aktivity jednoduché, mají několik typů:

Úkol - Řídí ho jeden zadavatel a agreguje více jednoduchých aktivit s více řešiteli. Je odpovědný za to, aby úkol vyskládal tak, že řešení aktivit integruje řešení úkolu.

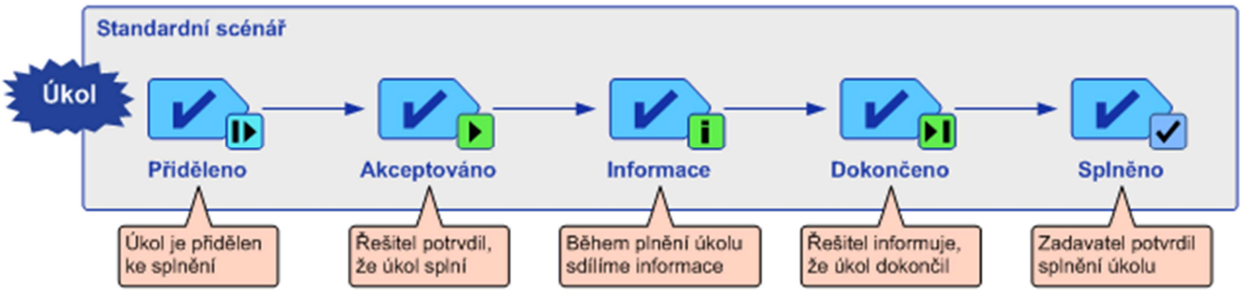
Milník - Vytyčuje nějaké významné body. Jednoduše vyznačí, co musí být v daný okamžik splněno. Podřízené aktivity milníku slouží pro ověření, zda byl stanovený milník dosažen. Aktivity, které však k jeho dosažení vedou, se plánují zvlášť (Milník jim není nadřízen).

Schůzka - Používá se pro organizaci schůzky. Řešitel je odpovědný primárně za organizaci schůzky, osobně se jí účastnit nemusí.

Nadřízená aktivita - Pouze sdružuje jiné úkoly pod jednotnou množinu a umožňuje jejich hromadnou správu, jako například ukončení, změnu řešitele a podobně. Podřízené aktivity jsou typicky typu: Rezervuj si čas.

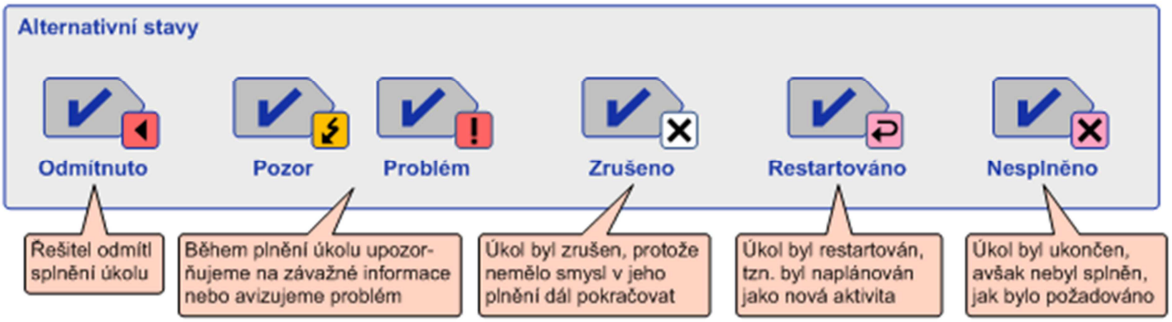
Stavy aktivit

Všechny aktivity, nezávisle na jejich typu, mají několik stavů, do kterých můžou vstupovat. Nastavování stavů je základním způsobem řízení v IS Unicorn Universe. Stavy jsou buď základní, nebo když se něco nezdaří, tak alternativní.



Obrázek 11: Základní scénář pro plnění úkolu
Zdroj: *KOVÁŘ, Vladimír. Unicorn ES Powered Company: Management, 2009*

Základním stavem při vytvoření aktivity je “Přiděleno”, kdy se aktivita vyobrazí v úkolovníku řešitele s jejím popisem a termínem vyřešení. Když ji řešitel uvidí a posoudí, že má všechny potřebné kompetence k jejímu vyřešení, nastaví stav na “Akceptováno”. Kdykoliv chce řešitel na něco upozornit, může nastavit aktivitu do stavu Informace, běžně to ale není nutné. Po dokončení se nastaví příslušný stav, který zadavatel schválí nastavením stavu “Splněno”. V takovém stavu je aktivita již ukončena a neměla by přecházet do dalších stavů.

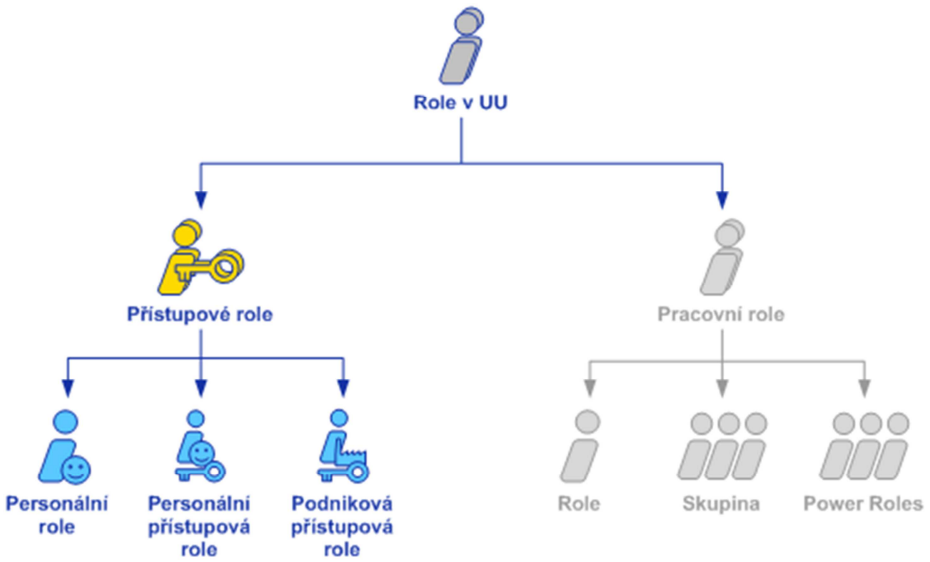


Obrázek 12: Alternativní stavy, které mohou nastat při plnění úkolu
Zdroj: *KOVÁŘ, Vladimír. Unicorn ES Powered Company: Management, 2009*

Protože ne vždy jde řešení aktivity snadno, existují i další stavy vyjadřující problémy při řešení, odmítnutí úkolu nebo jeho nesplnění.

Role

Přístup uživatelů do jednotlivých teritorií je zajištěn pomocí tzv. přístupových rolí (Access Roles). Každá přístupová role představuje smlouvu - dohodu o tom, za jakých podmínek bude uživatel využívat dané teritorium. Přístupové role jsou speciální druhy rolí a platí pro ně odlišná pravidla než pro pracovní role.



Obrázek 13: Role v systému
Zdroj: Interní materiál Unicorn a.s.

Typy přístupových rolí jsou:

Personální role - Je obrazem uživatele v systému UU a má napříč celým systémem jednotné unikátní ID. Zprostředkuje základní přístup do systému.

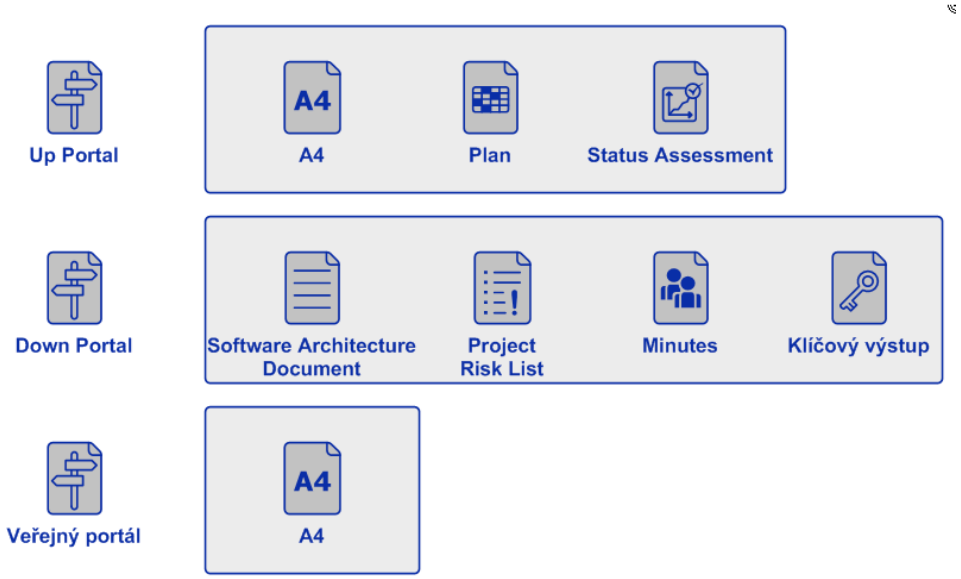
Personální přístupová role - Zprostředkovává přístup do teritorií a váže se k personální roli. V rámci teritoria je obsazována do pracovních rolí odpovědných za plnění úkolů.

Pracovní přístupová role - Pokud je nutné pro splnění úkolu přístup do jiného teritoria, vytvoří se pracovní role. Tu lze obsadit pouze na jednu konkrétní pracovní roli v rámci cizího teritoria.

5. Metodika řízení projektů v Unicorn Universe

Společnost Unicorn a.s. využívá k řízení projektů metodiku RUP. Dalo by se říct, že je celá společnost na této metodice postavená. Z výše zmíněných logických součástí systému (artefakt, role…) lze vytvořit základ pro implementaci různých metodik. Z dynamického pohledu rozumíme metodikou sadu činností, jejichž cílem je nastavení systému. Ze statického pohledu zase množinu informací zaznamenaných v systému, jejichž prostřednictvím je systém nastavován.

Portál projektu



Obrázek 14: Ukázka rozdělení systému
Zdroj: Interní materiál Unicorn a.s.

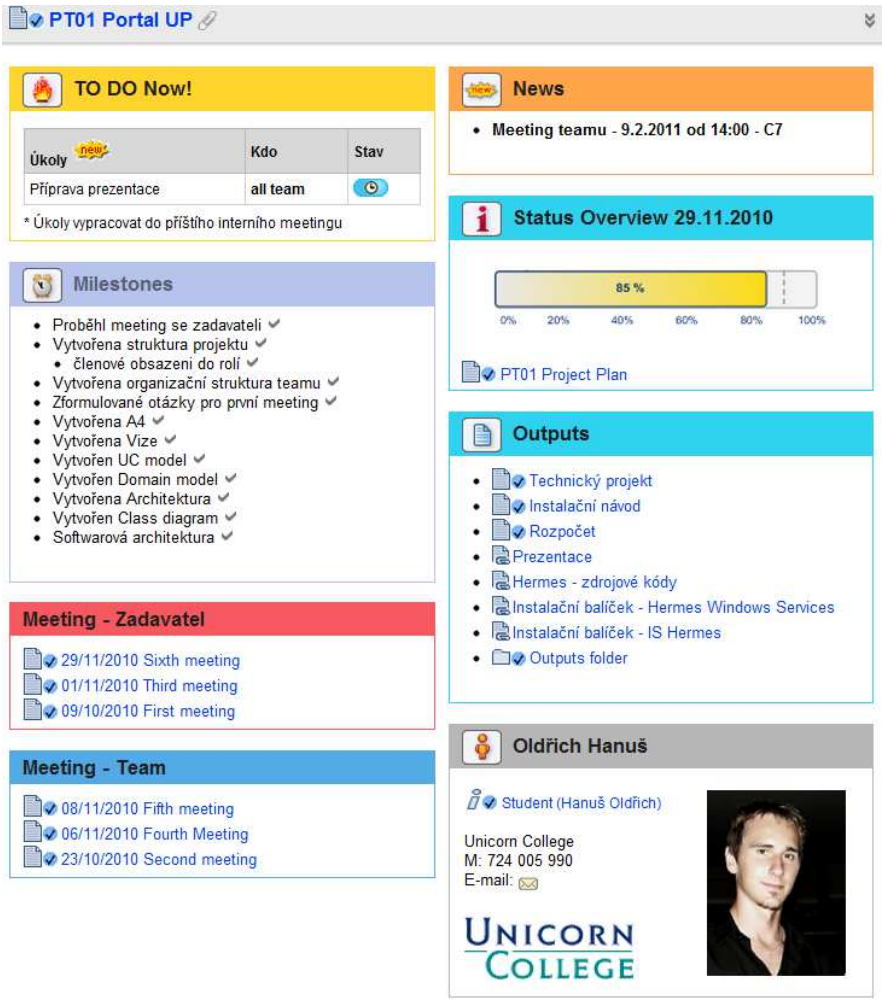
Zastřešuje projekt. Portál obsahuje odkazy na veškerou dokumentaci a artefakty, které v průběhu práce na projektu vzniknou. Jedná se v podstatě o výchozí rozcestník manažera jednotky a jeho teamu. Shrnuje a poskytuje důležité řídící informace z hlediska kvality, kvantity termínu a rozpočtu, pojmenovává problémy a rizika.

Fáze projektu

Zde jsou popsány jednotlivé fáze projektu dle metodiky RUP.

Inception

Počáteční fází se zahajuje projekt. Tuto fázi provází sběr požadavků, na základě kterého se určuje předmět a celkový rozsah projektu. V této fázi je vytvořen projektový portál (viz obrázek 15), ze kterého jsou postupně odkazovány důležité dokumenty. Projektový portál tvoří obvykle vedoucí projektu. Prvním výstupem fáze určeným pro zákazníka je dokument A4 a Úvodní studie.



Obrázek 15: Ukázka projektového portálu
Zdroj: Vlastní tvorba:ukázka portálu ze systému Unicorn Universe

Document A4

Jak již z nadpisu vyplývá, dokument má rozsah maximálně na jednu stránku formátu A4. Cílem je v takhle kompaktní podobě popsat základní aspekty projektu. Popisuje prostředí zákazníka a jeho problém. Následně nastíní, jak bude problém řešen prostřednictvím informačního systému. Umožňuje tak velice rychlý náhled do problematiky, pro rychlé obeznámení personálu obsazovaného do projektu a také pro stranu zákazníka, aby se ujistil, že se bude skutečně řešit to, co potřebuje. Protože dokument musí být srozumitelný i netechnicky zaměřeným manažerům, musí být psaný srozumitelně a obecně. Dobrou praktikou je zahrnutí lehce uchopitelného schématu.

Úvodní studie

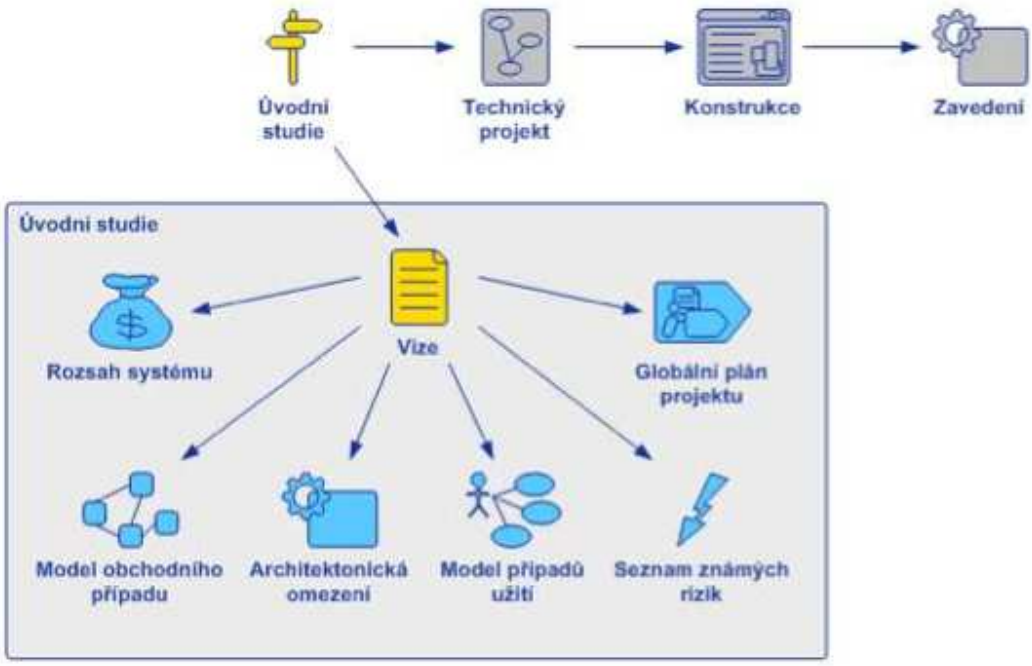
Obsahuje základní definice budovaného IS a slouží jako základ elaborační fáze. Musí obsahovat rozsah implementovaného systému. Dále v něm v něm musí být jasně napsané, které aspekty projekt řeší a také které aspekty už ne.

Obsahuje také podrobný popis fungování obchodu zákazníka a v jakém prostředí je situován. Musí analytikům dát jasně vědět, v čem leží základ podnikání zákazníka a kde vytváří svou přidanou hodnotu.

Od toho se také dále odvíjí popis modelu použití. Znamená to co lze v IS dělat, kteří uživatelé budou dané případy užití vykonávat a co je jejich cílem. K tomu se také přikládá hodnocení náročnosti a rizik pro implementaci daného případu použití. V této fázi ještě nemusí být model zcela kompletní. Jako postačující se uvádí 80% kompletnost vůči finální implementaci.

V další části dokumentu jsou také popsány architektonická rizika. V nich se uvádí, jaké je prostředí informačních technologií zákazníka a co z toho vyplývá. Příkladem mohou být zastaralé a uzavřené systémy bez veřejných služeb a z toho vyplývající problémy se synchronizací.

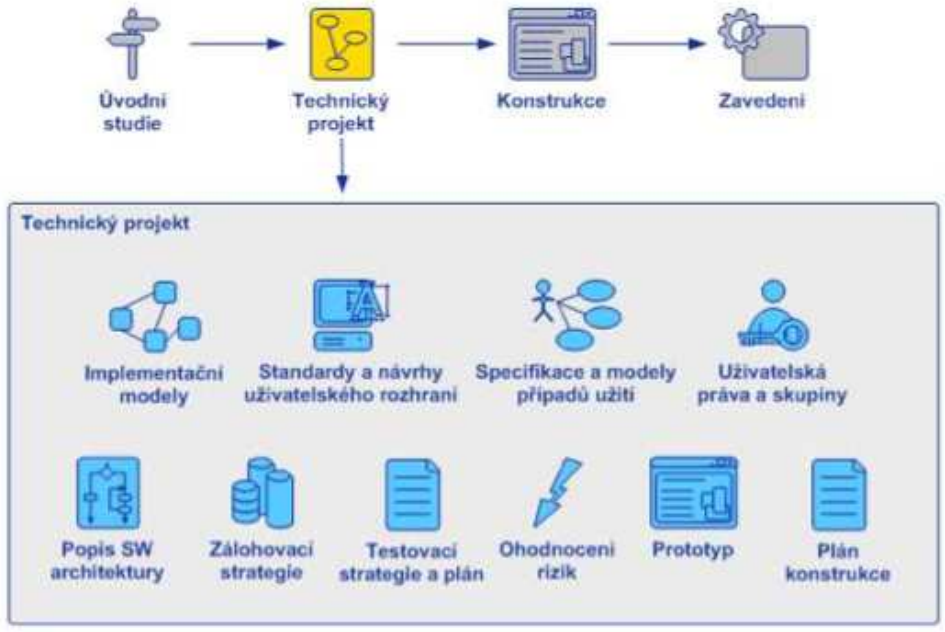
V závěru se shrnuje globální plán projektu se všemi fázemi a mezivýstupy, co bude a co nebude uděláno a za jaký čas se má co stihnout.



Obrázek 16: Základní části Úvodní studie,
Zdroj: Interní materiál Unicorn a.s.

Elaboration

V této fázi vzniká Technický projekt, který je klíčovou součástí elaborační fáze. Zaobírá se podrobným popisem toho, jak bude IS fungovat, na jaké architektuře bude vybudován, jak se budou eliminovat rizika, která jsou popsána v Úvodní studii, a také sleduje budoucí podobu IS z několika pohledů.



Obrázek 17 - Základní části Technického projektu,
Zdroj: Interní materiál Unicorn a.s.

První složkou Technického projektu jsou Implementační modely. Běžně se uvádí hlavně Třídní modely a Datové modely. Jejich podoba musí odpovídat současným specifikacím UML. Dále se alespoň pro některé části IS, pokud je to potřeba, dopracovávají také modely stavové, sekvenční, aktivitní a podobně.

Technický projekt dále obsahuje také standardy a návrhy uživatelského rozhraní. V něm jsou popsané obecné principy, jak se bude aplikace v určitých situacích chovat. Pokud je aplikace například postavena na formulářových vstupech, řeší se validace polí, jak se upozorní uživatel na špatný vstup a podobně. Uvádí se také, jak má aplikace vypadat, aby byly dodrženy grafické manuály zákazníka.

Dopracovává se zde také specifikace modelů a případů užití. Tato specifikace již musí být zcela kompletní a odsouhlasená zákazníkem. Jsou popisovány typy uživatelů, jaké bude IS mít a jaké úrovně oprávnění budou pro které funkčnosti dostupné.

Je nezbytné také podrobně popsat architekturu IS. Obecně se využívá nejvíce třívrstvová

architektura, složená s prezentační, aplikační a datové vrstvy. Ať už bude jakákoliv, musí být jasně specifikováno, jaké bude využívat technologie, jak budou jednotlivé komponenty komunikovat a které komponenty budou implementovat jaké skupiny funkcí. Musí se také uvádět, jak bude řešeno případné škálování a ochrana před výpadky.

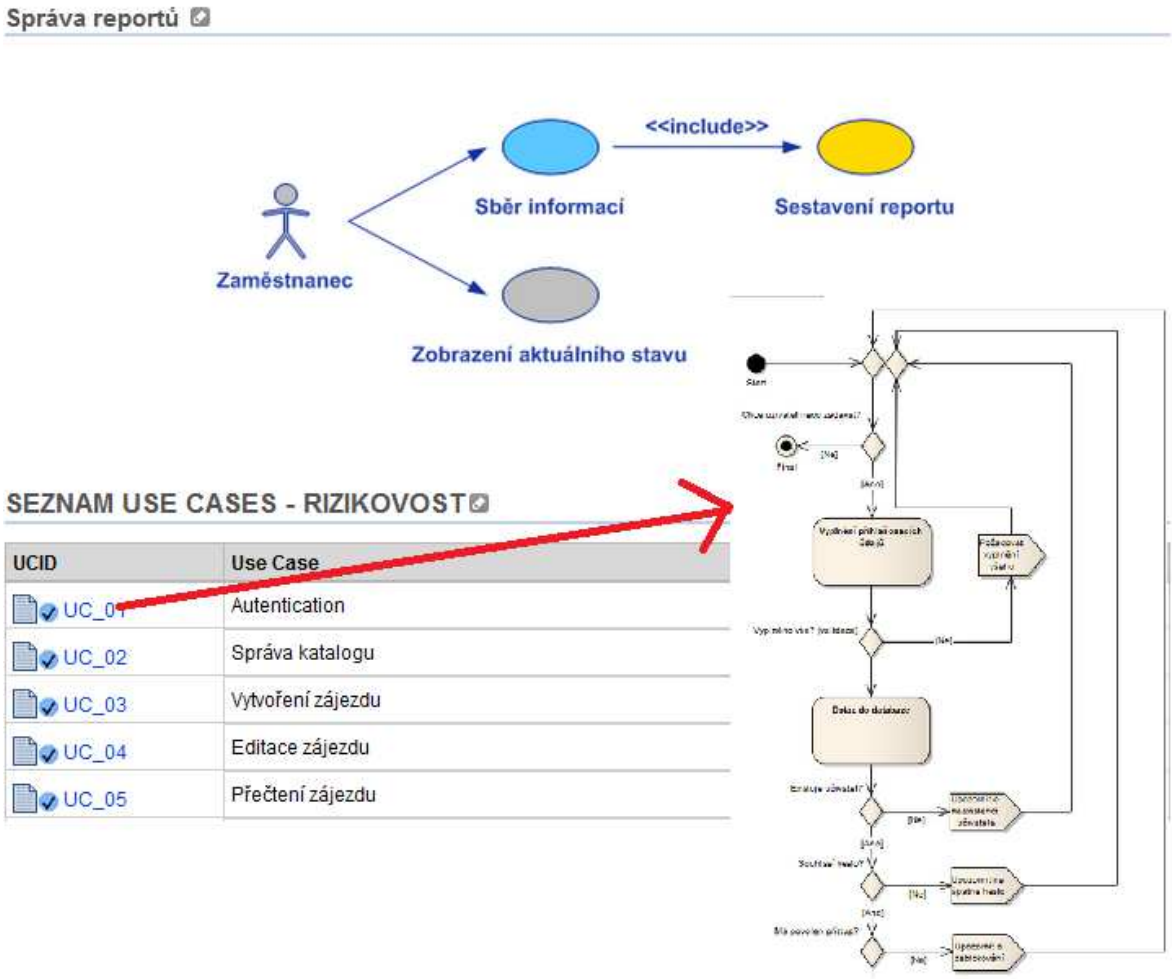
Na to navazuje část s popisem zálohování. Je zcela nepřípustné, aby zákazník přišel např. o seznam plateb klientů a podobně. Zálohovací strategie řeší jak a kdy se provádí částkové a celkové zálohy, kde a v kolika kopích se budou uchovávat a hlavně jak bude probíhat obnova do plného provozu v případě incidentu.

Po zohlednění funkcí a nároků na stabilitu a rychlost se v technickém projektu sestavuje testovací strategie a plán. Ten určuje rozsah testů od základních unit testů až po celkové testy při zátěži a poruchách. Design testů musí být navržen tak, aby se prověřily všechny nároky zákazníka na systém.

Samostatnou položkou je také seznam rizik. Ten uvádí všechna známá rizika, přiřazuje jim pravděpodobnosti výskytu, potencionální škody, které můžou napáchat a na základě toho také prioritu. U každého rizika se pak uvádí scénář řešení pro případ jeho vzniku.

Poslední složkou Technického projektu je plán konstrukce. Ten definuje počet iterací a také výčet funkcí, které se budou implementovat primárně. Definuje termíny jednotlivých iterací a cílové stavy aplikace po každé z nich.

K Technickému projektu se přikládá i prototyp aplikace. Ten běžně implementuje pouze základní funkce a i to běžně pouze jako skořápky bez skutečné logiky. Prototyp slouží hlavně pro zákazníka, aby se v ranních fázích před implementací identifikovaly případné problémy s použitelností.



Obrázek 18: Ukázka správy Případů použití
Zdroj: Vlastní tvorba

Construction

Unicorn považuje elaborační fázi za nejzdlouhavější a nejnáročnější. To ovšem neznamená, že se fáze konstrukce může podceňovat. Vzniká při ní nejdůležitější výstup vůbec, a tím je samotný informační systém. Tento systém však nevznikne najednou, ale iterativně a také po jednotlivých funkčnostech, což znamená inkrementálně. Každá iterace se skládá z části vývoje, následně testů, vyhodnocení a akceptace ze strany zákazníka.

Obrázek 19: Základní výstupy Konstrukce,
Zdroj: Interní materiál Unicorn a.s.

Ke každé iteraci tudíž vzniká, a dále se vyvíjí, několik výstupů. Prvním je rozpracovaná beta verze informačního systému. Pro ukončení iterace musí být tato verze spustitelná a tím pádem projít integračními testy. Z každé beta verze se archivují a v závěru projektu také odevzdávají, zdrojové kódy. Ty musejí být řádně dokumentované do takové míry, aby v kterékoliv iteraci mohl projekt teoreticky přebrat někdo jiný. To se samozřejmě nedělá, ale považuje se to za standard úrovně dokumentace. Ta musí obsahovat uživatelskou příručku, provozní dokumentaci, administrátorskou dokumentaci, programátorskou dokumentaci a instalační příručku. Na konci každé iterace a potažmo celé fáze konstrukce, jsou reportovány závěry testů. Na jejich základě se vyhodnocuje průběžný úspěch projektu. Průběžně se s každou iterací také upravuje plán nasazení. Ten popisuje, jak se má postupovat při nasazení systému do produkce. V pozdějších iteracích se také začíná provádět školení. Ta jsou určena pro připravované administrátory na straně zákazníka a také pro řadové uživatele.

Transition

Poslední fáze je fáze zavedení. U ní se počítá s tím, že její plán byl navržen v době konstrukce. Předmětem zavedení je plynulé spuštění a přechod na nový IS. Obvykle se to nestane najednou, ale nejdříve se začíná s pilotním provozem na omezené skupině uživatelů. Obvykle na úrovni jediného oddělení, pobočky a podobně. Největší úskalí jsou přenosy dat, synchronizace v době postupného zavádění, nebo zabezpečení neustálého provozu v době přechodu pokud nelze odstavit systémy ani na omezenou dobu.

Obrázek 20: Základní výstupy Zavedení,
Zdroj: Interní materiál Unicorn a.s.

Výstupem této fáze je samozřejmě IS. Ten je ale už v odladěné podobě, ne pouze beta, jako u konstrukce. Právě testování a pilotní provoz je to, co udělá z bety stabilního kandidáta k nasazení. Obvykle totiž až testování v pilotním provozu odhalí všechny chyby. V Zavedení také vznikne SLA, tj. dohoda o úrovni služeb, kde se definují přípustné doby výpadků, odezvy a přípustného zatížení, nebo doby obnovy po výpadku a také cena za servis a údržbu.
Zřídí se také primární a sekundární podpora. Cílem primární je řešit běžné problémy a dělat asistenci tápajícím uživatelům. Všechno, co je nad rámec, se přesměruje na sekundární podporu, která už může i zasahovat do kódu a opravovat chyby.

6. Řízení projektů v porovnání s Rational Team Concertem

Stejně jako Rational Team Concert, tak Unicorn Universe implementuje řadu funkčností pro řízení a podporu projektů, jako je plánování, správa požadavků v teamu, sledování transparentnosti a zdraví projektu, workflow management. Unicorn Universe je ovšem pouze informační systém, nikoliv vývojový nástroj. Pokud by se porovnával z hlediska podpory řízení projektů, tak je potřeba zmínit součásti, které Unicorn Universe oproti Rational Team Concertu neimplementuje. Mezi nejdůležitější součásti patří:

- kontinuální správou buildů
- verzování zdrojových kódů
- bugtrackingové systémy, jako je například JIRA

Z toho vyplývá, že zde není implementována integrace s ostatními vývojovými nástroji, které se při vývoji software používají. Systém se z hlediska řízení používá převážně na workflow management a správu dokumentace. Unicorn Universe podporuje pouze historickou správu dokumentů a dat za pomoci snapshotů. Nejedná se tudíž o produkt určený pro vývoj software, ale pouze pro organizaci a správu projektu z hlediska managementu.
Unicorn Universe je ovšem velice modulární systém, kde se dají jednotlivé funkčnosti navrhnout, implementovat a customizovat přímo na místě bez nutnosti programování.

Závěr

Tato práce měla za cíl nastínit aplikaci metodiky Rational Unified Process při řízení projektů v prostředí Unicorn Universe, a tím také zlepšit orientaci budoucích pracovníků při práci s tímto informačním systémem.

V úvodní části byl proto vysvětlen rozdíl mezi tradičním a agilním přístupem, což vytvořilo základ pro pochopení samotné metodiky Rational Unified Process a její aplikaci v informačním systému společnosti Unicorn.

Z konstrukce samotného systému lze odvodit, že systém je velice modulární a customizovatelný. Pokud se jeho metodika správně navrhne, lze se z hlediska workflow managementu a dokumentace vyrovnat osvědčeným nástrojům pro řízení projektů, jako je například Rational Team Concert od společnosti IBM.

Zdroje

[1] Concept: Lifecycle Objectives Milestone. [online]. 2006 [cit. 2012-12-12]. Dostupné z: http://rup.unicorncollege.cz/#core.base_rup/guidances/concepts/lifecycle_objectives_9E53DCF8.html

[2] VÍT, Stinka a Michal KÖKÖRČENÝ. Metodika vývoje software - jak se řídí projekt. [online]. 2010 [cit. 2012-12-12]. Dostupné z: www.unicornuniverse.eu Kód artefaktu UCL-BT:SDP.CZ/LEC01/PORTAL

[3] BUCHLÁK, Peter. Use Case diagram. [online]. 2010 [cit. 2012-12-12]. Dostupné z: www.unicornuniverse.eu Kód artefaktu UCL-BT:OAD11W.CZ/LEC04/PORTAL

[4] KIMR, David. Unicorn Universe - úvod. [online]. 2010 [cit. 2012-12-12]. Dostupné z: www.unicornuniverse.eu Kód artefaktu Unicorn Universe – úvod

[5] ŠIBIL, Jan a Petr CIOCHOŇ. Vrstvy informačních systémů. [online]. 2010 [cit. 2012-12-12]. Dostupné z: www.unicornuniverse.eu Kód artefaktu UCL-BT:ISA10W.CZ/LEC02/GL

[6] BLAŽOVÁ, Romana. KDO, CO a KDY v procesu testování. [online]. 2010 [cit. 2012-12-12]. Dostupné z: www.unicornuniverse.eu Kód artefaktu UCL-BT:TST11S.CZ/LEC02/GL

[7] KOVÁŘ, Vladimír. Integrace a vývoj. [online]. 2011 [cit. 2012-12-12]. Dostupné z: www.unicornuniverse.eu Kód artefaktu UCL-BT:CIT11W.CZ/LEC04/GL