Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Katedra informatiky a výpočetní techniky

Diplomová práce

Popis softwarových procesů použitelný pro nástroje řízení projektů

Plzeň, 2013 Petr Pícha

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne …….. …..……

Petr Pícha

Poděkování

Autor práce by tímto rád poděkoval:

Doc. Ing. Přemyslu Bradovi MSc., Ph.D za celkové vedení, motivaci, veškerou vstřícnost, asistenci, zpřístupnění nástrojů a údajů a poskytnuté informace,

Bc. Pavlu Kraftovi za revizi textů v popisu procesu předmětu KIV/ASWI z nástroje IBM Rational Method Composer

a studentům předmětu KIV/ASWI v akademickém roce 2012/2013 za zpětnou vazbu ohledně hlavních výstupů této diplomové práce.

Abstract

Title: Software process description usable for project management tools

Summary: This thesis deals with software developmentprocesses, their methodologies and patterns, options of their modeling and converting these models, or their substantial parts to a templateform suitablefor projectconfiguration and change management systems. In particular itexamines the options and potential of such models created in the IBM Rational Method Composertool and the possibility of extracting the created templates and importing them into theconfiguration and change management tool Rational TeamConcert also by IBM. The practical example on which these steps were tested and examined is the software development process used in the course Advanced Software Engineering at the Department of Computer Science and Engineering of the Faculty of Applied Sciences at the University of West Bohemia. The outputs of the thesis are the model of the process in HTML and PDF format and the template to be used in Rational TeamConcert instance of the department and should be used mainly as a guidance and supporting material for the students of the course.

Anotace: Tato diplomová práce se zabývá procesy vývoje software, jejich metodikami a vzory, možnostmi jejich modelování a převodu těchto modelů nebo jejich významných částí do formy šablon použitelných pro nástroje řízení projektů. Konkrétně zkoumá vlastnosti a možnosti těchto modelů vytvořených v nástroji IBM Rational Method Composera možnosti převodu těchto šablon do nástroje pro řízení projektůRational Team Concert rovněž od společnosti. Praktickým příkladem, na němž byly tyto kroky zkoumány a odzkoušeny je proces vývoje software užívaný v předmětu Pokročilé softwarové inženýrství vyučovaném na Katedře informatiky a výpočetní techniky Fakulty aplikovaných věd Západočeské univerzity v Plzni. Výstupy práce jsou model procesu ve formátu HTML a PDF a šablony pro užití v instanci Rational Team Concert katedry a měly by být sloužit především jako podpůrný materiál pro studenty předmětu.

Obsah

Seznam obrázků

Seznam tabulek

1. Úvod

Tato diplomové práce se zabývá procesy vývoje software (dále pouze procesy), jejich metodikami a vzory, jejich konfigurací a přizpůsobení danému konkrétnímu účelu, možnostmi jejich modelování v pokročilých specializovaných softwarových nástrojích a možností převoditelnosti těchto modelů do systému pro řízení projektů.

Praktickým výsledkem práce je vytvoření rozsáhlého popisu procesu užívaného týmovými projekty při vývoji softwarových systémů v rámci předmětu *Pokročilé softwarové inženýrství* (ASWI) vyučovaného na *Katedře informatiky a výpočetní techniky* (KIV) *Fakulty aplikovaných věd* (FAV) *Západočeské univerzity v Plzni*. Tento popis byl vytvořen v nástroji *Rational Method Composer*® (RMC) společnosti *IBM*®.Model procesu byl následně využit pro vytvoření šablony studentských projektů v rámci předmětu ASWI v nástroji pro jejich řízení *Rational Team Concert*® (RTC), který je rovněž produktem společnosti *IBM*. Výsledné produkty práce byly ověřeny garantem, současnými studenty a absolventy předmětu KIV/ASWI.

V textu práce je nejprve definován a popsán proces vývoje software obecně. Jsou vysvětleny jeho základní složky a principy a stručně popsány nejpoužívanější metodiky. Dále se text zabývá postupy sestavení procesu pro konkrétní účel z těchto metodik, jejich praktik a konceptů.Další část práce tvoří popis nástrojeRMC v takové míře detailu, v jaké byl v průběhu práce prozkoumán a používán. Následuje obecný popis nástrojů pro řízení projektů a jejich zástupců.

V další části textu je podrobně popsán postup tvorby modelu procesu v nástroji RMC a využití některých jeho základních funkcí, jakož i postup převodu vytvořeného modelu na šablonu projektu pro nástroj RTC.Dále je v textu práce podrobně popsán proces studentských projektů předmětu KIV/ASWI, stejně jako jeho vazby na dříve popsané metodiky. Následuje podrobnýpopis vytvořeného popisu tohoto procesu v nástroji RMC.

V poslední části práce jsou vzneseny návrhy pro údržby a rozšíření vytvořeného popisu procesu ASWI, a rovněž návrhy a možnosti využití znalostí a zkušeností nabytých při jeho implementaci.Poté jsou diskutovány přínosy a výsledky celé práce.

Text práce má využití jako návod pro práci s RMC a převod procesních šablon do RTC. Důvodem zahrnutí těchto návodů do práce je dosavadní absence podobného materiálu v českém jazyce a jisté úzké zaměření jednotlivých dostupných návodů pro RMC v jazyce anglickém. Ostatní výstupy práce slouží jako výukový materiál pro studenty předmětu KIV/ASWI a jejich snazší adaptaci na proces a zahájení projektů.

2. Proces vývoje software

Tato kapitola se zabývá procesem vývoje software v obecné teoretické rovině, jeho základními složkami, hlavními přístupy v jeho historickém vývoji, konkrétními příklady metodik a vzorů a přístupy k sestavení a konfiguraci procesu pro konkrétní problém reálného světa.

2.1 Definice procesu

Proces vývoje software jako obecný pojem představuje množinu praktik (specifický způsob vykonání činnosti nebo dosažení cíle), postupů (sekvencí konkrétních kroků a činností), konceptů a časových sledů událostí, podle nichž je vyvíjen softwarový produkt. Jedná se o jakousi šablonu, či směrnici obvykle uzpůsobenou softwarovou firmou nebo jinou institucí, která slouží jako rámcový návod pro projekty v oblasti softwarového vývoje.Projekty jsou konkrétními instancemi procesů s přesně určeným konkrétním finálním produktem, který je v jejich průběhu vyvíjen.

Nutnost vytvářet definovat, konfigurovat a vymezit procesvývoje software vznikla v podstatě zároveň se vznikem samotného softwarového inženýrství a rostla spolu s potřebou zapojení více než jednoho vývojáře a jiných aktérů do vývojového procesu. Definovaný proces umožňuje lepší automatizaci, adaptaci a opakovatelnost vývoje v těch krocích, které se mění málo nebo vůbec, v závislosti na velikosti konkrétního projektu a účelu finálního produktu.

*Pozn.:*Pojem proces obecně nereferuje samozřejmě jen k oboru vývoje software, ale objevuje se napříč všemi odvětvími organizované lidské činnosti. Nicméně v tomto textu je pod pojmem proces myšlen konkrétně proces vývoje software.

2.2 Základní principy

Než přejdeme ke konkrétním metodikám a modelům procesů, popišme nejprve základní aspekty tohoto konceptu obecně a nezávisle na konkrétní instanci. Základní obecné pojmy, respektive složky, v rámci procesu jsou**role,artefakty**,**aktivity**,**toky práce**,**disciplíny**,a **další procesní elementy**.

2.2.1 Role

Role je označení pro pozici, funkci nebo sadu odpovědností osoby vzhledem k procesu. Konkrétními rolemi mohou být například vývojář, tester, analytik, architekt, projektový manažer, vedoucí vývojového týmu, vývojový tým, apod.

Jak už předchozí výčet napovídá, jedna role neoznačuje vždy jednu konkrétní osobu. Naopak, role může představovat skupinu lidí (např. vývojový tým), může být stejná pro několik lidí

(obvykle existuje více než jeden tester, více než jeden vývojář, atd.), a stejně tak může jedna osoba zastávat několik rolí (zvláště u menších vývojových týmů může být např. architekt zároveň vývojářem, apod.). Role je tudíž pojem na určitém stupni abstrakce nezávislý na mapování těchto rolí přímo na konkrétní osoby.

2.2.2 Artefakty

Artefakt je pojem označující jakoukoli věcnou složku procesu. Jedná se obvykle o část informací, které jsou v průběhu procesu vytvářeny, měněny nebo jinak používány.

Artefaktymohou nabývat různých konkrétních forem od stavů celého procesu, ústních dohod, myšlených seznamů, nákresů na papíře či tabuli, přes modely nebo jejich části, formální i neformální dokumenty, výkonné části kódu a testů, výsledky testů, projektové či iterační plány, průběžné verze software, až po finální produkt.

2.2.3 Aktivity

Aktivita je kompaktní soubor úkolů, či kroků vedoucí k vytvoření výstupů ze vstupů. Jinak řečeno je to miniaturní proces, ve kterém osoba zastávající určitou roli vytváří, mění či využívá některé artefakty. Aktivita je tudíž kromě svého názvu, průběhu a kroků, definována rolí, která jí vykonává, a svými vstupními a výstupními artefakty.

U aktivit je zřejmě největším problémem jejich granularita z časového hlediska. Každý proces lze rozdělit na několik dílčích fází, ty dále na menší a menší, až na úroveň jednotlivých kroků, proveditelných časově v řádu hodin, ne-li minut. Ideální jemností (či naopak hrubostí) rozdělení procesu na aktivity je zhruba ta míra, kdy je aktivita vykonávaná jednou rolí[1], vykonatelná v řádu jednotek, maximálně několika málo desítek hodin, má omezenou, maloua jasně identifikovatelnou sadu vstupů a výstupů, a je souborem kroků úzce tematicky spojených natolik, že jejich oddělením by vznikly dvěa více aktivit se stejnou množinou vstupů i výstupů a stejnou vykonavatelskou rolí.

Klasickými příklady aktivit mohou být „*Vyjednat požadavky*“, „*Provést test*“ nebo „*Napsat uživatelskou*“ dokumentaci.

2.2.4 Toky práce

Tokem práce (anglicky *workflow*) je v kontextu procesu myšlena časová návaznost mezi jeho jednotlivými fázemi, aktivitami, iteracemi, úkoly a kroky.Tok práce může být popsán jak strukturovaným textem, tak například diagramem (viz *2.3*).

[1] Přítomnost pouze jedné role v aktivitě není pravidlem. Další role mohou být volitelné, tj. mohou a nemusí se na ní podílet, ale mohou existovat i aktivity, u nichž je přítomnost dvou rolí nezbytná. Příkladem je vyjednávání požadavků na vyvíjený produkt, kde musí být přítomen jak zástupce vývojového týmu (nejčastěji analytik), tak zástupce (zástupci) zákazníka.

Zatímco některé aktivity (a jiné jednotky, na něž se proces dělí) jsou zcela nezávislé, jiné vyžadují, aby jim některé další předcházely, nebo je následovaly. Tyto závislosti (v některých případech i se vztahy mezi aktivitami, rolemi, artefakty a událostmi) vyjadřuje právě tok práce. Ten může mít různý záběr v rámci procesu. Například tok prácevyznačující návaznost fází zjevně popisuje celý projekt, i když ne do hloubky. Tok prácemůže ale vyjadřovat i sled aktivit v rámci fáze nebo iterace.

2.2.5 Disciplíny

Disciplíny jsou vlastně určité kategorie či obory činnosti se společným zaměřením a cílem, do kterých jsou rozřazeny jednotlivé aktivity, potažmo i artefakty. U rolí je situace o něco složitější. Je pravda, že většina rolí hraje hlavní úlohu v jedné nebo více disciplínách, obvykle se ale také výrazně podílí na některých dalších.

Příklady disciplín jsou „*Implementace*“, „*Testování*“, „*Řízení projektu*“, „*Analýza a design*“, „*Řízení verzí a změn*“, atd.

2.2.6 Další procesní elementy

V procesu mohou hrát roli i prvky nezařaditelné do jedné z prvních čtyř kategorií. Může se jednat o nejrůznější šablony a příklady dokumentů, návody na použití nástrojů, směrnice, dílčí postupy, užívané koncepty a praktiky atd.

Příslušnost těchto elementů k některé z disciplín není vždy jasná, tzn.: některé jsou snadno zařaditelné (např. plán projektu jasně spadá do disciplíny řízení projektu), a některé mohou hrát roli ve více, ne-li všech disciplínách v závislosti na tom, jaké disciplíny má daný konkrétní proces definovány.

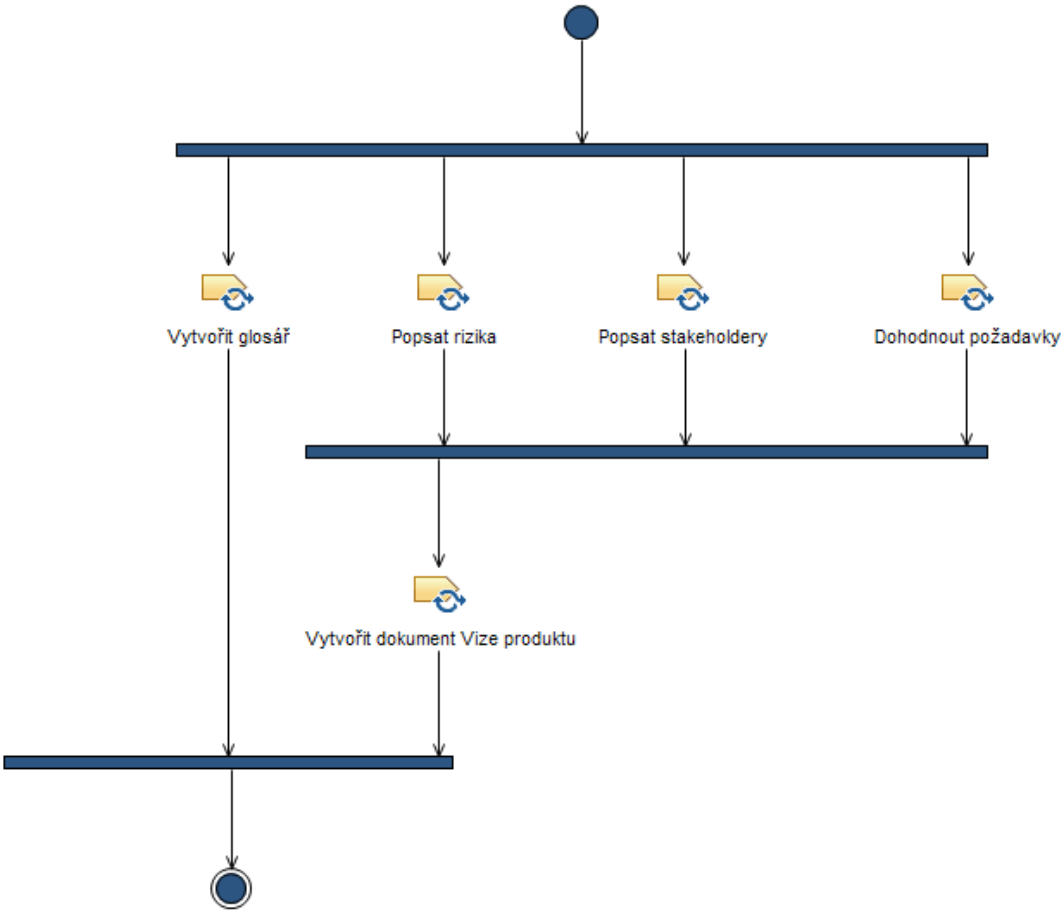
2.3 Základní možnosti modelování procesů

Nyní se podívejme na nejzákladnější možnosti modelování procesů. Hlavním účelem modelu procesu je zachytit časové závislosti jednotlivých aktivit v průběhu projektu a vztahy mezi aktivitami, artefakty a rolemi.

2.3.1 Diagram aktivity

Diagram aktivity (workflow diagram, či diagram toku práce) je obrazová reprezentace návazností mezi dílčími dynamickými částmi procesu, jinak řečeno tok práce (viz *2.1.4*). Proto stejně jako tok práce (viz *2.2.4*) může zachycovat sled aktivit v rámci celého procesu, fáze nebo jiné určité části celého průběhu procesu.

Diagram aktivit může mít několik přesně definovaných podob, jako jsou například UML diagram aktivit nebo BPMN.



Obrázek 2.1– Příklad diagramu aktivity

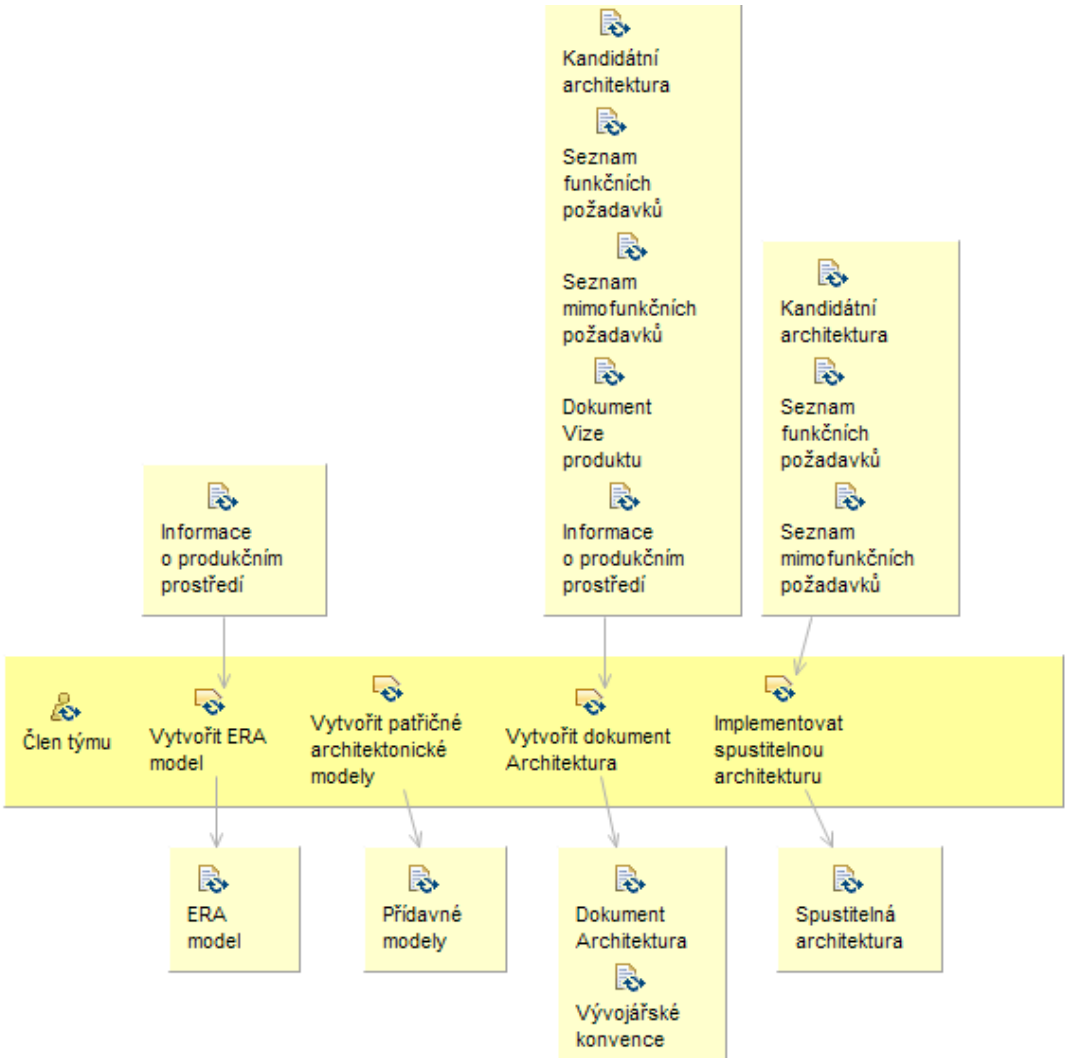
Příklad diagramu aktivity je na obrázku *2.1*. Základními prvky tohoto konkrétního diagramu jsou, kromě jednotlivých úkolů a šipek znázorňujících návaznosti, hradla pro určení závislosti více úkolů na dokončení jednoho nebo naopak a počáteční a koncový uzel celé aktivity. Dalšími možnými prvky v diagramu (nejsou na obrázku) jsou například rozhodovací uzly a následně možné cykly nebo textové popisy jednotlivých závislostí.

2.3.2 Diagram detailu aktivity

Druhý základním typem modelu je model vyznačující vztah mezi aktivitami, rolemi a artefakty, nazývaný diagram detailu aktivity.

Diagram detailu aktivityv základu graficky zobrazuje jednu aktivitu, jednu roli v rámci této aktivity činnou, všechny úkoly této role v rámci dané aktivity a vstupní a výstupní artefakty každého úkoly.

Obdobným způsobem může diagram zobrazovat aktivitu jako celek (bez rozdělení na úkoly), nebo kompletní model aktivity, tzn. soubor výše popsaných diagramů, jeden pro každou roli v aktivitě činnou.



Obrázek 2.2– Příklad diagramu detailu aktivity

Příklad diagramu detailu aktivity je znázorněn na obrázku *2.2*.

2.4 Metodiky vývoje a procesní vzory

Jako samotné softwarové inženýrství, i konkrétní metodiky a vzory procesů mají svůj historický vývoj. V tomto oddíle jsou popsány nejdůležitější etapy tohoto vývoje a nejvýznačnější a dosud nejpoužívanější zástupci metodik a vzorů.

Následující citát z knihy Scotta Amblera a Matthew Holitzi zdůrazňuje, že žádná z definovaných metodik není užívána bez úprav pro konkrétní účel a každá má své silné a slabé stránky.

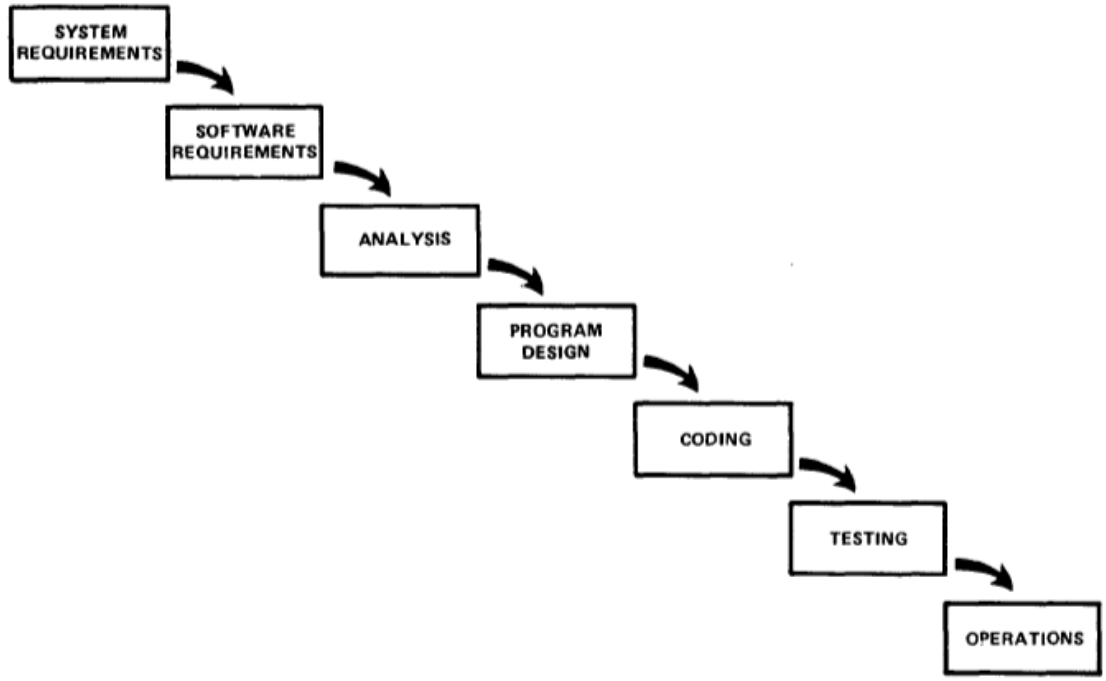
„As software development has evolved over the last 70-plus years, it has had several dominant models or methodologies. Each had reasons for coming into being, and really no model is used as is; models are almost always tailored to suite their unique needs. Each model has its benefits and drawbacks.“[1]

2.4.1 Sekvenční metodiky

Prvním uceleným přístupem k procesům byly logicky nejjednodušší sekvenční metodiky, které vznikaly od 70. let 20. století.

Ty dělí proces do fází, z nichž v každé de facto probíhají aktivity pouze jedné určité disciplíny. Konkrétně nejprve probíhají analýzy a modelování business kontextu vyvíjeného software, po jejich ukončení sestavování požadavků na finální produkt, poté analýzy a design řešení, následně implementace, pak testování a poté konečné kroky a vydání, resp. předání samotného software.

Tento přístup je dodnes v hojné míře aplikován i některými z předních softwarových firem na trhu a to i přes jeho zjevné nedostatky. Jedním z hlavních je zejména malá míra zpětné vazby od zákazníka v průběhu vývoje. Zákazník sice může souhlasit s analýzami a designem řešení, ale jedná-li se o laika v oboru IT, nedokáže si představit jejich aplikaci a celkové řešení v praxi, což vede k jeho časté nespokojenosti s finálním produktem. Nejen tento faktor pak může vést k nutnosti vrátit se do předchozích fází vývoje, což je ale díky konstrukci metodiky velmi obtížné, až nemožné,nebo alespoň finančně i časově náročné.



Obrázek 2.3 – Schéma vodopádového modelu vývoje software[2]

Vodopádový model

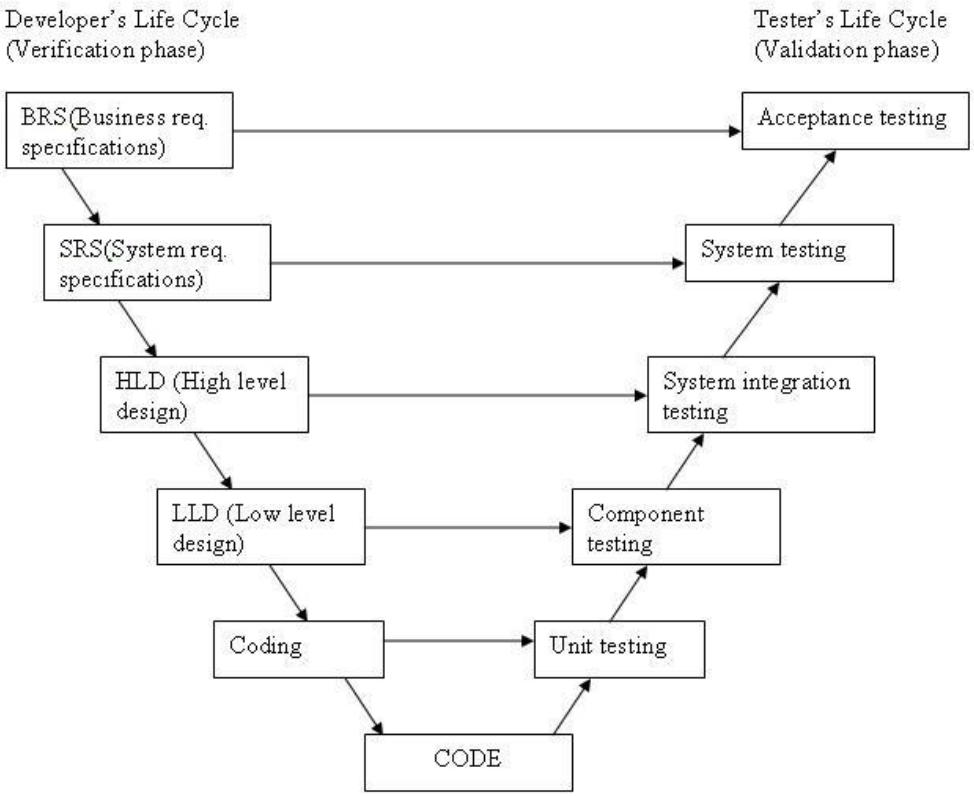
Nejznámějším zástupcem sekvenčních metodik je vodopádový (*waterfall*) model, který v roce 1970 poprvé formálně definoval Winston W. Royce.

[2] Obrázek přejat z [22].

Tento model s sebou nese všechny problémy skupiny sekvenčních vzorů procesů. Postup vývoje v klasickém vodopádovém modelu ilustruje obrázek *2.3*.

Vodopádový model v dalších dekádách doznal určitá vylepšení a změny jako zpětné cesty mezi fázemi (stačí si na obrázku *2.3* představit oboustranné šipky) nebo reakci na změny požadavků během provozu, která může vyvolat opakování procesu od první fáze, jíž se týká, místo iterace celého procesu.

Přesto, ač je proces dobře definovaný a snadno naučitelný a aplikovatelný, zůstává *vodopádový model* značně nepružným a zkostnatělým, což je zejména v dnešní době stále se zvyšujících frekvence změn značnou nevýhodou.



Obrázek 2.4– Schéma V-modelu vývoje software[3]

V-model

V-model je zástupce sekvenčních metodik, jež se snažil do jisté míry zmenšit nedostatky vodopádového modelu a rizika z nich plynoucí.

Jedná se v podstatě o pouhé rozšíření původního vodopádového modelu o řadu fází různých typů testování prováděných po samotné implementaci software k postupnému ověření fází

[3] Z obrázku je jasně patrný původ názvu V-model. Obrázek přejat z [26].

předchozích, obrazně řečeno, od zdola nahoru (nejdříve se ověřuje fáze těsně před implementací, až nakonec fáze první v celém procesu).

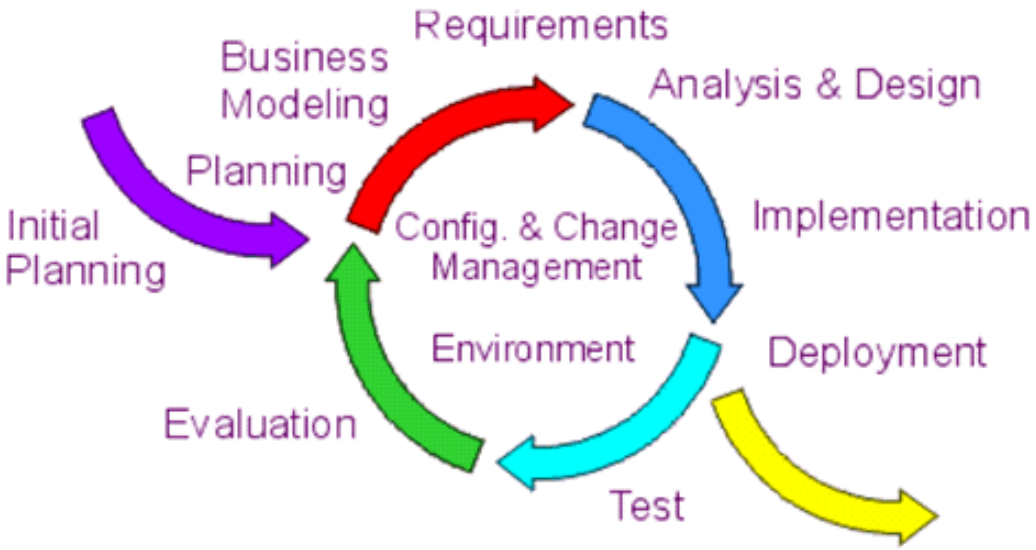
Základní princip *V-modelu* znázorňuje obrázek *2.4*.

2.4.2 Přírůstkové a iterativní metodiky

První silnou odezvou na nedostatky sekvenčních metodik byl příchod přírůstkových (neboli inkrementálních) a iterativních metodik v 80. – 90. letech minulého století.

Hlavní předností těchto metodik bylo jejich tzv. řízení rizika (*risk driven development*), který se snažil o eliminaci, nebo alespoň minimalizaci, všech zásadních rizik projektů již v raných fázích jejich průběhu.

Toho efektu dosahovaly tyto metodiky, jak už název napovídá, rozdělením vývoje na iterace, v jejichž průběhu vždy projekt o něco postoupí napříč všemi (nebo skoro všemi[4]) disciplínami, což vyvolá nárůst celkové aktuální hodnoty vyvíjeného produktu – přírůstekneboli inkrement. Po každé z těchto iterací jsou výsledky prezentovány zástupcům strany zákazníka a sbírány jejich připomínky, reakce a nároky na změny.



Obrázek 2.5 – Schéma iterace[5]

Průnik většího počtu disciplín a oblastí činnosti v rámci jednotlivých iterací nutně zvýšil i potřebu sestavování plánů. Plány mohou být iterační, ale mohou pokrývat i průběh celého projektu a existovaly samozřejmě i před příchodem iterativních metodik. Nicméně z výše

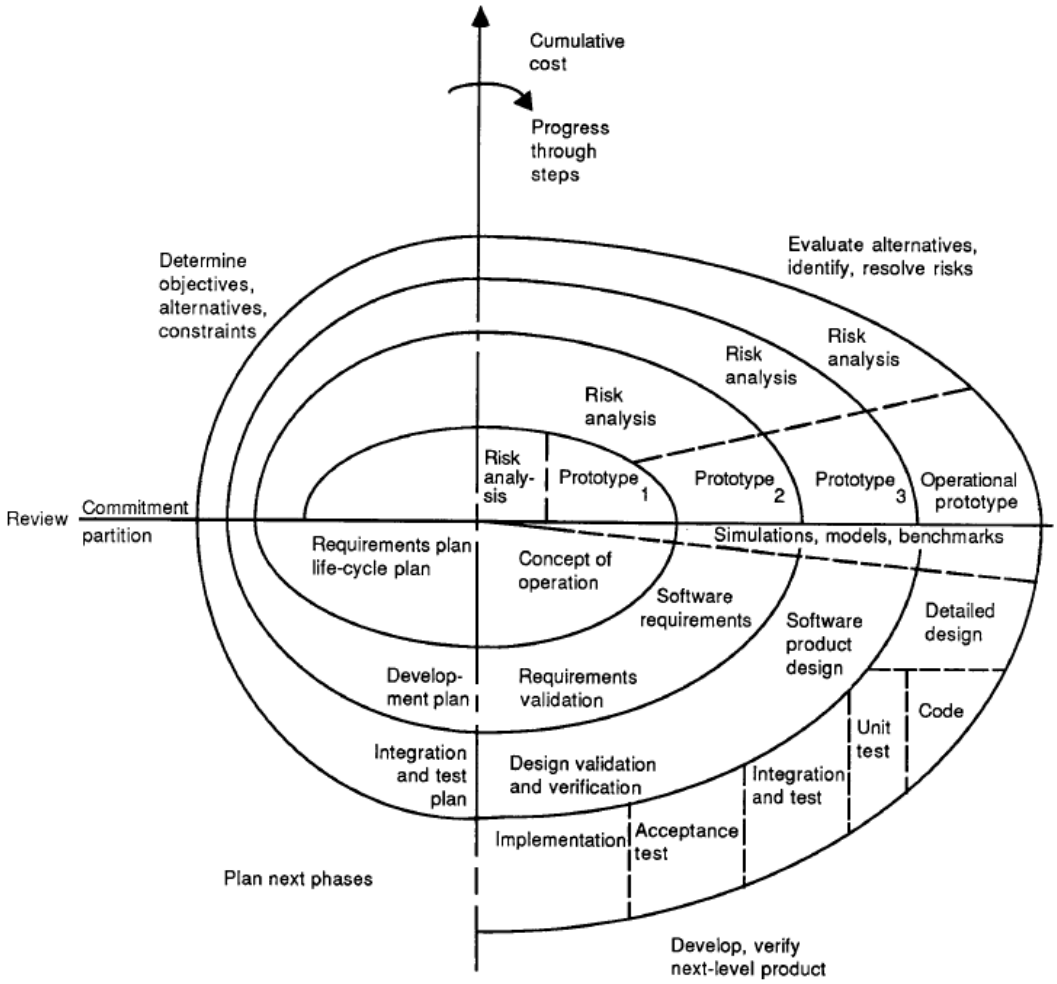
[4] Je jasné, že v prvotních iteracích projektů nebude probíhat např. implementace, stejně jako v posledních nebude zastoupeno modelování business logiky a požadavků.
[5] Obrázek převzat z [24].

popsaného důvodu je jejich přítomnost v projektech s iterativním vývojem mnohem významnější.

Základní princip iterací představuje diagram iterace na obrázku *2.5*.

Spirálový model

Spirálový model z 80. let 20. století může být považován za prvního zástupce iterativních metodik. Více informací viz [2].



Obrázek 2.6 – Schéma spirálového modelu vývoje software[6]

Podle něj vývoj v každé otáčce (resp. iteraci) má čím dál větší záběr, tzn., že analyzuje, modeluje, implementuje a testuje větší a větší část konečné funkcionality a podoby finálního produktu. Po každé takové otáčce proces produkuje prototyp s omezenou funkčností, který je ale vhodný k testování a verifikaci zákazníkem.Obrázek *2.6*ilustruje proces vývoje software podle *spirálového modelu*.

[6] Obrázek převzat z [2].

Unified Process

Unified Process[7] (*UP*) je klasickým a často používaným zástupcem iterativních metodik, který vznikl na konci 90. let 20. století.

Spíše než o do detailu definovaný proces se jedná o vymezenou formu či framework složený z několika zásadních praktik, který společnosti a organizace činné v oboru softwarového vývoje přebírají, upravují a detailněji specifikují pro své konkrétní účely (více o tomto přístupu přizpůsobování procesu v oddíle*2.5.1*). Tomu napomáhá i fakt, že *UP* a všechny procesy na něm založené jsou dobře škálovatelné a mohou podle nich probíhat malé, střední i rozsáhlé projekty.

Celý proces v *UP* je rozdělen na 4 fáze:**Inception** (Zahájení),**Elaboration** (Rozpracování),**Construction** (Konstrukce)a **Transition** (Předání),z nichž každá je ukončena tzv. milníkem[8] (bližší informace k fázím a milníkům viz následující oddíl*IBM® Rational Unified Process®*).

Kromě fází jsou zásadními praktikami v rámci *UP*důsledný iterativní a přírůstkový vývoj, důraz na soustavné hledání a odstraňování rizik projektů, proces řízený požadavky (resp. funkčností; *use case driven development*) a výsadní postavení architektury.

IBM® Rational Unified Process®

Rational Unified Process (*RUP*) je jedním z produktůřady*Rational®* společnosti *IBM* a zároveň první konkrétní metodikou odvozenou od UP (viz předchozí sekce *Unified Process*Unified Proces). Na rozdíl od ní však podléhá obchodní značce a je o něco specifičtěji definovaný.

Přesto se jedná o nejznámějšího a zřejmě nejpoužívanějšího zástupce derivátů *UP* frameworku, a proto jsou pojmy UPa *RUP* v praxi často zaměňovány. Podle zdroje [3] může být *RUP* definován třemi způsoby:*RUP* jako přístup k vývoji software[9],*RUP* jako dobře definovaný a strukturovaný proces softwarového inženýrství a *RUP* jako produkt poskytující procesní framework[10] přizpůsobitelnývlastním potřebám.

Hranice mezi první a druhou definicí je nejasná, ale v zásadě se v této sekci budeme zabývat především*RUP* jako definovaným modelem procesu.

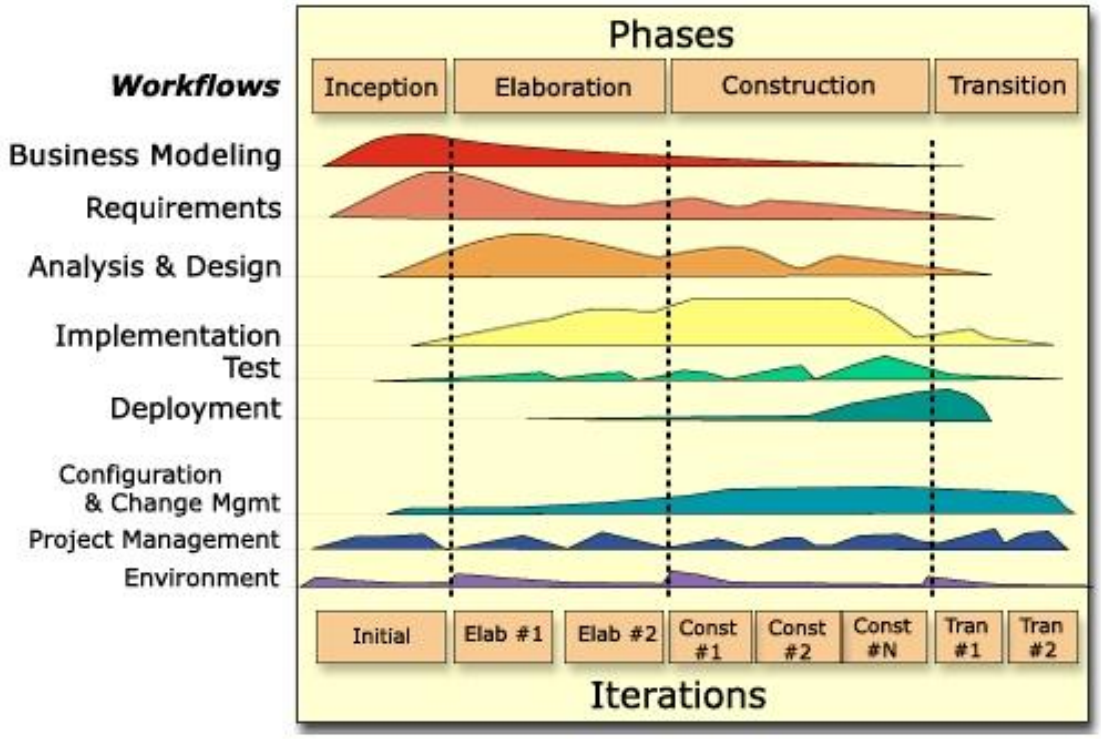
---

[7] Nazýván též Unified Software Development Process.

[8] Milník si lze představit jako cílovou pásku jedné konkrétní fáze. Není přesně definována jeho pozice v čase ani počet iterací, který k němu vede. Každý milník má pouze dánu množinu cílů a popis stavu projektu, kdy je možné považovat jej za dosažený.

[9] *RUP* v tomto smyslu představuje pouze základní myšlenky iterativního a riziky řízeného vývoje a přidává k nim několik dalších víceméně filozofických praktik.

[10] Součástí tohoto frameworku je i nástroj *Rational Method Composer*, jímž se hlouběji zabývají kapitoly 3 a 5.



Obrázek 2.7– Schéma vývoje software podle RUP[11]

Proces v rámci *RUP* metodiky je vymezen především dvěma aspekty, či osami, jak je vidět na obrázku *2.7*. Horizontální osa je časová a představuje rozdělení procesu na čtyři fáze podle UP frameworku. Vertikální osa pak symbolizuje jednotlivé disciplíny v *RUP* definované. Na obrázku *2.7* jsou to od shora dolů:

- **modelování businessu**,
- **požadavky**,
- **analýza a design**,
- **implementace**,
- **testování**,
- **nasazení / vydání**,
- **řízení verzí a změn**,
- **řízení projektu**
- a **prostředí**.

Celé schéma na obrázku *2.7* potom znázorňuje poměrné zastoupení aktivit v rámci jednotlivých disciplín v průběhu fází. Fáze jsou dále děleny na iterace.Každá z fází má minimálně jeden hlavní výstupní artefakt a každá je také ukončena dosažením příslušného milníku.

[11] Obrázek převzat z [25].

Tabulka 2.1 – Fáze RUP s jejich cíly, milníky a hlavními artefakty

Fáze	Hlavní cíle	Milník	Hlavní artefakt
Inception	Zjistit obsah a rozsah projektu, určit požadavky, identifikovat rizika, rozhodnout, zda projekt realizovat, či zrušit	LifeCycle Objectives (LCO)	Vize produktu (dokument)
Elaboration	Navrhnout a ověřit architekturu, stabilizovat požadavky, implementovat a otestovat kostru architektury	LifeCycle Architecture (LCA)	Architektura (dokument nebo jiná forma popisu)
Construction	Implementovat většinu funkčnosti a průběžně testovat	Initial Operatinal Capability (IOC)	Beta-verze (90 % funkčnosti produktu)
Transition	Dokončit implementaci, testování, dokumentaci a provést nasazení / předání produktu do ostrého provozu	Product Release (PR)	Finální produkt (release)

Fáze, jejich hlavní cíle, milníky a nejpodstatnější artefakty ve stručnosti ilustruje tabulka *2.1*.

Vývojový tým v procesech metodiky *RUP* obsahuje velmi dobře definované role jako analytik, architekt, vývojář, tester, projektový manažer, atd., které mohou být díky aspektu škálovatelnosti procesů mapovány i na velmi malé týmy, nebo naopak více specifikovány (jako byznys analytik, databázový architekt, apod.).

Metodika *RUP* klade zvýšený důraz na brzké a soustavné soustředění vývojového týmu na:

- hlavní rizika projektu,
- tvorbu především výkonného kódu,
- reakci na příchozí změny od začátku projektu,
- brzké vystavění základu spustitelné architektury,
- znovupoužitelnost komponent a (pokud možno) sestavování software z nich,
- důslednou kooperaci týmu a jeho konzistentní práci,
- zajištění přidané hodnoty pro zákazníka
- a zajištění kvality produktu už v průběhu vývoje, ne až na jeho konci.

Pozn.: Detailní popis metodiky *RUP* není předmětem této práce a více informací o něm naleznete v [3].

Open Unified Process

Dalším zástupcem derivátů modelu UP je otevřená (*open source*) metodika*Open Unified Process* (*OpenUP*). Stručný popis metodiky poskytuje následující citace.

„OpenUP is a lean Unified Process that applies iterative and incremental approaches within a structured lifecycle. OpenUP embraces a pragmatic, agile philosophy that focuses on the collaborative nature of software development. It is a tools-agnostic, low-ceremony process that can be extended to address a broad variety of project types.“[4]

Jedná se o model, jež do standardních praktik UP vnáší prvky agilního vývoje (viz oddíl*2.4.3*), jakými jsou menší důraz na nástroje a formalitu artefaktů, politiku tzv. „mikropřírůstků“ (malých nárůstů funkčnosti dodávaných zákazníkovi velmi často, tzn. po každé iteraci) a vývojové týmy samostatně si určující svojí strukturu.

Díky své charakteristice otevřenosti prochází *OpenUP* neustále vývojem, na kterém se podílí široká komunita odborné společnosti.

Pozn.:Popis *OpenUP* metodiky je k dispozici na webových stránkách [4] a je vytvořen v nástroji pro modelování a popis procesů *Eclipse Process Framework (EPF) Composer*, který je otevřenou (opensource) obdobou nástroje *IBM Rational Method Composer* (viz kapitola 3).

Enterprise Unified Process

Jak říká, následující citát, metodiky RUP je sice dobrým, přesto pouhým začátkem, který nepopisuje části procesu po předání, resp. nasazení produktu projektů.

„There is more to IT than system development – the RUP is a good start, but you need more. The EUP extends the RUP to cover the entire system lifecycle, including both production and system retirement. The EUP also extends the RUP to cover cross-system issues. It was introduced in 1999 and has evolved since them; it will continue to evolve over time.“[5]

Enterprise Unified Process (*EUP*) se soustředí na rozsáhlé enterprise systémy složené z několika různých aplikací (např.: kombinace databáze zaměstnanců, redakčního systém, účetní aplikace, atd.) obvykle užívané ve funkčním provozu v dlouhodobých časových horizontech.

EUP je vlastně rozšířením metodiky RUP (viz sekce *IBM® Rational Unified Process®* výše v tomto oddílu), ke které(jak je vidět na obrázku v příloze *B*[12]) přidává dvě nové fáze

[12] Zmenšená forma obrázku vložená zde by byla kvůli svým rozměrům nečitelná.

Production (Provoz) a **Retirement** (Vyřazení) a celkem 8 dalších disciplín. Na obrázku v příloze B jsou to od shora dolů:

- **provoz a podpora**,
- **podnikové modelování businessu**,
- **řízení portfolia**,
- **podniková architektura**,
- **strategie znovupoužití**,
- **řízení lidských zdrojů**,
- **podniková administrativa**
- a **vylepšování softwarového procesu**.

Unified Process for Education

Posledním zde uváděným příkladem derivátu UP (viz sekce Unified Proces) je *Unified Process for Education* (*UPEDU*) vyvinutý na montrealské polytechnice pro účely výuky procesů vývoje software. *UPEDU* je stejně jako model procesu předmětu KIV/ASWI, který je hlavním předmětem praktické části této práce (podrobně popsán v kapitole 6), do jisté míry formou RUP (viz sekce *IBM® Rational Unified Process®* výše v tomto oddílu) upravenou pro potřeby studentských projektů tak, aby bylo pro ně setkání s iterativními metodikami a získávání zkušeností s tímto modelem vývoje software co nejschůdnější. Stručně jej charakterizuje následující citát.

„The Unified Process for EDUcation, or UPEDU, is a web-enabled set of software engineering best practices that provide you with guidance to streamline your team's development activities. UPEDU has been customized from the RUP - an industry-wide process platform - for the educational environment.The Unified Process for EDUcation unifies the software development team and enhances team communication. Using online browser navigation, each team member has instant access to UPEDU's knowledge base and process guidelines from their desktop.“[6]

Kompletní popis procesu *UPEDU*je přístupný na webových stránkách, viz [6].

2.4.3 Agilní metodiky

Základem pro historicky nejnovější skupinu vzorů procesů, agilní metodiky, je **Manifest pro agilní vývoj software**, sepsaný v roce 2001 skupinou vývojářů snažících se posunout vývoj tzv. „lehkých“ (*lightweight*) metodik vývoje.

Tento manifest je krátký text v originálním znění (v angličtině) čítající 68 slov a jeho esencí jsou 4 hlavní pravidla, která vyjadřují priority agilního vývoje. Tyto pravidla stanovují, že přednost mají:

- **lidé a** jejich **interakcepřed** definovanými **procesy a nástroji**,
- **funkční software před**komplexní**dokumentací**,
- **spolupráce** se zákazníkem **před** vyjednanými **smluvními podmínkami**,
- **reakcena změnupřed** striktním **dodržováním plánu**.

Agilní metodiky se soustředí na to, co vidí jako faktory nejvíce ovlivňující úspěch projektů vývoje software. To zahrnuje

- komunikaci jak se zákazníkem, tak v rámci vývojových týmů – ta minimalizuje možnost nedorozumění a špatného pochopení požadavků,
- dodání především funkčního produktu – ten má totiž pro zákazníka logicky větší hodnotu než podpůrné dokumenty a další formální artefakty,
- a především (jak už název napovídá) snadnou a rychlou reakci na změny požadavků zákazníka a tím úpravukurzu projektu (mnohdy drastickou) v jeho průběhu – a to i za cenu porušení plánu nebo procesních praktik.

Největší inovací, kterou agilní přístup oproti historicky předešlým metodikám zavádí, je právě úprava samotného procesu, jeho postupů a praktik v samotném průběhu jednotlivých projektů na základě sebereflexe vývojového týmu, jak říká i následující citát.

„The developers who created agile understood the importance of creating a model in which each iteration in the development cycle „learned" from the previous iteration. The result was metodology that was more flexible, efficient, and team-oriented than any of the previous models."[1]

Účelem takovýchto úprav je, aby se proces stal co nejefektivnějším již v průběhu aktuálního projektu, místo aby čekal na vyhodnocení sebraných zkušeností na jeho konci a úpravy se tak promítly až do projektů budoucích.

Agilní vývoj se v posledních deseti letech stal fenoménem a populárním přístupem, jež přebírá stále větší počet hlavně malých a středních společností. Tomu napomáhá i další zásadní odlišnost od předchozích přístupů k vývoji, kterou je jistá míra otevřenosti a vůle institucí užívajících agilní metodiky vyměňovat si mezi sebou zkušenosti, přínosy a problémy při jejich zavádění.

Tento fakt potvrzuje například vznik *Agilní asociace*, což je volné uskupení vývojářů a dalších příznivců agilních metodik s dceřinou asociací i v České republice snažící se šířit povědomí a zkušenosti v oboru užívání agilních metodik. Pro potřeby sdílení zkušeností pořádá asociace ročně konferenci a během roku i několik lokálních setkání (tzv. „*Open Café*") v Praze, Plzni a dalších městech.

„Cílem Agilní Asociace je zvýšit povědomí o Agilních metodách řízení, a vytvořit platformu pro sdílení informací a zkušeností z oblasti Agilních metod.“[7]

Scrum

Metodika *Scrum* je zřejmě nejznámějším zástupcem agilních přístupů k vývoji software představeným v roce 1995 Kenem Schwaberem a Jeffem Sutherlandem na konferenci *Object-Oriented Programming, Systems, Languages & Applications* v Austinu ve státě Texas ve Spojených státech amerických. Z toho je vidět, že agilní metodiky existovaly dříve, než byly oficiálně sdruženy a pojmenovány (viz úvodní sekce tohoto oddílu *Agilní metodiky*).

Scrum je pro mnohé rozporuplnou a těžko pochopitelnou metodikou hned z několika důvodů. Jedním z nich je fakt, že ač má *Scrum* podivuhodně málo striktně daných pravidel a praktik, přesto je dobře a přesně definován. Přechod na *Scrum* z robustnějších a detailněji popsaných metodik, jakou je například RUP (viz *2.4.2*), představuje více než naučení se nových postupů a struktur. Jedná se spíše o změnu myšlení, filozofie a samotného přístupu k procesu vývoje software. Esenci pro mnohé matoucího charakteru *Scrumu* dobře ilustruje Ken Schwaber v následujícím citátu.

„On one hand, Scrum is disarmingly simple. The process, its practices, its artifacts, and its rules are few, straightforward, and easy to learn. … On the other hand, Scrum's simplicity can be deceptive. Scrum is not a prescriptive process; it doesn't describe what to do in every circumstance.“[8]

Jak již bylo výše zmíněno, *Scrum* definuje velmi málo velice jednoduchých konceptů a praktik, kromě těch, které jsou společné pro všechny agilní metodiky (viz úvodní sekce tohoto oddílu *Agilní metodiky*). V následujícím seznamu jsou vyjmenovány a stručně popsány stěžejní principy Scrumu.

- **Sprint** – *Sprint* je obdoba *Scrumu* pro iteraci. Obvykle se jedná o 30 denní vývojovou fázi mezi dvěma schůzkami se zákazníkem, na kterých jsou předváděny výsledky dosavadního průběhu projektu (tzv. ***Sprint demo***) a domlouván postup pro nastávající *sprint* (tzv. ***Planning meeting***). Nicméně délka 30 dnů není pravidlem a jako celý proces *Scrumu* je škálovatelná podle potřeb konkrétního projektu.
- **Backlog** – Pod pojmem *backlog* se skrývá *Scrumu* vlastní forma plánu. Obvykle se zavádí jak backlog projektu, tak jednotlivých *sprintů*. Je to vlastně seznam aktivit a úkolů určených pro etapu projektu, kterou backlog pokrývá, opatřených prioritami, stupni závažnosti, odhady náročnosti, současným stavem jejich zpracování a dalšími atributy. Podoba se různí od souboru poznámek přes seznam na tabuli, papírový

dokument, až po prostor projektu určený ve specializované aplikaci (např. viz kapitola 4).

- **Role v projektu**
  - **Vývojový tým** – V projektech *Scrumu* je vývojový tým samoorganizační, což znamená, že si jeho členové sami mezi sebou určují specializované role (analytik, vývojář, tester, atd.) a odpovědnosti, aniž by jim do tohoto procesu kdokoli zasahoval z vnějšku. Informace o vnitřní struktuře týmu je pro vyšší instance (např. vedení společnosti, zákazník, atd.) nepotřebná a neznámá. Proto zodpovědnost za vyvíjený produkt nese tým jako celek.
  - **Vlastník produktu** – Název role je poněkud zavádějící. Jedná se vlastně o zástupce tzv. *stakeholderů* (osob, jimž má výsledný produkt přinést přidanou hodnotu; např. vedení společnosti nebo zákazník), potažmo osobu fungující jako spojení mezi stakeholdery a vývojovým týmem. Zajímá se především o přínos projektu (resp. *sprintu*) pro potřeby a obchodní záměry zákazníka a předává týmu požadavky na změny za účelem maximálního zvýšení tohoto přínosu vzhledem k aktuálnímu stavu projektu.
  - **Scrum Master** – *Scrum Master* je role, která v doposud popisovaných metodikách nemá paralelu. Jedná se v podstatě o jakéhosi mentora, či dozorčí osobu, která je přítomna schůzkám i samotné práci týmu. Jeho hlavní úlohou není dohlížet na věcnou stránku projektu, nýbrž sledovat, mentorovat a vynucovat pochopení principů samotného *Scrumu* a dodržování pravidel jeho procesu. Jinými slovy jeho snahou je, aby ostatní osoby v projektu činné dodržovaly praktiky a postupy metodiky *Scrum*, popř. procesu na ní postaveném.
- **Daily Scrum** – Pojem *Daily Scrum* označuje obvykle krátké každodenní synchronizační setkání vývojového týmu a *Scrum Mastera*, prováděné nejčastěji na začátku každého pracovního dne. Cílem schůzky je probrat odvedenou činnost z předešlého dne a její výsledky, objevené problémy a na jejich základě domluvit postup prací dne následujícího.
- **Iterační retrospektiva** – Retrospektiva je setkání vývojového týmu a *Scrum Mastera* na konci každé iterace za účelem sebereflexe a sebekritického zhodnocení průběhu uplynulé fáze vývoje z formálního pohledu. Tým by si měl vyměnit názory na to, kde byli ve sledování plánu, postupu, procesu a aplikaci praktik jeho silné a slabé stránky, a případně upravit proces tak, aby tým maximalizoval efektivitu svojí práce v nadcházející iteraci.
- **Timeboxing** – *Timeboxing* není čistě záležitostí *Scrumu*, dokonce ani agilních metodik, ale nejčastěji je tato praktika používána právě v jejich procesech. Pojem označuje

striktní časové vymezení nějaké aktivity (nejčastěji schůzek). Účelem je docílit toho, aby se zúčastněné osoby soustředily díky časovému tlaku pouze na nejdůležitější témata, úkoly a kroky a nezabíhalydo zbytečných detailů nebo se nezabývali aktuálně nepodstatnými otázkami. Po uplynutí předem stanoveného časového kvanta činnost končí bez ohledu na to, v jakém stavu se diskuze či úloha v daném momentě nachází.

Disciplined Agile Delivery

Druhým v tomto textu zmíněným zástupcem agilních metodik je *Disciplined Agile Delivery* (*DAD*).Jak citace níže říká, *DAD* je hybridní agilní přístup s orientací na učení se agilnímu přístupu vývoje a důrazem na lidi v procesu. Životní cyklus projektů podle *DAD* ilustrují obrázkyv příloze B[12].

"The Disciplined Agile Delivery (DAD) decision process framework is a people-first, learning-oriented hybrid agile approach to IT solution delivery. It has a risk-value delivery lifecycle, is goal-driven, is enterprise aware, and is scalable."[9]

Na nich je zřetelné rozdělení životního cyklu na fáze **Inception**, **Construction** a **Transition**. Oproti UPmetodikám (viz *2.4.2*) tudíž *DAD* nemá fázi **Elaboration**, jejíž aktivity z velké části přecházejí do fáze **Construction**. Rovněž je na obrázcíchv příloze Bpatrný agilní charakter metodiky, např. na použití každodenních koordinačních schůzek, retrospektiv a okamžitého zařazování nových požadavků do seznamu pracovních položek (*backlogu*).

DAD je hybridním procesem a kromě základu fází z UP využívá i praktiky jiných metodik. Jejich příklady a některé z jejich praktik, které DAD přebírá, jsou následující:

- **Scrum** (viz předchozí oddíl*Scrum*) – realizace pracovních položek podle priorit, vlastník produktu zodpovědný za reprezentaci zákazníků a jejich zájmů, produkce potenciálně použitelného řešení v každé iteraci, atd.,
- **Extrémní programování** (***Extreme Programming***) – průběžná integrace, refactoring, vývoj řízený testy, kolektivní vlastnictví, atd.,
- **Agilní modelování** (***Agile Modeling***) – praktiky dokumentace po zkompletování vize požadavků, vize architektury, modelování iterace, průběžná dokumentace, produkce modelů až v momentě potřeby (*just-in-time*), atd.,
- **Unified Process** (viz *2.4.2*) –„lehké" (*lightweight*) milníky, explicitní fáze, důraz na potvrzení architektury v prvních iteracích, redukování rizik všech typů v raných fázích životního cyklu, atd.,
- **Agilní data** (***Agile Data***) – refactoring databáze, databázové testy, agilní modelování dat, agilní enterprise strategie, atd.
- a **Kanban –** omezování souběžně vykonávané činnosti, vizualizace práce, atd.

DAD je příkladem fúze praktik několika metodik a následného vytvoření nového procesu, o které se dále píše v oddíle *2.5.2*.

2.5 Konfigurace procesu pro konkrétní účel

Žádná z popsaných, ani jiných existujících metodik nepředstavuje univerzální řešení pro proces vývoje software bez nutnosti úpravy pro každý jednotlivý konkrétní účel, či dokonce přímo projekt[13]. Proto každá společnost či jiná entita zabývající se (opakovaným) vývojem software musí nejen některou metodiku po zvážení možností převzít, ale před její samotnou aplikací rovněž přizpůsobit svým aktuálním a konkrétním potřebám. Obecně k tomuto cíli vedou dva postupy, kterým se věnují následující oddíly.

2.5.1 Převzetí a úprava definované metodiky

První možností, jak vytvořit svůj vlastní proces (resp. jeho definici), je (po důkladné úvaze a výběru) převzetí nejvhodnější kompletní metodiky a provedení jejích větších či menších úprav (postup „*shora dolů*“). To může znamenat přidání vlastních definovaných, odebrání nebo nahrazení několika jejích praktik prvky jiných metodik.

Dobrým příkladem je odvození procesů RUP a UPEDU od původní metodiky UP nebo nadstavba EUPnad RUP (všechny jmenované viz *2.4.2*).

2.5.2 Sestavení procesu z dílčích praktik definovaných metodik

Druhou možností je výběr sady vhodných praktik a postupů z různých metodik a vystavění vlastního procesu na jejich základě (postup „*zdola nahoru*“).

Vhodnými případy takovéhoto spojení fragmentů několika metodik jsou např. proces OpenUP (viz *2.4.2*), který spojuje UP (viz *2.4.2*Unified Proces) a agilní metodiky (viz *2.4.3*), DAD (viz *2.4.3*) tvořený sjednocením prvků UP, Scrumu (viz *2.4.3*) a několika dalších procesů, a také hlavní konkrétní proces zkoumaný v rámci této práce – proces KIV/ASWI (viz kapitola 6).

[13] Takovéto univerzální řešení je v softwarovém inženýrství často označováno jako „*silver bullet*“ (stříbrná kulka) a jeho dosavadní absence je frekventovaně zdůrazňována frází „*there's no silver bullet*“ (volně přeloženo „neexistuje jediné univerzální řešení“).

3. Modelování procesů pomocí specializovaných nástrojů

Tato kapitola popisuje pokročilé možnosti modelování a definování procesů vývoje software s využitím specializovaných softwarových nástrojů. V jejím úvodu je popsán samotný důvod k využití těchto nástrojů a následně jsou podrobně popsány možnosti a prvky popisu dostupné v jednom jejich konkrétním příkladu, *IBM Rational Method Composer* (RMC), který byl zvolen k detailnějšímu prozkoumání v rámci této diplomové práce.

Důvodem volby RMC byla jeho vazba na další produkt firmy *IBM*, tentokrát nástroj pro řízení projektů *Rational Team Concert*. Ten je využíván k řízení zhruba poloviny projektů v rámci předmětu KIV/ASWI na ZČU a model procesu tohoto předmětu a následné možnosti jeho převodu na šablonu importovatelnou do RTC byly hlavními body praktické části této diplomové práce.

3.1 Obecná motivace

Základní možnosti modelování procesů zmíněné v oddíle *2.3* mají po překročení určité hranice rozsahu projektů značně omezenou vypovídající hodnotu a při pokusu tímto způsobem modelovat proces s mnoha elementy jako celek rovněž narůstá úroveň jejich nepřehlednosti. Je tudíž nutností mít možnost popsat nebo modelovat proces efektivnějším, přehlednějším a srozumitelnějším způsobem.

Obecně vzato, definice a popis procesu může mít téměř libovolnou podobu v závislosti na jeho komplexnosti a nutnosti opakovaného využití. Může se například jednat o jednoduchý dokument popisující strukturu a základní principy daného procesu prostřednictvím prostého textu,rozličných modelů (např.: UML).

Se stále vzrůstající škálovatelností procesů a potřebou užívat jednotný model pro obsahově i rozsahově odlišné projekty v rámci společnosti nebo jiné software vyvíjející instituce roste však potřeba využití pokročilých softwarových nástrojů pro modelování procesů (nezávisle na konkrétních specifikáchtěchto procesů) se širokou a komplexní škálou funkcí.Příkladem takového nástroje je *EPF Composer*, který je využíván k publikaci popisu procesu metodiky OpenUP (viz *2.4.2*). Vzhled popisu OpenUP metodiky ve formátu HTML stránek, vytvořený v tomto nástroji je znázorněn na obrázku v příloze C[12].

Tato práce se však primárně zabývá obdobným nástrojem společnosti *IBM* s názvem *Rational Method Composer*.

3.2 O nástroji IBM Rational Method Composer

Pozn.: Veškeré reference, popisy a postupy provádění akcí v RMC uvedené v tomto textu se vztahují k verzi *RMC7.5.1*. Distribuce sebou nepřináší lokalizovanou verzi uživatelského prostředí v českém jazyce. Nicméně v následujícím textu jsou základní pojmy nástroje a struktury popisů v něm vytvořených přeloženy do češtiny s anglickými variantami v závorkách.

„Produkt IBM Rational Method Composer je pružná platforma pro správu procesů s nástrojem pro tvorbu metod a knihovnou procesů, jež v rámci podniku napomáhá implementovat měřitelná zlepšení, návrhy systémů a procesy poskytování softwaru. Nástroje Rational Method Composer umožňují vytvářet, upravovat, spravovat a publikovat popisy procesů. Knihovny procesů a postupů poskytují nejlepší postupy, které můžete přímo využívat v dodané podobě, nebo podle potřeby upravovat v rámci vytváření vlastních procesů.“[10]

„RMC je IDE (Integrated development environment) pro flexibilní tvorbu a management procesů. Obsahuje nejpoužívanější nástroje a bohaté knihovny. Je určen na pomoc společnostem pro implementaci efektivních procesů k nasazení na úspěšné SW a IT projekty.[11]

Rational Method Composer je nástroj od společnosti *IBM* určený pro tvorbu komplexních popisů procesů, jejich elementů a dokonce i popisů konkrétních projektů.RMC je integrován do vývojového prostředí *Eclipse*, s nímž je přímo distribuován. Částí distribuce je knihovna s kompletními popisy několika procesů na bázi metodiky RUP (viz *2.4.2*) a všech jejích úkolů, rolí, artefaktů, pomocných materiálů a dalších elementů, jež mohou uživatelé RMC využít k vytvoření vlastních modelů procesů.

Hlavním výstupem RMC jsou vytvořené konfigurace procesů publikované ve formátech HTML stránek, PDF dokumentu nebo dokumentu aplikace *Microsoft® Word®* s různými možnostmi nastavení formátu a přizpůsobení obsahu.

Hlavním důvodem pro volbu RMC byla jeho příslušnost k produktové řadě *Rational®* společnosti *IBM*, do které patří i nástroj pro řízení projektů *Rational Team Concert* (viz *4.4*) a možnost převodu modelů z RMC do systému RTC, jejíž prozkoumání bylo jedním z hlavních úkolů práce.

3.3 Uživatelské rozhraní RMC a jeho perspektivy

Základy uživatelského prostředí IDE *Eclipse*jsou dobře známy softwarovým vývojářům pracujícím s jazykem *Java*, uživatelům RTC nebo jiných *Rational* aplikací a mnoha jiným. Ze zřejmých důvodů obsáhlosti a divergence od hlavního předmětu této práce nebudou v tomto

textu popisovány. Zmíněny jsou pouze nutné části a prvky, o které IDE obohacuje samotné RMC, jako jsou např.: perspektivy.

Perspektiva IDE *Eclipse* je vlastně konfigurací uživatelského rozhraní definovanou zobrazenými podokny, také nazývanými jako pohledy (*Views*). V *Eclipse* obecně, tedy nejen v RMC, je možné použít jednu z předem definovaných perspektiv, upravit jí přidáním, odebráním nebo přeuspořádáním pohledů, a také vytvořit si zcela vlastní perspektivu včetně jejího pojmenování. Nicméně správa a práce s perspektivami obecně není předmětem této práce a v následujícím textu jsou tedy popsány pouze nejdůležitější předem definované perspektivy nástroje RMC.

RMC s sebou přináší tři hlavní perspektivy s různými účely.

- **Authoring** – Pro uživatele RMC je *Authoring* zřejmě nejdůležitější perspektiva. Je určena k samotnému vytváření modelů a popisů procesů a jejich elementů. Popis práce v perspektivě *Authoring* je hlavní náplní této kapitoly. Jedna z možností jejího rozložení a nejpodstatnější prvky, na které se odkazují další části textu, ukazuje obrázek v příloze C[12].Hlavní účelna obrázku označených oblastí UI je následující.
    - ***Editor Area*** – Hlavní pohled perspektivy *Authoring*, který umožňuje samotnou úpravu popisů jednotlivých prvků modelu procesu.
    - ***Library View*** – Pohled realizovaný formou stromové struktury knihovny(viz *3.4.1*), sloužící pro vyhledávání požadovaných prvků popisu procesu a jejich kontejnerů (viz *3.4.1*), jejich otevření pro editaci, kopírování, vkládání, mazání a další operace.
    - ***ConfigurationView*** – Pohled zobrazující strukturu momentálně otevřené konfigurace (viz *3.4.5*), čímž poskytuje představu o tom, jak bude vypadat obsah a struktura výstupních dokumentů (viz *685.2.1*).
    - ***Properties View***–Užitečný pohled zobrazující vlastnosti a atributy (včetně možnosti úpravy) označeného elementu při editaci procesního vzoru, nebo přímo celého procesu (viz*3.4.4* a *5.1.7*).
- **Browsing**–Perspektiva *Browsing* slouží k prohlížení vytvořených konfigurací procesů (viz *3.4.5*) před generováním jejich výstupních forem (viz*5.2.1*) a tím ověření, že výsledná podoba odpovídá představám uživatele RMC před publikováním procesů, které může být časově náročné. Perspektiva se chová jako interní webový prohlížeč prostředí *Eclipse* a do HTML formátu průběžně konvertuje pouze ty stránky konfigurace, na které chce uživatel přejít. Ukázka vzhledu perspektivy spolu s některými částmi uživatelského rozhraní, na které je v textu odkazováno, je na obrázku v příloze C.

- **Tailoring** – V perspektivě *Tailoring* se z modelu procesu vytváří popis konkrétního projektu. Tato funkce je určena zejména projektovým manažerům, ale především pokročilým a zkušeným uživatelům RMC. Popis použití této perspektivy, a tudíž postupu tvorby popisu konkrétního projektu v RMC, není předmětem diplomové práce a nebude v tomto textu detailněji dokumentován.

*Pozn.:*RMC má i další perspektivy jako je *Process Builder* a *Asset Management*. První je krátce zmíněna v sekci *5.1.5–Pomocný materiál*, druhá je součástí produktu *Rational Asset Manager*®distribuovaného spolu s RMC. Nicméně jejich využití nespadá do obsahu této práce, a proto není v tomto textu popsáno.

3.4 Struktura popisu procesu v RMC

Popis procesu vývoje software v nástroji RMC se skládá z mnoha prvků, z nichž některé představují základní elementy procesu, jako jsou role, aktivity, artefakty, některé jsou prostředky pro jejich seskupování a kategorizaci a některé představují samotné procesy či jejich části, nebo jsou pomocnými elementy pro export finálních verzí popisů. V tomto oddíle jsou všechny zásadní prvky popisu v RMC vyjmenovány a stručně zdokumentovány.

3.4.1 Kontejnery prvků popisu procesu v RMC

Než budeme moci popsat samotné prvky, z nichž se v RMC model procesu skládá, řekneme si něco o základních možnostech jejich kategorizace z pohledu uživatele RMC. Analogií pro tyto kontejnery jsou např.: složky v souborovém systému nebo projekty a balíky v rámci implementace kódu v jazyce *Java*.

Knihovna

Knihovna (*Method Library*)představuje nejvyšší úroveň kontejneru prvků v RMC a obsahuje všechny další kontejnery a všechny elementy, se kterými je momentálně možno manipulovat. V IDE může být v jednu danou chvíli otevřena pouze jedna knihovna. Knihovna je dále dělena na plug-iny.

Plug-in

Plug-in v RMC obvykle zapouzdřuje veškeré prvky, bez ohledu na jejich typ (tzn. role, aktivity i artefakty a všechny ostatní elementy) kromě konfigurace (viz *3.4.5*), ale obvykle tematicky spojené a určené pro modelování příbuzných procesů, nebo sloužící jako ucelené rozšíření jiného plug-inu (tak jak je tomu u plug-inů obecně i mimo RMC, např.: plug-iny IDE *Eclipse*). Jinak lze plug-in chápat i jako soubor prvků popisujících konkrétní metodiku, nebo jen její praktiku.

Úroveň segmentace a obsahu plug-inů je čistě v kompetenci uživatele. Analogií plug-inů v programování v jazyce *Java* by mohl být projekt. Plug-in může obsahovat jeden i více procesů a jeho prvky mohou odkazovat nebo být v relaci s prvky jiných plug-inů. Plug-iny mohou (ale nemusí) být také seskupovány do hierarchie balíků (*packages*) pomocí tečkové notace. Např.: *cz.zcu.plug-in_1*, *cz.kiv.plug-in_2*, *cz.kiv.plug-in_3*, *cz.plug-in_4*.

Každý plug-inje definován svým názvem, prezentačním názvem (rozdíl mezi názvem a prezentačním názvem je vysvětlen v sekci *5.1.3–Karta Popis*.), slovním popisem, popř.: referencemi na jiné plug-iny a verzí.

Plug-in obsahuje dvě základní složky: **Obsah metody**[14] (*Method Content*) a **Procesy** (*Processes*). Obsah metody je úložištěm všech elementů, ze kterých se popis procesu nebo jeho části (záleží na obsahu a účelu plug-inu) skládá,v podsložce **Balíky obsahu** (*Content Packages*) a jejich kategorií v podsložkách **Standardní kategorie** a **Vlastní kategorie** (*Standard Categories*, resp. *Custom Categories*). Složka Procesy pak uchovává samotné vytvořené procesy životního cyklu (*Delivery Processes*) a procesní vzory (*Capability Patterns*).

Balíky obsahu

Balíky obsahu (*Content Packages*) jsou v rámci plug-inu první úrovní kategorizace prvků procesu (z vývojářského pohledu) ovlivnitelnou uživatelem. Uživatel RMC si zde může vytvořit pro přehlednost a lepší vyhledávání libovolný počet balíků včetně jejich zanoření do hierarchie.Krom zanořených balíků může každý z nich obsahovat čtyři složky: **Role** (*Roles*), **Úkoly** (*Taks*), **Výsledky práce** (*Work Products*) a **Pomocné materiály** (*Guidance*), nebo jejich podmnožinu v závislosti na prvcích procesu, které balík obsahuje. Tyto složky jsou automaticky generovány a udržovány samotným RMC (tzn., že při vytvoření prvnírole se automaticky vytvoří složka **Role** a každá další role je do ní zařazena; to analogicky platí i pro další tři složky). Více o konkrétních typech prvků v oddíle *3.4.3*.

3.4.2 Prostředky pro kategorizaci prvků popisu procesu v RMC

RMC nabízí několik prostředků ke kategorizaci prvků procesu. Dvěma hlavními skupinami jsou standardní kategorie (*Standard Categories*) převážně dopředu vytvořené v knihovně distribuované přímo s RMC a vlastní kategorie (*Custom Categories*), které jsou zcela v režii uživatele.

Standardní kategorie

Standardní kategorie jsou předem vytvořené prostředky seskupování prvků nabízené samotným RMC. Dělí se na několik typů.

[14] Metodu v tomto kontextu lze vnímat jako praktiku, či celou metodiku.

- **Disciplíny (*Disciplines*)** – Disciplíny jsou prostředkem pro kategorizaci úkolů víceméně podle disciplín metodiky RUP (viz *2.4.2*). V knihovně distribuované spolu s *RMC7.5.1* jsou vytvořeny disciplíny:
  - ***Analysis & Design***,
  - ***Configuration & Change Management***,
  - ***Deployment***,
  - ***Environment***,
  - ***Implementation***,
  - ***Project Management***,
  - ***Requirements***,
  - ***Test***.
- **Domény (*Domains*)** – Domény jsou obdobou disciplín pro výsledky práce a ve standardní knihovně je jejich vytvořená sada obdobná, jako ta uvedená výše u disciplín.
- **Druhy výsledků práce (*Work Product Kinds*)** – Tyto kategorie jsou další možnosti rozdělování výsledků práce, tentokrát podle jejich charakteru a účelu. Ve standardní knihovně nabízené možnosti zahrnují:
  - ***Asessment*** (stanovení/ohodnocení/výměr),
  - ***Concept*** (koncept),
  - ***Infrastructure*** (infrastruktura),
  - ***Model*** (model),
  - ***Model Element*** (prvek modelu),
  - ***Plan*** (plán),
  - ***Project data*** (data projektu),
  - ***Solution*** (řešení),
  - ***Specification*** (specifikace).
- **Sady rolí (*Role Sets*)** – Tyto kategorie, jak už název napovídá, slouží k rozřazení rolí v procesu. V knihovně jsou některé zakomponovány, nicméně na rozdíl od obecné a téměř vše pokrývající kategorizace úkolů a výsledků práce (viz výše), jsou sady rolí více závislé na konkrétním modelu a proto více ponechány v režii uživatele.
- **Nástroje (*Tools*)** – Nástroje ve významu standardních kategorií RMC jsou prostředkem pro řazení jednoho speciálního druhu pomocným materiálů, kterými jsou návody k nástrojům (*Tool Mentors*;viz*3.4.3–Pomocné materiály*). Tudíž pro každý nástroj v procesu používaný se vytváří kategorie a do ní jsou vkládány všechny návody na jeho použití. Jako u rolí existují ve standardní knihovně i předem vytvořené nástroje, ale jejich palety v procesech používané jsou obvykle natolik podnikově, projektově nebo jinak závislé, že je běžnou praxí vytvořit si kategorie Nástroje pro svoje vlastní potřeby.

Vlastní kategorie

Pro další možnosti seskupování prvků procesu pro přehlednost a lepší vyhledávání během vytváření modelu, i během prohlížení samotných výsledných HTML a PDF popisů, jsou mimo standardních kategorií i vlastní kategorie (*Custom Categories*).Ty jsou zcela v režii uživatele vytvářejícího popis procesu a umožňují mimo jiné tvorbu hierarchie jednotlivých kategorií, možnost seskupování prvků do kategorií nezávisle na jejich typu (ve stejné kategorii mohou být zástupci úkolů, výsledků práce, pomocných materiálů, rolí i další kategorie) a několik možností jejich řazení. Vlastní kategorie se dále také využívají jako tzv. pohledy (*Views*) konfigurace procesu (více viz *3.4.5*)

Značky

Značky (*Tags*) jsou poslední možností kategorizace prvků popisu procesu, avšak poněkud odlišné od možností předcházejících, neboť nejsou zobrazovány ve stromové struktuře plug-inů v pohledu *Library View*. Nicméně přidání značky je možné k jakémukoli elementu knihovny, včetně samotných plug-inů, celých procesů, procesních vzorů i kategorií. Jednou z možností využití značek je automatické řazení prvků do vlastních kategorií při jejich vytváření (viz *5.1.8*). Pokročilejší možnosti využití značek nebyly v rámci této diplomové práce zkoumány, neboť se jedná o pokročilou techniku práce s RMC.

3.4.3 Vlastní prvky popisu procesu v RMC

Nyní můžeme přistoupit k představení a vysvětlení účelu jednotlivých prvků popisu procesu v RMC. Jak již bylo zmíněno v oddíle *3.4.1*, tyto prvky se ve struktuře plug-inu nacházejí v uživatelem vytvořených balících pod složkou **Balíky obsahu** (*Content Packages*).

Role

Role ve smyslu prvků modelu procesu v RMC jsou de facto přímou analogií rolí jako jedné ze základních složek popisu procesu obecně (viz *2.2.1*).Model procesu v RMC i obecně se nezajímá o mapování na konkrétní osoby, pouze určuje vztah rolí k aktivitám a artefaktům.

Úkoly

Úkoly představují činnosti v procesu vyžadující určité kvantum práce, času a zdrojů, vykonávanou jednou nebo více osobami v procesu zainteresovanými a vytvářející, měnící nebo používající některé z artefaktů procesu.

U této konkrétní skupiny elementů je velmi obtížné popsat jejich relaci k jedné z obecných složek procesů, konkrétně aktivitám (viz *2.2.3*). V terminologii RMC totiž existují aktivity (*Activities*), úkoly (*Tasks*) a kroky (*Steps*).

- **Úkoly** jsou prvky popisu procesu a v adresářové struktuře modelu každý z nich představuje jeden soubor.
- **Aktivity** jsou části procesu, do nichž jsou tyto úkoly, resp. jejich deskriptory (viz *5.1.7– Deskriptory a jejich odlišnost od standardních prvků popisu*), seskupovány pro větší přehlednost výsledného procesu.
- **Kroky** jsou pak jen jednou částí popisu úkolu, kterými je vyjádřen soubor sekvenčně po sobě následujících atomických a obvykle časově velmi krátkých činností nutných ke splnění úkolu.

Je tedy na uvážení uživatele jak daný proces rozdělí, a kde určí hranici granularity vhodnou pro aktivity, úkoly a kroky.

Pozn.: Konkrétně v rámci této diplomové práce byly úkoly definovány pro činnosti dobře definovatelné jak slovně, tak množstvím zdrojů (lidí, kteří je vykonávají, a artefaktů, které úkoly potřebují nebo vytvářejí), a s odhadem časové náročnosti obvykle v řádu maximálně jednotek hodin. Hlavním důvodem byl předpoklad, že po exportu procesů ve formě šablon pracovních položek (viz *5.2.2*) a následném vytvoření jeho šablony v systému pro řízení projektů *IBM Rational Team Concert* (viz *4.4* a *5.3*), což bylo jedním z hlavních cílů práce, budou úkoly RMC modelu mapovány na pracovní položky oblasti projektu v RTC.

Výsledky práce

Výsledky práce (*Work Products*) je v terminologii RMC míněna tatáž skupina prvků procesů, jako pojmem artefakty z oddílu*2.2.2*. Jedná se tedy o jakékoli objekty, či neživé entity nesoucí informace o projektu a jeho současném stavu.

V RMC je možné vytvářet výsledky práce tří různých typů podle jejich úrovně formality a složitosti formy. Bohužel jeden z typů výsledků práce je označen přímo slovem **Artefakt**, což poněkud znesnadňuje oddělení významu tohoto pojmu v kontextuRMC a v teorii procesů obecně.

Určení typu konkrétního vytvářeného výsledku práce je zcela na uvážení uživatele RMC. Zde je popsán jejich význam tak, jak byl aplikován v praktické části této diplomové práce.

- **Výstupy (*Outcomes*)** – Výstup představuje stav projektu nebo popis nějakého jeho aspektu v podobě neformálního média, jako např.: čistě myšlené nebo sesbírané informace, náčrt na tabuli nebo papíře, seznam nebo zápis schůzky v podobě neformálního dokumentu.

- **Artefakty (*Artifacts*)** – Artefaktem je v kontextu RMC myšlen obvykle dokument na určité úrovni formality obvykle ve formátu souboru, *wiki* stránky nebo vytištěného dokumentu.
- **Předávané (*Deliverables*)** – Pod tímto pojmem se skrývají velmi podstatné dokumenty často složené z několika seznamů či jiných artefaktů, jiné komplexní objekty přítomné v projektu, jako je vývojářská infrastruktura, vývojové prostředí, kompletní verze produktu včetně dokumentací či označená průběžná konfigurace v systému pro řízení verzí.

Tabulka 3.1– Typy pomocných materiálů v RMC

Typ Pomocného materiálu (Guidance)	Anglický název v RMC	Příklad užití	Možnost připojit ext. dokumenty
Seznam kroků	Checklist	kontrola činnosti, řízení schůzky	Ne
Koncept	Concept	popis konkrétní formy dokumentu, modelu nebo v praxi běžně používaného konceptu (use-case diagram, proof-of-concept, apod.)	Ne
Příklad	Example	konkrétní příklad artefaktu	Ano
Směrnice	Guideline	směrnice/návod pro práci	Ne
Faktor odhadu	Estimation Consideration	faktor při odhadování časové náročnosti úkolu	Ne
Praktika	Practice	definice praktiky (timeboxing, iterativní vývoj, apod.)	Ne
Hlášení	Report	zápis schůzky, hlášení o chybě	Ne
Znovupoužitelná položka	Reusable Asset	aktivní položka procesu s možností opakovaného využití	Ano
Rámcový harmonogram	Roadmap	rámcový harmonogram pro komplexní činnost	Ne

Podpůrný materiál	Supporting Material	obecný podpůrný materiál	Ne
Šablona	Template	šablona dokumentu	Ano
Definice pojmu	Term Definition	definice užívaného pojmu, položka glosáře	Ne
Návod k nástroji	Tool Mentor	návod pro konkrétní akci v nástroji	Ne
Informační dokument	Whitepaper	formální či oficiální dokument nesoucí informace využitelné v procesu	Ano

Pomocné materiály

RMC poskytuje možnost vytvářet i elementy představující např.: vzory dokumentů, šablony, návody k nástrojům, popisy konceptů a praktik. Ty jsou souhrnně označeny jako pomocné materiály (*Guidance*) a některé z nich mohou mít kromě vlastního popisu v RMC připojeny i zdroje ve formě dokumentů, či internetových stránek.Tyto prvky v procesu pomáhají čtenářům jeho popisu lépe pochopit konkrétní principy, postupy a definované formáty artefaktů.Seznam možných typů těchto pomocných materiálů a některé jejich další aspekty zachycuje tabulka *3.1*.

3.4.4 Popisy kompletních procesů nebo jejich částí v RMC

V RMC existují dva typy prvků představující sekvenční soustavu činností a tím pádem buď celý proces, nebo jeho část. Jsou jimi procesní vzory (*Capability Patterns*) a samotné procesy životního cyklu(*Delivery Processes*),a jak je uvedeno v oddíle *3.4.1–Plug-in*, jsou ve struktuře plug-inu uchovány ve složce **Procesy** (*Processes*).

V rámci těchto procesních modelů je možné definovat i některé další prvky jejich popisu, které nelze vytvořit ve výše popsané části plug-inu (obsahu metody – *Method Content*; viz *3.3.1–Plug-in*). Těmi jsou

- **fáze (*Phases*)**,
- **milníky (*Milestones*)**,
- **iterace (*Iterations*)**,
- **aktivity (*Activities*)**,
- **deskriptory úkolů (*Task Descriptors*)**.

Fáze, milníky a iterace jsou analogické s jejich teoretickými vzory popsanými v oddíle*2.4.2* (resp. sekci *Unified Process*) a aktivity z pohledu RMC byly vysvětleny v oddíle *3.4.3*,

sekci*Úkoly*. Deskriptory úkolů jsou v zásadě pouze jejich kopiemi a budou více popsány v sekci *5.1.7– Deskriptory a jejich odlišnost od standardních prvků popisu*.

*Pozn.:*Kromě výše popsaných pěti prvků, z nichž se struktury procesů nebo jejich částí mohou skládat, je možné použít i vnořené procesní vzory (viz následující sekce).

Procesní vzory

Podle definice v samotném RMC jsou procesní vzory (*Capability Patterns*) chápány jako obecné a znovu použitelné části procesu tvořené sekvencí úkolů (seskupených to fází, iterací nebo vnořených procesních vzorů) vhodné k lehčímu modelování procesů vzájemně příbuzných, lišících se pouze škálou nebo specifickým zaměřením.Může se jednat o kostru procesu tvořenou pouze prázdnými fázemi a milníky nebo o část procesu, která se ve stejné nebo téměř stejné podobě v jeho struktuře opakuje, jako např. iterace nebo některá z aktivit.

Procesy životního cyklu

Posledním možným typem prvku plug-inu jsou samotné procesy životního cyklu (*Delivery Processes*). Ty se od procesních vzorů liší pouze tím, že popisují kompletní strukturu celých modelovaných procesů, a jsou obvykle nejdůležitější částí modelů v pokročilých nástrojích a hlavním důvodem jejich užívání. Např. popis OpenUP metodiky v *EPF Composeru*, jehož hlavní částí je právě „*Delivery Process*“ složený z „*Capability Patterns*“ a dalších prvků (viz [4])

3.4.5 Konfigurace

Posledním, ale obzvláště pro generování popisů projektů do výstupních dokumentů nezbytným, prvkem RMC jsou konfigurace. Ty stojí zcela mimo plug-iny a ve stromové struktuře knihovny v pohledu *Library View* je jejich složka**Konfigurace** (*Configurations*) s nimi na stejné úrovni.

Konfigurace je prvek, který určuje finální podobu výstupních dokumentů RMC. Kromě obecných popisů a množství voleb pro vzhled a úpravu obsahu výstupů, je definován především tím, které plug-iny nebo jejich vybrané části (včetně procesů životního cyklu) v ní budou zahrnuty, a budou tak obsaženy ve finálním výstupu.

Dalším významným atributem konfigurace jsou pohledy (*Views*). Ty jsou dány vybranými vlastními kategoriemi a určují uspořádání obsahu výstupních dokumentů. Každá konfigurace musí pro úspěšné generování výstupu, neboli publikování, obsahovat alespoň jeden pohled.

3.5 Shrnutí

V této kapitole byly zevrubně popsány všechny základní prvky RMC modelu procesu. Jejich širší popisy a návod pro manipulace s nimi za účelem sestavení kompletního modelu procesu a jeho publikování ve výstupních formátech budou uvedeny v kapitole 55.1.

4. Systémy pro správu projektů

Při stupňujícím se rozsahu a komplexnosti obsahu jednotlivých softwarových projektů vzrůstá nejen potřeba využití specifických aplikací pro popis procesů a jejich pokročilých funkcí (viz kapitola 3), ale i potřeba zapojení nástrojů pro vytváření plánů projektů, kontrolování jejich plnění, správu všech artefaktů, dohledatelnosti jejich vazeb na autory, úkoly a konkrétní konfigurace produktu a obecnou kontrolu a správu stavů projektů. Tato kapitola se zabývá obecným popisem, základními funkcemi a konkrétními příklady těchto nástrojů.

4.1 Obecný popis a funkce ALM nástrojů

Softwarovým produktům pro komplexní správu projektů se v odborné praxi souhrnně říká *nástroje pro správu projektů*, resp.*nástroje pro správu změn averzí*, nástroje pro *CCM* (*Configuration and Change Management*), nebo také *ALM* (*Application Lifecycle Management*) nástroje.

*Pozn.:*Existují i nástroje pouze pro správu změn, tzv. *bugtrackery* (příkladem může být *Flyspray*), i nástroje pouze pro správu verzí (konfigurací), tzv. *SCM* (*Software Configuration Management*) nástroje (např.: *Subversion*).

Podle zdroje [12] musí nástroj disponovat funkcemi pro(nebo alespoň podporovat) několik oblastí správy projektů, aby mohl být klasifikován jako ALM nástroj. Množina těchto oblastí je následující:

- správa požadavků (**Requirements Management**),
- správa testů, jejich provádění a výsledků (**Test Management**),
- správa průběhu projektu (**Project Management**),
- správa případů užití produktu (**Use Case Management**),
- správa úkolů (**Issue Management**),
- správa změn (**Change Management**),
- správa vydávání verzí (**Release Management**),
- správa iterací (**Iteration Management**),
- správa spolupráce (**Collaboration Management**).

To v praxi znamená především možnosti vytváření projektů a jejich plánů ve formě dekomponované na fáze, které tvoří soubory úkolů, možnost přístupu celého vývojového týmu, přikládání dokumentů, připojení na úložiště zdrojových souborů a ostatních artefaktů v rámci projektu, ukládání a označování stavů úložiště, automatické generování grafů, seznamu aktualit, hlášení o blížícím se datu plánovaného vydání verze (tzv. *release*) produktu, komunikační nástroje pro synchronizaci týmu, a další.

Dalšími nároky na ALM nástroj jsou celková integrace jednotlivých oblastí správy, kompletní dohledatelnost vazeb mezi úkoly, artefakty, jejich autory, plány, vydávanými verzemi, konfiguracemi a fázemi projektu, do kterých patří, a přístup přes jednotné uživatelské rozhraní. To v uživateli budí dojem, že pracuje s jediným systémem, přičemž v realitě jde o řadu integrovaných softwarových komponent.

Softwarových produktů vybavených funkcemi pro výše zmíněné účely je celá řada. Tato kapitola se v krátkosti zabývá třemi z nich. Zvláštní pozornost je pak věnována aplikaci *IBM Rational Team Concert*, která patří do stejné řady produktů jako RMC (viz kapitola 3), a jejíž možnosti použití šablon pracovních položek exportovaných právě z RMC (viz *5.2.2*) je jedním z hlavních předmětů zkoumání v rámci této diplomové práce.

4.2 Redmine

Prvním příkladem nástrojů pro správu projektů je otevřený (*opensource*) softwarový produkt *Redmine*, který stručně charakterizuje následující citát z jejích webových stránek.

„Redmine is a flexible project management web application. Written using the Ruby on Rails framework, it is cross-platform and cross-database. Redmine is open source and released under the terms of the GNU General Public License v2 (GPL).“[13]

Nástroj *Redmine* je distribuován ve formě serverové aplikace (aktuálně ve verzi *2.3.1*), k níž se po instalaci přistupuje přes webové rozhraní v jakémkoli internetovém prohlížeči (ukázka je na obrázku v příloze D[12]).

Mezi možnosti*Redmine*, kroměobecných funkcí ALM nástrojů popsaných v oddíle *4.1*, patří:

- využití vestavěných *wiki* stránek,
- nahrávání přídavných externích dokumentů,
- fóra,
- kalendář,
- informačních emaily zasílané členům týmu v projektu,
- široká škála dalších funkcí a nastavení v závislosti na konfiguraci serveru a úrovni oprávnění uživatelů zapojených v projektu.

Redmine nemá přímo vestavěný systém pro správu verzí produktu, je však snadno propojitelný se systémy*Subversion*, *Mercurial*, *CVS* a *Git*, které tuto funkci obstarávají.

Redmine je po RTC druhým nástrojem pro správu používaným v projektech v rámci předmětu KIV/ASWI (viz kapitola 6), konkrétně ve spojení se systémem *Subversion* pro správu verzí.

4.3 Atlassian®Jira®

Dalším v praxi široce užívaným nástrojem pro správu projektů je systém *Jira®*, která je produktem společnosti *Atlassian®* a momentálně je k dostání ve verzi *5.2*. Kromě komerční verze je dostupná i časově omezená licence zdarma. Nástroj Jira v krátkosti popisuje následující citace.

„JIRA is the project tracker for teams planning, building and launching great products.Thousands of teams choose JIRA to capture and organize issues, work through action items, and stay up-to-date with team activity.“[14]

Parametry a funkce nástroje *Jira* jsou v mnoha ohledech obdobné jako u systému *Redmine* a ukázka jeho uživatelského prostředí je na obrázku v příloze D[12].

Specifiky nástroje jsou například plug-in *GreenHopper* zaměření na plánování agilně zaměřených projektů, propojení s nástrojem *Confluence* pro sdílení artefaktů a kolaboraci zeměpisně oddělených týmů, vlastní dotazovací jazyk pro komplexní vyhledávání *JIRA Query Language*, apod. Mezi zákazníky firmy *Atlassian* užívající nástroj *Jira* patří *Apache Software Foundation*, *Air France*, americká televizní stanice *CBS* nebo plzeňská společnost *Eurosoftware* a široká škála dalších společností a institucí po celém světě.

Detailnější informace jsou k dispozici na internetových stránkách společnosti *Atlassian* (viz [14]).

4.4 IBM®Rational TeamConcert®

Produkt *Rational Team Concert®* společnosti *IBM®* je posledním a nejdůležitějším zástupcem nástrojů pro správu projektů konzultovaným v tomto textu. Tento nástroj je spolu s *Redmine* používán pro řízení projektů v rámci předmětu KIV/ASWI (viz kapitole 6) na Západočeské univerzitě v Plzni.

RTCje součástí většího balíku softwarových produktů s názvem *Rational Solution for Collaborative Lifecycle Management®*je momentálně dostupný v aktuální verzi *4.2.0*. Verze pro 10 a méně uživatelů s časově omezenými licencemi je dostupná zdarma. Nástroj je stručně charakterizován následující citací z internetových stránek společnosti IBM.

„IBM Rational Team Concert integrates task tracking, source control, and agile planning with continuous builds and a configurable process to adapt to the way you work.“[15]

Od dříve zmiňovaných nástrojů *Redmine* a *Jira* se RTC liší ve dvou základních aspektech.

1) RTC se skládá ze serverové aplikace s uživatelským rozhraním přístupným přes internetový prohlížeč (ukázka je na obrázku v příloze D) a klientské části, která může

být integrována v IDE *Eclipse* (ukázka na obrázku v příloze D) nebo *Microsoft® Visual Studio®*.

2) Součástí RTC je i integrovaný systém pro správu verzí vyvíjeného produktu (to je patrné na obrázku v přílozeD). Je tak komplexním nástrojem pro podporu vývoje softwarových produktů, k němuž uživatelé již nepotřebují žádné další aplikace.

Pro tento text jsou zásadními i některá terminologická označení používaná v RTC. Samotné projekty se v RTC označují jako ***oblasti projektu***. Jednotlivé položky konkrétních úkolů, v jejich rámci s atributy jako priority, odhad časové náročnosti, atd. (tím pádem vlastně položky *backlogu*) se pak v terminologii RTC nazývají ***pracovní položky***(work items).

Předmětem této práce není podrobně popisovat možnosti a funkce nástroje RTC. Informace o nich jsou dostupné ve zdrojích [15] a [16]. V oddíle *5.3* v následující kapitole jsou uvedeny pouze informace podstatné pro importování šablon pracovních položek z nástroje RMC a jejich připojení a využití do šablon projektů v RTC, což je jedním z hlavních cílů této diplomové práce.

5. Postup tvorby popisu procesu v RMC a jeho přenos do RTC

Tato kapitola zevrubně popisuje kompletní postup a specifika modelování procesu a tvorbu jeho popisů v RMC a generování jeho výstupů. Dále pak kapitola obsahuje popis postupu importu jednoho z výstupů – konkrétně šablon pracovních položek – do systému RTC a jejich zapojení do celkové šablony oblasti projektu využitelné v tomto nástroji.

Celá kapitola tak může sloužit jako návod pro komplexní operace tvorby popisu procesu v RMC, jeho převodu do šablony pro projekty v RTC a jejich následné propojení. Komplexní návod tohoto postupu v českém jazyce doposud neexistuje, což je také jedním z důvodů jeho zanesení do tohoto textu.

5.1 Postup modelování procesu v RMC

Tento oddíl popisuje podrobněji postup vytváření jednotlivých prvků a jejich skládání do celkových modelů procesů a může tak spolu s popisy v oddíle *3.4* sloužit jako návod pro používání RMC pro uživatele začátečníky.

Pozn.: Komplexnější informace a návody jednotlivých činností manipulace s RMC jsou velmi dobře rozepsány v samotné nápovědě nástroje (viz [17]), jejíž jedinou nevýhodou je absence verze v českém jazyce. Dalšími zdroji informací o RMC v českém jazyce jsou například výstupní dokument oborového projektu Eduarda Chromíka ze ZČU z roku 2011 (viz [11]) nebo diplomová práce Radka Kohúta z *Masarykovyuniverzity v Brně* (viz [18]).

5.1.1 Přípravné kroky

Po spuštění programu je uživateli prezentována uvítací obrazovka s několika volbami. Před samotným používáním nástroje je vhodné přečíst si alespoň několik úvodních kapitol nápovědy produktu dostupných pod volbou „*Overview*“. Po té je rovněž doporučeno projít několik výukových lekcí s praktickými cvičeními pod volbami „*First Steps*“ a „*Tutorials*“. Ty poskytují prostřednictvím ukázkových a pomocných procesů, konfigurací a dalších elementů RMCdostatečnou bázi dovedností pro první kroky tvorby vlastního popisu procesu.

Pro přechod přímo do vlastního vývoje slouží volby označená „*Workbench*“, kteráotevře samotné uživatelské prostředí RMC v perspektivě *Authoring* (viz *3.3*).

5.1.2 Otevření knihovny

Program by měl při spuštění automaticky otevřít standardní knihovnu distribuovanou s RMC. Pokud se tak nestane, je možné tuto nebo jinou knihovnu otevřít užitím hlavního menu UI.

Pozn.: Veškeré prvky standardní knihovny nelze přímo editovat. Jejich využívání a drobné úpravy pro potřeby popisu vlastního procesu je prováděno pomocí jedné z možností tzv.

proměnlivosti obsahu (*Content Variability*) při vytváření vlastních prvků popisu procesu (viz *5.1.3– Provázání obsahu elementů popisu*).

Je také možné začít vytvářet zcela novou knihovnu bez otevření některé již existující, nicméně pro nové uživatele RMC je vhodnější mít oporu a vzor v knihovně distribuované s programem a vytvářet v ní pouze vlastní plug-iny.

Pozn.: Zde popisovaná verze *RMC7.5.1* byla v době zahájení práce distribuována se dvěma verzemi knihovny a sice *lib.7.5.rup* a *lib.7.5.0.1.prac*[15]. Obě obsahují kompletní popisy metodiky RUP a několik procesů různých rozsahů na ní postavených. Liší se však organizací prvků a jejich kontejnerů, způsobem tvorby plug-inů a sestavování procesu. Pro použití v rámci této diplomové práce byla proto vybrána první zmiňovaná, neboť je srozumitelnější a lépe uchopitelná pro nové uživatele RMC.

5.1.3 Aspekty úpravy popisů prvků procesu společné více typům prvků

Některé z operací vytváření a editace jednotlivých elementů plug-inů jsou shodné nebo obdobné pro většinu typů těchto elementů. Ty jsou proto shrnuty v tomto oddíle. Pro definice jednotlivých typů elementů viz *3.4*.

Většina základních operací, jako je vytváření, přesuny, kopírování a mazání elementů, je dostupná a intuitivně proveditelná přes kontextové menu vyvolané stiskem pravého tlačítka myši nad konkrétním uzlem ve stromové struktuře knihovny v pohledu *Library View* (viz obrázek v příloze C[12]). Specifika jednotlivých typů prků při jejich vytváření, úpravě a dalších operacích jsou pak popsány v oddílu *5.1.5*.

Vytvoření prvku

Kontextové menu RMC vždy umožňuje pouze vytvoření takového prvku, který je podřízen (obvykle přímo) kontejneru, nad kterým bylo vyvoláno. Po zvolení prvku, který chce uživatel vytvořit, se buď objeví dialogové okno se základními nastaveními vlastností prvku, nebo se nově vytvořený prvek automaticky otevře pro úpravy v pohledu *Editor Area* (viz obrázek v příloze C).

Karta Popis

Editace popisu prvku v pohledu *Editor Area* (viz obrázek v příloze C) je u většiny prvků rozdělena na několik karet, každou se speciálním účelem.

První z nich je vždy karta **Popis** (*Description*), na které je možné zadat nebo změnit základní informace o prvku. Karta je přítomná v editaci všech elementů plug-inu včetně plug-inu

[15] Práce s touto verzí knihovny využívá funkce sestavení procesů s využitím perspektivy *Process Builder* (viz *3.3*)

samotného, balíku obsahu a konfigurace. Níže jsou popsány její jednotlivá pole, z nichž všechny se nemusí nutně objevit u všech typů elementů, ale jsou zásadní složkou popisu většiny z nich.

- ***Name*** – V tomto textovém poli by měl být uveden hlavní název elementu. Tento název nebude zobrazován ve výstupních dokumentech RMC a jedná se vlastně o identifikátor, který musí být alespoň v rámci daného typu elementu a balíku obsahu unikátní. RMCrovněž uchovává většinu elementů ve formě samostatných souborů, označených právě hodnotou tohoto pole. Proto by měl název mít odpovídající formu, tj. skládat se pouze z neakcentovaných malých písmen, číslic a dalších obecně povolených znaků v názvech souborů (např.: podtržítka nebo pomlčky) a nezačínat číslicí.
- ***Presentation name*** – Prezentační název představuje titul elementu ve formě, v jaké bude zobrazen ve výstupních dokumentech RMC. Proto v tomto poli neexistují restrikce jako u pole „*Name“*.
- ***Brief description*** – Pole slouží k uchování stručného popisu/definice/významu elementu.
- ***Tags*** – Celá sekce **Značky** (*Tags*) karty **Popis** slouží k připojení značek k elementu. Popis manipulace se značkami bude blíže vysvětlen v oddíle *5.1.8*.
- ***Main description***[16] – Pole pro hlavní, kompletní a detailní popis elementu v míře, kterou si zvolí uživatel.
- ***Key considerations***[16] – V tomto poli je vhodné uvést hlavní aspekty elementu, jež je třeba vzít v úvahu při jeho použití v procesu z pohledu čtenáře finálního dokumentu s popisem procesu. Na příklad u úkolu: jaké faktory zvážit při jeho plnění, apod.
- ***Version iformation*** – Protože je RMC vybaveno vnitřním verzovacím systémem, je na kartě **Popis** sekce **Informace o verzi** (*Version information*) a poli **Verze** (*Version*), **Datum změny** (*Change date*), **Popis změny** (*Change description*), **Autoři** (*Authors*) **Poznámka o autorských právech** (*Copyright*). Poznámkou o autorských právech a jejím využitím se mimo jiné krátce zabývá oddíl*5.1.9*Poznámka o autorských právech.
- ***Content variability***– Sekce karty **Popis** zabývající se nastavením proměnlivosti obsahu, resp. vztahu nově vytvářeného elementu k některému z již existujících. Touto tématikou se zabývá speciální sekce *Provázání obsahu elementů popisu*níže.
- ***Icon*** **–** Sekce karty **Popis** umožňující přidat k elementu vlastní ikonu různou od té, kterou má implicitně nastavenu.

[16] Pole je vybaveno pokročilými možnostmi formátování svého obsahu. Ty jsou dostupné po kliknutí na ikonu s popisem „*Open rich text editor*“ nalevo od názvu pole.

Provázání obsahu elementů popisu

V RMC je možné několika způsoby provázat obsah nově vytvořeného elementu s obsahem některého z prvků odpovídajícího typu z původní knihovny nebo jiného již dříve vytvořeného. K tomu slouží sekce „*Content variability*“ na kartě **Popis.** Ta nabízí pět hodnot pole „*Variability type*“, které vyjadřují povahu vztahu elementu k jeho vzoru, a pole „*Base*“, ve kterém se po jeho připojení objeví vzor elementu.

Možnosti relace a jejich významy jsou následující:

- ***Not applicable*** – Element je zcela soběstačný a není v relaci s žádným vzorem.
- ***Contributes*** – Při volbě tohoto vztahu element přispívá k obsahu svého vzoru. To v praxi znamená, že při jeho zařazení do struktury procesu element převezme popisy svého vzoru a přidá k nim svůj vlastní obsah. Konkrétně slovní popisy v polích na kartě **Popis** příspěvkového elementu budou přidány za převzatý text vzorů. Pole „*Presentation name*“ v popisu elementu, který přispívá k jinému, nemá význam, neboť jeho obsah bude bez úprav převzat ze vzoru. Oproti tomu na patřičných místech budou přidány vztahy s ostatními elementy (pomocnými materiály, rolemi, výsledky práce, atd.) definované v přispívajícím elementu k původním množinám vzoru. Je dokonce možné u úkolu přidat kroky (viz *5.1.5–Úkol*) a vložit je před, mezi nebo za kroky původního úkolu.
- ***Extends*** – Při tomto typu relace element rozšiřuje svůj vzor. To znamená, že přebírá jeho textové popisy v těch polích, které sám nemá vyplněné, a v ostatních texty nahrazuje vlastními. Vliv na relace s ostatními elementy je obdobný jako u typu „*Contributes*“.
- ***Replaces*** – Element nahrazuje svůj vzor ve všech jeho relacích s ostatními prvky, přičemž nepřebírá nic z textů vzoru. Toho lze využít, chceme-li použít knihovní element se všemi jeho definovanými vazbami na elementy ostatní, ale nevyhovuje nám jeho slovní popis.
- ***Extends and Replaces*** – Poslední možnost provázání obsahů elementů kombinuje předchozí dva typy vazby. Nahrazuje vzorový element ve všech jeho relacích s ostatními prvky, přebírá jeho textové popisy, tam kde je aktuální element nemá definovány, a nahrazuje ostatní.

Pozn.: Pro více informací o možnostech a detailech jednotlivých typů provázání viz [17].

Karta Pomocné materiály

Karta **Pomocné materiály** (*Guidance*) je společná pro všechny základní prvky popisu procesy a jejich standardní kategorie (tzn. úkoly, role, výsledky práce, pomocné materiály, sady rolí, nástroje, disciplíny, domény a druhy výsledků práce).

Na kartě je pouze seznam pomocných materiálů (viz *3.4.3*)připojených k aktuálně editovanému elementu, který lze snadno upravovat pomocí tlačítek „*Add…*“ (a následného dialogového okna; pro přidání reference na pomocný materiál k aktuálnímu elementu) a „*Remove*“ (pro odstranění reference na označenou položku seznamu).

*Pozn.:*Pro manipulaci se seznamy kategorií, jakož i jinými seznamy v pohledu *Editor Area*, nelze využít funkci*drag and drop*.

Karta Kategorie

Karta **Kategorie** (*Categories*) je společná v editaci úkolů, výsledků práce a rolí.

Obsahuje dva seznamy kategorií: **Standardní kategorie** (resp. u rolí **Sady rolí**) a **Vlastní kategorie**. Manipulace se seznamy je prostá a využívá tlačítka „*Add…*“ a následného dialogového okna pro zařazení elementu do další kategorie a tlačítka „*Remove*“ pro vyřazení elementu z kategorie označené v seznamu.

Karta Náhled

Karta označená jako **Náhled** (*Preview*) zobrazuje element otevřený v pohledu *Editor Area* v podobě HTML stránky tak, jak bude vypadat po publikaci konfigurace procesu (viz *5.2.1*).

Jedná se vlastně o stejnou funkci, kterou poskytuje i perspektiva *Browsing* (viz *3.3*) s tím, že je generována pouze stránka aktuálního elementu a nikoli celé konfigurace, což je mnohdy časově náročné (podle rozsahu konfigurace).

Kartou disponuje pohled pro úpravu všech elementů, které mohou být viditelné čtenáři publikované verze konfigurace kromě procesních vzorů a procesů samotných. To zahrnuje úkoly, role, výsledky práce, pomocné materiály a všechny jejich standardní i vlastní kategorie.

Kopírování prvků

Kopírování všech prvků, včetně např.: i balíků obsahu, je realizováno standardním způsobem přes schránku. Nejjednoduššími způsoby provedení jsou volby „*Copy*“ a „*Paste*“ v kontextovém menu nad prvkem, který chcete kopírovat, resp. kontejnerem, kam chcete prvek vložit, nebo obdobně pomocí známých klávesových zkratek (*Ctrl + C, Ctrl + V*).

5.1.4 Založení plug-inu

Pro vytvoření plug-inu lze opět využít kontextové menu a následný dialog UI. RMC nabízí následujících několik možností založení plug-inu.

- **Vytvoření prázdného plug-inu**.
- **Vytvoření plug-in ze šablony** – umožňuje vytvořit plug-in s určitými přednastavenými vlastnostmi a atributy z jedné z nabízených šablon podle účelu, ke kterému má plug-in sloužit.
- **Vytvoření plug-inu z jednoho z již existujících plug-inů nebo balíků** – vytvoří v podstatě kopii již existujícího plug-inu nebo balíku několika plug-inů.
- **Vytvořit příspěvkový plug-in** – vytvoří balík obsahující pouze příspěvkové prvky(*contributors*) rozšiřující elementy některého z již existujících plug-inů.

Plug-iny je také možné kategorizovat do balíků. Ty nelze založit přímo přes volbu některé z položek menu, ale pro jejich vytvoření nebo zařazení plug-inu do existující hierarchie balíků stačí použít tečkovou notaci v poli „*Name*“ plug-inu (např. „*cz.zcu.muj_plug-in*“ vytvoří plug-in s názvem „*muj_plug-in*“ v balíku „*zcu*“ umístěném v nadřazeném balíku „*cz*“).

Kromě běžných polí popsaných v oddíle *5.1.3–Karta Popis* disponuje karta **Popis** plug-inu rovněž sekcí „*Referenced Plug-ins*“ se seznamem plug-inů, na něž se aktuální odvolává (je s nimi v nějakém vztahu). Manipulace se seznamem je obdobná jako se seznamy na kartách **Pomocné materiály** a **Kategorie** v odpovídajících sekcích oddílu *5.1.3*.

5.1.5 Specifika popisů a manipulace s jednotlivými typy prvků RMC

V tomto oddíle jsou popsány karty, pole a seznamy popisu a postupy manipulace specifické pro konkrétní typy prvků modelu procesu v RMC.

Balík obsahu

Vytvoření balíku obsahu se provádí přes kontextové menu vyvolané nad složkou **Obsah metody** (*Method Content*), popř. nad některým z již existujících balíků, chcete-li vytvořit balík vnořený.

Jelikož balíky nejsou částí výstupních dokumentů (jedná pouze o kontejnery pro zpřehlednění a kategorizaci úkolů, rolí, pomocných materiálů a výsledků práce), jsou možnosti editace balíků malé a omezují se pouze na pole „*Name*“, „*Presentation name*“, „*Brief description*“ a sekci **Značky** (*Tags*). Ve spodní sekci karty **Popis** je pak pouze neupravitelný seznam ostatních balíků, na jejichž prvky mají ty uložené v aktuálním balíku vazbu („*Dependencies*“).

Funkcí společnou pro balíky a prvky jejich obsahu je možnost přesunu, resp. položka „*Move*“ v jejich kontextovém menu. Pomocí ní je možné přesouvat balíky (nebo jejich vybrané prvky)

v rámci jejich hierarchie. Ovlivnit (ze zřejmých důvodů) nelze pouze rozdělení balíku na složky **Role** (*Roles*), **Úkoly** (*Tasks*), **Výsledky práce** (*Work Products*) a **Pomocné materiály** (*Guidance*), které jsou automaticky generovány RMC podle obsahu konkrétního balíku.

Role

Kromě základních karet a upravitelných polí popisu, má pohled editace role (*Editor Area*) navíc pouze katru **Výsledky práce** (*Work products*). Ta obsahuje dva seznamy artefaktů.

Prvním je seznam výsledků práce, za něž je osoba v dané roli zodpovědná („*Responsible for*“), Přidání či odebrání výsledků práce ze seznamu se provádí stejným způsobem, jako u seznamů již dříve v tomto textu popsaných (tlačítka „*Add…*“ a „*Remove*“).

Druhým je neupravitelný seznam výsledků práce takových úkolů, v jejichž popisu je daná role uvedena jako tzv. primární vykonavatel („*Primary performer*“; viz následující sekce*Úkol*).

Úkol

Při editaci úkolu jsou oproti obecným prvkům na kartě **Popis** dvě další textová pole, konkrétně „*Purpose*“, kde je možné uvést hlavní účel daného úkolu, a „*Alternatives*“ sloužící k zanesení popisu alternativního nebo nestandardního průběhu plnění úkolu a jeho podmínky.

Obě pole jsou stejně jako „*Main description*“ a „*Key considerations*“ vybaveny pokročilými možnostmi formátování svého obsahu. Ty jsou dostupné po kliknutí na ikonu s popisem „*Open rich text editor*“ nalevo od názvu pole.

Dalším specifikem popisu úkolu je karta **Kroky** (*Steps*). Zde je možné v případě potřeby vytvořit seznam kroků vedoucích ke splnění úkolu a jejich popisů. Kroky lze rovněž upravovat, mazat nebo přeuspořádat pomocí ovládacích prvků na kartě.

Další specifickou kartou editace úkolu je karta **Role**(*Roles*). Ta obsahuje dva seznamy rolí:**Primární vykonavatelé** (*Primary performers*) a **Další vykonavatelé** (*Additional performers*). Jinými slovy v nich lze určit, které role se plnění úkolu mají účastnit povinně a účast kterých je v rámci úkolu volitelná, či závislá na okolnostech.

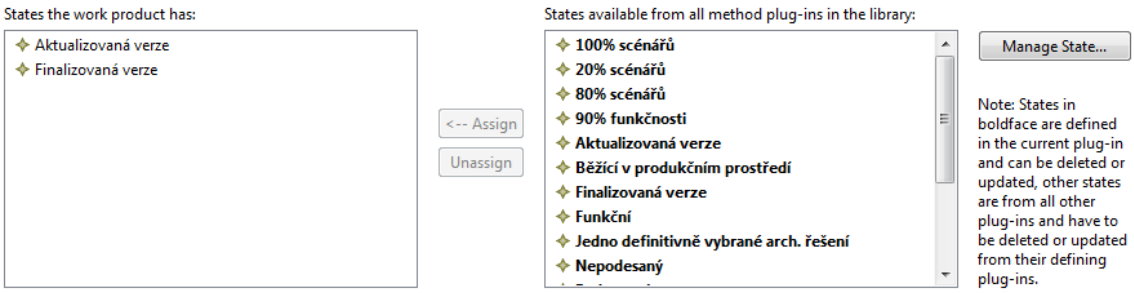
Třetí specifickou kartou úkolu je karta **Výsledky práce** (*Work Products*). Zde lze opětupravovat následující tři seznamy, které označují vstupní a výstupní artefakty úkolu:**Povinné vstupy** (*Mandatory inputs*),**Volitelné vstupy** (*Optional inputs*)a **Výstupy** (*Outputs*).

Pozn.: Editace úkolu obsahuje ještě kartu **Odhad** (*Estimation*). Ta se používá k zdokumentování rámcového odhadu časové náročnosti úkolu. To je však pokročilejší technikou, kterou se nezabývá tato práce.

Výsledek práce

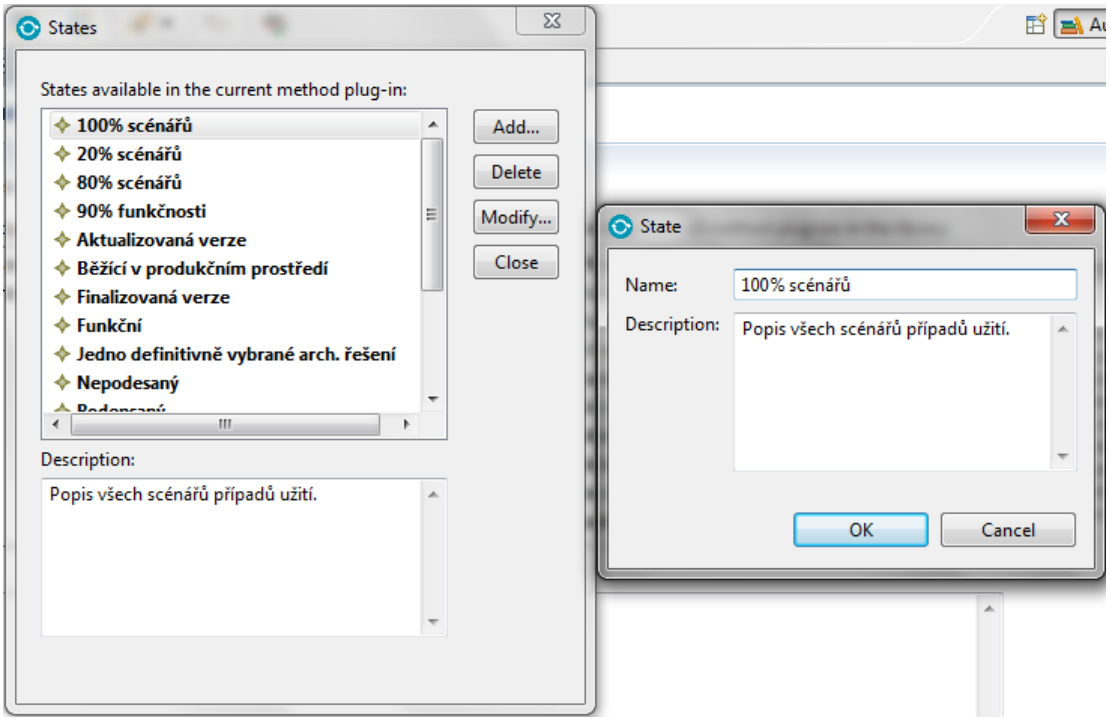
Při zakládání nového výsledku práce neexistuje v kontextovém menu položka „*Work Product*“. Je totiž nutné rovnou určit jeden z možných typů výsledku práce(**Výstup**, **Artefakt** nebo **Předávaná**; viz *3.4.3–Výsledky práce*).

Editace všech tří typů výsledků práce má dvě společná specifika. Prvním je pole „*Purpose*“ na kartě **Popis** shodné s tím u úkolů (viz předcházející sekce*Úkol*).



Obrázek 5.1– Část karty Stavy

Druhým je karta **Stavy** (*States*), kde lze definovat množinu stavů, ve kterých se výsledek práce může během projektů nacházet, a jejichž zavedení má význam pro informační hodnotu celého modelu procesu. Nejpodstatnější část karty **Stavy** je na obrázku *5.1*.



Obrázek 5.2– Dialogové okno správy stavů

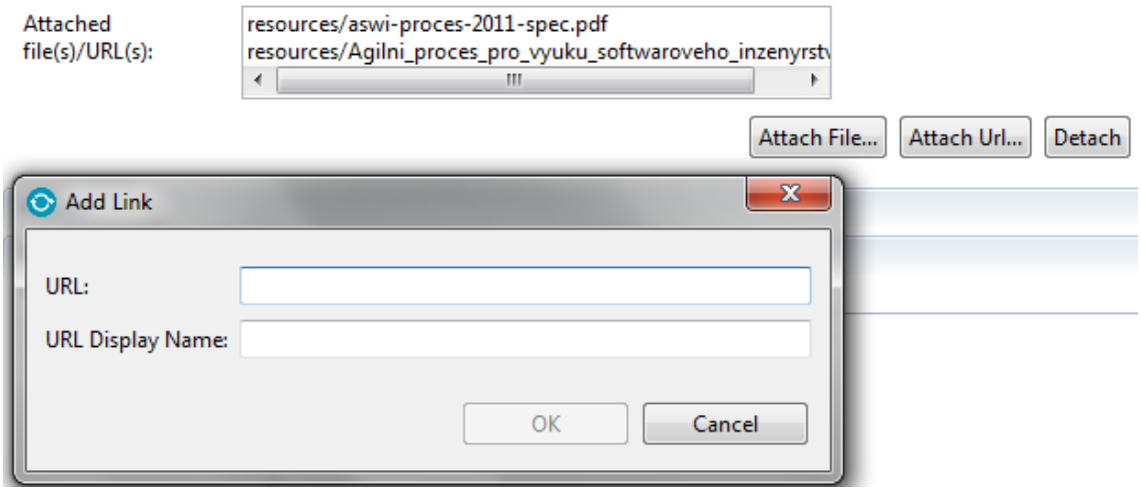
Pro správu množiny stavů (vytváření, úpravu a mazání) přidělitelných výsledku práce slouží tlačítko „*Manage States*“a v následující dialogová oknana obrázku *5.2*.

Pozn.:Je důležité zdůraznit, že úpravy a smazání stavu se projeví u všech výsledků práce, kde byly dané stavy použity.

Výsledek práce typu **Předávaná** (*Deliverable*) má navíc ještě jednu specifickou kartu. Tou je karta **Části předávané** (*Deliverable Parts*). V ní lze spravovat seznam jiných výsledků práce, které jsou součástí této předávané, neboť předávané jsou často složeny z několika artefaktů (např. průběžná konfigurace vyvíjeného software v nástrojích pro správu verzí, kompletní předávaný produkt i s dokumentací nebo i dokument *Vize* složený z jednotlivých dílčích popisů stakeholderů, rizik, požadavků, atd.).

Pomocný materiál

Většina typů pomocných materiálů v RMC (které zachycuje tabulka *3.1*) nemá oproti obecným prostředkům popisu v oddílu *5.1.3* žádná specifika. Existují pouze dvě výjimky.



Obrázek 5.3– Pole a dialogované okno pro přidání externích dokumentů

První výjimkou je možnost připojení externích dokumentů u pomocných materiálů typu **Příklad** (*Example*), **Znovupoužitelná položka** (*Reusable Asset*), **Šablona** (*Template*) a **Informační dokument** (*Whitepaper*). Pole pro toto vložení dokumentů s názvem **Připojené soubory/URL** (*Attached file(s)/URL(s)*) se nachází na kartě **Popis**. Sekce karty je vybavena tlačítky pro snadné přidání a odebrání dokumentů ze souborů nebo odkazů na webové stránky (viz obrázek *5.3*).

Druhou výjimku tvoří typ pomocného materiálu **Praktika** (*Practice*). Ta má ve své editaci na místo karty **Pomocné materiály** kartu **Odkazy** (*References*). Zde je možné spravovat seznam všech prvků, které náleží do popisu právě upravované praktiky. Do praktiky lze přidat odkaz na jakýkoli element RMC popisu procesu včetně standardních a vlastních kategorií, balíků obsahu, celých plug-inů, kompletních procesů nebo jejich vybraných částí. Seznam je vybaven ovládacími prvky pro přidání, odebrání a řazení svých položek.

Pozn.: Praktiky jako kontejnery plug-inů využívá knihovna *lib.7.5.0.1.prac* a perspektiva *Process Builder* k odlišnému způsobu sestavování popisu procesu, než který je popisován v této diplomové práci. Pro více informací a návody na tento postup využijte nápovědu a návody v samotném RMC (viz [17]).

Standardní kategorie

Jak víme ze sekce*3.4.2*, standardní kategorie jsou několika typů.

Obecně se nedoporučuje, obzvláště novým uživatelům RMC, vytvářet nové disciplíny (*Disciplines*), domény (*Domains*) a druhy výsledků práce (*Work Product Kinds*), aleraději možností vlastních kategorií (*Custom Categories*). Nicméně vytváření sad rolí (*Role Sets*) a nástrojů (*Tools*) je bez rizika a tvoří součást běžné praxe užívání nástroje RMC.

Pozn.: Disciplíny a sady rolí lze u rozsáhlých plug-inů dále kategorizovat do skupin (*Discipline Groupings*, resp. *Role Set Groupings*).

Popis standardních kategorií má v podstatě pouze jedno specifikum, a to kartu se seznamem elementů toho typu, k jejichž kategorizaci jsou určeny (to znamená karta **Úkoly** u disciplín, karta **Role** u sad rolí, karta **Návody pro nástroje** u nástrojů, atd.). Princip jejich editace je však vždy stejný.

Kromě standardní práce se seznamem je zajímavou funkcí možnost automatického řazení pomocí ovládacího prvku nadepsaného „*Sort Type*“. Nabízené způsoby řazení jsou:

- ***Manual*** – ruční,
- ***Alphabetic*** – abecedně vzestupně,
- ***Reverse alphabetic*** – abecedně sestupně,
- ***Element Type*** – podle typu elementu.

*Pozn.:*U popisu disciplín lze navíc nalézt kartu **Odkazovaný tok práce** (*Reference Workflow*). Její využití však nebylo v rámci této diplomové práce prozkoumáno.

Vlastní kategorie

Při editaci vlastní kategorie si lze všimnout karty **Přiřadit** (*Assign*). Její princip je naprosto stejný jako u specializovaných karet elementů u standardních kategorií (viz předchozí sekce *Standardní kategorie*) s tím rozdílem, že do vlastní kategorie je možné přiřadit jakýkoli element bez ohledu na jeho typ.

Druhou speciální kartou vlastních kategorií je karta **Dotaz** (*Query*). Zde je možné definovat podmínky, za kterých je nově vytvořený element popisu procesu zařazen do této kategorie.

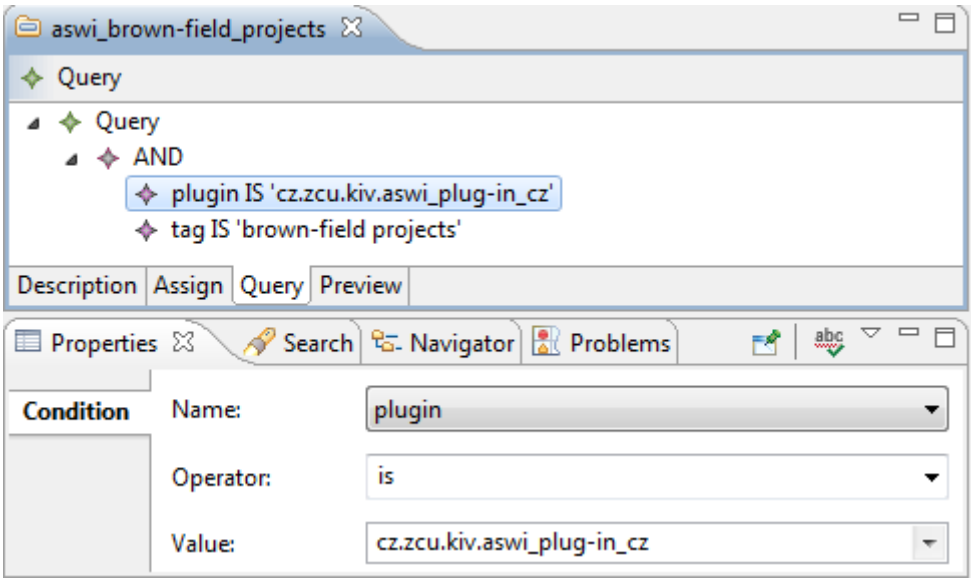
V kontextovém menu nad uzlem „*Query*“ se nachází volba „*New Child* > *Condition Group*“, neboli vytvoření podřazené skupiny podmínek. Po označení skupiny podmínek je možné v pohledu *Properties View* (viz obrázek v příloze C[12]) nastavit, zda pravdivostní hodnotu skupiny bude určovat konjunkce (*AND*) nebo disjunkce (*OR*) hodnot jejích potomků.

Tabulka 5.1– Možnosti nastavení podmínky pro zařazení elementu do standardní kategorie

Hodnota pole „Name“	Hodnota pole „Operator“	Možné hodnoty pole „Value“	Podmínka je splněna pokud …
plugin	is	existující plug-inu	element je částí vybraného plug-inu
	is not		element není částí vybraného plug-inu
tag	is	existující značka	element je označen vybranou značkou
	is not		element není označen vybranou značkou
type	is	typ elementu (úkol, role, plug-in, skupina disciplín, šablona, atd.)	element má stanovený typ
	is not		element nemá stanovený typ

Volbou „*New child* > *Condition*“ nad skupinou podmínek se do ní vloží nová podmínka. Tu lze sestavit v pohledu *Properties View* ze tří částí. Možnosti nastavení podmínky demonstruje tabulka *5.1*.

Podmínky a jejich skupiny lze různě kombinovat a zanořovat pomocí voleb „*New Child*“ (nový podřazený prvek) a „*New Sibling*“ (nový souřadný prvek) v kontextových menu nad nimi vyvolanými.



Obrázek 5.4– Příklad a nastavení dotazu

Příklad sestavené dotazu a možnosti jeho nastavení demonstruje obrázek *5.4*.

Mezi specifika vlastních kategorií patří postup jejich přesouvání v hierarchii stromové struktury knihovny v pohledu *Library View*, přesněji řečeno v hierarchii vlastních kategorií jako takových (do jiných částí plug-inu pochopitelně být přesunuty nemohou). Ten je totiž lehce odlišný od přesouvání balíků obsahu a jejich prvků popsaného v sekci *Balík obsahu* tohoto oddílu (viz výše). Pro přesun kategorie z jedné části složky **Vlastní kategorie** do druhé je nutné nad požadovanou kategorií vyvolat kontextové menu a zvolit položku „*Reassign*“ a v následném dialogovém okně vybrat nové umístění kategorie.

5.1.6 Vytvoření konfigurace

Před samotným vytvořením procesu (popř. procesního vzoru) a skládáním jeho struktury je nutné vytvořit konfiguraci, protože každému novému procesu je při jeho vytvoření povinné zadat hlavní konfiguraci, do níž patří.

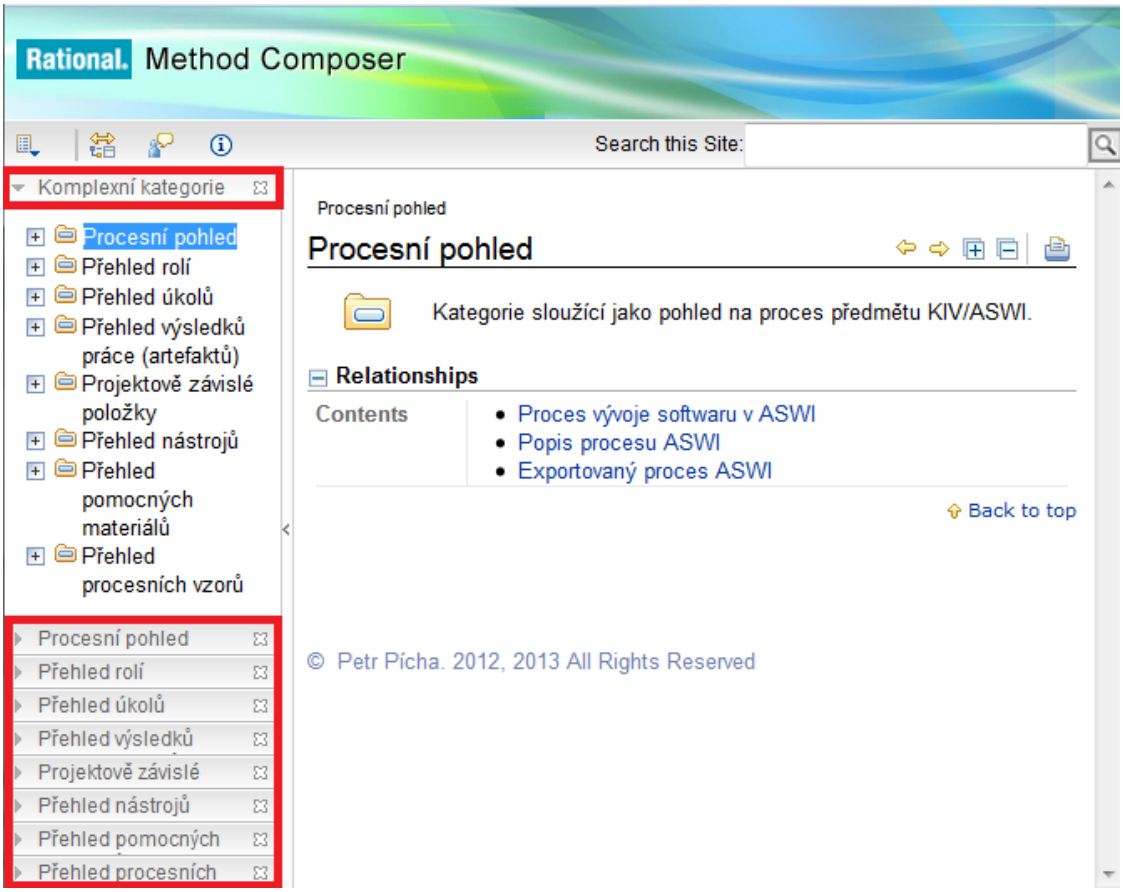
Postup vytvoření konfigurace začíná stejným krokem jako vytvoření jakéhokoli jiného elementu, ovšem tentokrát vyvoláním kontextového menu nad složkou **Konfigurace** (*Configurations*) stojící na stejné úrovni s jednotlivými plug-iny. Na kartě **Popis** nemá nová konfigurace žádné nové prvky. Zato všechny ostatní karty její editace jsou v tuto chvíli uživateli zcela nové.

Pozn.: Jednotlivé ovládací prvky a volby na většině karet editace konfigurace v tomto oddíle nebudou popsány, jelikož by se ve většině případů jednalo pouze o vypisování anglických označení voleb ovlivňujících formát a detaily obsahu publikované verze konfigurace

v dokumentech a jejich překlad do českého jazyka. Některé z voleb užité v konkrétní konfiguraci procesu předmětu KIV/ASWI jsou zmíněny v oddíle *6.2.5*.

Karta Výběr plug-inů a balíků

První z nových karet má název **Výběr plug-inů a balíků** (*Plug-in and Package Selection*) a slouží k označení plug-inů nebo jejich částí, které mají být v konfiguraci zařazeny. Uživatel je zde rovněž upozorňován na chyby, nekonzistence a konflikty ve výběru jak vyznačením plug-inů a balíků, které jsou jejich zdrojem v jejich seznamu označeném „*Content*“, tak především v pohledu „*Problems View*“, který se při zjištění problémů automaticky otevře.



Obrázek 5.5– Ukázka vzhledu konfigurace ve formě publikovaného HTML dokumentu

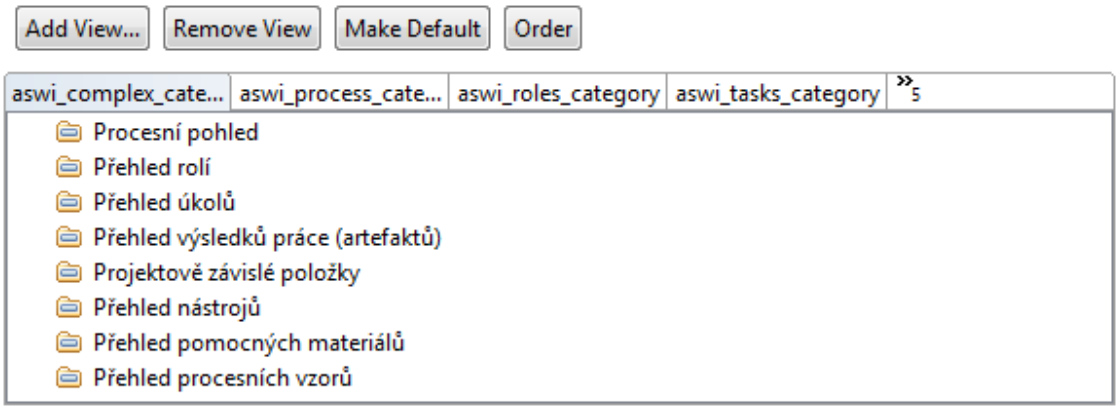
Karta Pohledy

Druhou a pro publikování konfigurace nejpodstatnější kartou je karta **Pohledy** (*Views*). Pohledy mají největší význam pro publikovanou konfiguraci ve formátu HTML (viz *5.2.1*). Tvoří jednotlivé záložky celého HTML dokumentu, kterými je kategorizován jeho obsah (obvykle velmi rozsáhlý) pro větší přehlednost. Ukázka celkové podoby publikovaného HTML dokumentu konfigurace s červeně vyznačenými záložkami jednotlivých pohledů je na obrázku *5.5*.

Na kartě **Pohledy**se nachází ovládací prvky pro správu pohledů. Každý pohled je definován prostřednictvím vytvořené vlastní kategorie a v některých případech i kategorie standardní (pro oboje viz oddíl *5.1.5*). Každá konfigurace by měla obsahovat alespoň jeden pohled, aby její publikovaná forma měla smysl.

Přidané pohledy jsou reprezentovány záložkami v horní části pole, a to samotné pak zobrazuje strukturu vybrané vlastní kategorie, resp. pohledu. Toto pole nepovoluje editaci pohledů a v případě potřeby je tedy nutné editovat přímo samotnou vlastní kategorii.

Při vytvoření více než jednoho pohledu by měl jeden z nich být označen jako výchozí tlačítkem „*Make Default*“. Ten bude pak zobrazen automaticky při otevření HTML dokumentu konfigurace.



Obrázek 5.6– Pole pro správu pohledů konfigurace

Seznam pohledů konfigurace a ovládací prvky pro jejich správu ukazuje obrázek *5.6*.

Ostatní karty editace konfigurace

Další karty označené jako **Obecné možnosti publikování** (*Publish General Options*), **Možnosti publikování HTML** (*Publish HTML Options*) a **Možnosti publikování dokumentu** (*Publish Doc Options*) slouží k nastavení různých voleb a aspektů pro publikování konfigurace, z nichž některé budou popsány v oddíle *6.2.5*, jak již bylo řečeno výše v této části textu.

5.1.7 Práce s procesy

Postup práce s procesními vzory (*Capability Patterns*) a samotnými procesy životního cyklu (*Delivery Processes*) je totožný, a proto bude v tomto oddíle popsán nezávisle na tom, o který z těchto dvou typů elementu se jedná.

Vytvoření

Pro vytvoření procesu (resp. procesního vzoru) opět stačí použít kontextové menu nad složkou ***Delivery Processes*** (resp. ***Capability Patterns***). Ta se nachází v nadřazené

složce**Procesy**(*Processes*), která je hierarchicky přímo podřazena plug-in v pohledu *Library View*. Jak již bylo řečeno dříve, procesu (resp. procesnímu vzoru) je nutné při vytvoření zadat výchozí konfiguraci.

Kontejnery procesů

Protože procesy a vzory nejsou součástí obsahu metody (**Method Content**; viz *3.4.1*) a nelze je tak seskupovat do balíků obsahu (viz *3.4.1* a *5.1.5*), je možné vytvářet také **Balíky procesů** (*Process Packages*) pomocí stejného kontextového menu, jako u vytváření samotných procesů.

Specifika karty Popis

Karta **Popis** má při úpravě procesů a vzorů několik specifických polí pro jejich přesnější vymezení.

Hlavním novým prvkem karty je ale sekce ***Configurations*** na jejím konci, která umožňuje spravovat odkazy na konfigurace, kterých je proces součástí. Výchozí konfigurace procesu definovanou při jeho vytváření lze změnit pomocí tlačítka „*Make Default*“.

Deskriptory a jejich odlišnost od standardních prvků popisu

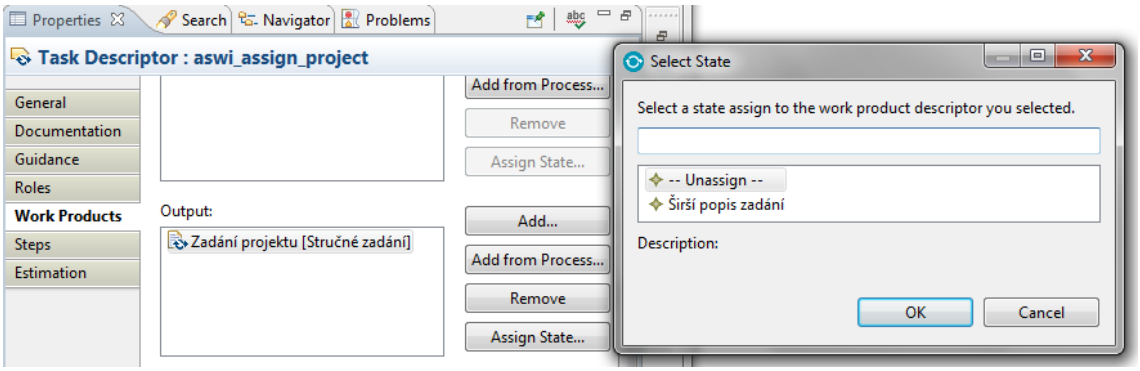
Než budeme moci popsat ostatní karty editace procesu, je nutné definovat pojem v této oblasti práce s RMC často používaný, a sice pojem *deskriptor*.

Deskriptor je vlastně jakýmsi obrazem nebo kopií prvku popisu ve struktuře samotného procesu. Deskriptory se objevují ve třech typech podle toho, jestli jsou obrazy úkolů, výsledků nebo rolí[17]. Deskriptory lze upravovat a odlišit je tak od původních prvků, jichž jsou kopiemi. Vazbu na svůj vzor si však zachovávají.

Editace deskriptorů je přístupná v pohledu *Properties View* (viz obrázek v přílozeE[12]) při jejich označení na příslušné kartě popisu procesu (viz dále). Karty popisu deskriptorů jsou víceméně shodné s kartami samotných úkolů, výsledků práce a rolí s tím rozdílem, že karta **Popis** se rozpadla na karty **Obecné** (*General*) a **Dokumentace** (*Documentation*). Pokud má deskriptor některé pole nevyplněné, přebírá hodnotu tohoto pole od svého vzoru.

Kromě změn názvu a dalších textových popisů, lze konkrétně u deskriptorů úkolů například omezit množinu kroků, přidat nebo odebrat vstupní a výstupní výsledky práce, včetně určení jejich konkrétního stavu (viz *5.1.5–Výsledek práce*), role a pracovní pomůcky s ním spjaté.

[17] Konkrétně se tedy jedná o elementy typu „***Task Descriptor***“, „***Work Product Descriptor***“ a „***Role Descriptor***“.



Obrázek 5.7–Přiřazování stavu k deskriptoru výsledku práce

Pro přidání stavu k výsledku práce je karta**Výsledky práce** (*Work Products*) v pohledu *Properties View* příslušného deskriptoru vybavena tlačítkem „*Assign State…*“ a následným dialogovým oknems nabídkou přístupných stavůdaného výsledku práce (viz *3.5.5*). Vybraný stav bude u výsledku práce zobrazen v hranatých závorkách za jeho názvem (viz obrázek *5.7*).

Karta Struktura rozkladu práce

Nejvýznamnější kartou při tvorbě popisu projektu je karta **Struktura rozkladu práce** (*Work Breakdown Structure*), která zobrazuje strukturu procesu složenou z jednotlivých úkolů a jejich seskupování do fází, iterací a aktivit. Obsah karty je organizován do formy tabulky. Její příklad pro účely demonstrace a popisu ukazuje obrázek v příloze E[12].

Ve sloupci „*Presentation Name*“ jsou uvedeny prezentační názvy jednotlivých prvků a rovněž znázorněna jejich hierarchie. Ve sloupci „*Predecesors*“ jsou uvedeni předchůdci prvku, tj. ty, které mu sekvenčně předcházejí v diagramu aktivit (viz sekce *Diagramy*dále v tomto oddílu). Sloupec užívá identifikátorů prvků ze sloupce „*Index*“. Sloupec „*Model Info*“ uchovává informace o vazbě prvků na jejich vzor (*Content Variability*; viz*5.1.3–Provázání obsahu elementů popisu*) v případě, že taková vazba existuje. Sloupec „*Type*“ informuje o typu prvku. Může nabývat hodnot:

- ***Delivery Proces*** – obvykle pouze kořenoví uzel (první řádek) u modelu procesu,
- ***Capability Pattern*** – kořenoví uzel u procesních vzorů, nebo vnořený procesní vzor,
- ***Phase*** – definovaná fáze procesu (viz sekce *Skládání procesu* dále v tomto oddíle),
- ***Milestone*** – milník obvykle ukončující fázi (viz sekce *Skládání procesu* dále v tomto oddíle),
- ***Iteration*** – definovaná iterace procesu (viz sekce *Skládání procesu* dále v tomto oddíle),
- ***Activity*** – aktivita; seskupení deskriptorů úkolů (viz sekce *Skládání procesu* dále v tomto oddíle),
- ***Task Descriptor*** – deskriptor úkolu (viz předchozí sekce *Deskriptory a jejich odlišnost od standardních prvků popisu*).

Dalších šest sloupců pak jen zobrazuje nastavitelné atributy prvků. Těch může být využito pro jejich přesnější popis, ale významnější roli hrají při vytváření instancí procesu (projektů) v perspektivě *Tailoring* (viz *3.3*). Následuje jejich přehled a význam po zaškrtnutí políčka ve sloupci.

- ***Planned*** – prvek je plánován (využívá se především k označení prvků procesu pro export – viz *5.2.2*).
- ***Repeatable*** – prvek je opakovatelný.
- ***Multiple Occurrences*** – prvek se může objevit několikrát[18].
- ***Ongoing*** – prvek je průběžný, tzn. soustavně aktivní v rámci nějakého časového období.
- ***Event-driven*** – výskyt prvku v procesu nebo časový bod zahájení jeho průběhu je podmíněn nějakou událostí z vnějšku projektu.
- ***Optional*** – výskyt prvku v procesu je volitelný.

Ostatní karty editace procesu

Další kartou přítomnou při editaci procesu je karta **Alokace týmu** (*Team Allocation*). Ta zobrazuje stejnou strukturu dělení procesu na fáze, iterace a aktivity, ovšem místo deskriptorů úkolů ukazuje rozložení deskriptorů rolí.

Obdobná je situace i u karty **Užití výsledků práce** (*Work Product Usage*), která stejným způsobem zobrazuje rozložení a využití výsledků práce v procesu pomocí jejich deskriptorů.

Informace z předchozích třech karet pak sumarizuje karta **Souhrnný pohled** (*Consolidated View*), která zobrazuje ve struktuře procesu všechny přítomné deskriptory bez ohledu na jejich typ.

Poslední karta **Odhad** (*Estimation*) pak ve struktuře procesu zobrazuje informace o odhadech časové náročnosti jednotlivých úkolů, resp. jejich deskriptorů, tak jak byly zadány na stejnojmenné kartě popisů úkolů (viz *5.1.5–Úkol*).

Pozn.: Využití předešlých čtyř popsaných karet představuje prvky pokročilejší práce s RMC a vyžadují větší míru zkušeností s nástrojem, než jaká byla získána v průběhu této diplomové práce. Stejně tak manipulace s nimi není doporučena novým uživatelům RMC.

[18] Rozdíl mezi opakovatelností a několikanásobnou přítomností prvku není zcela jasný. Zřejmě se ale jedná o rozdíl na úrovni pravidelného opakování se stejným průběhem (***Repeatable***) a nepravidelného opětovného vykonání stejného úkolu nad jinou množinou dat (***Multiple Occurences***).

Skládání procesu

Sestavení vlastního procesu, resp. procesního vzoru, je v nástroji RMC obecně možné provést mnoha způsoby. Tento oddíl proto popisuje ten, který byl použit k modelování konkrétního procesu předmětu KIV/ASWI (viz *6.2*) v rámci praktické části této diplomové práce.

Pro sestavení procesu je třeba otevřít jeho položku knihovny RMC pro editaci (v pohledu *Editor Area*) a přejít na kartu **Struktura rozkladu práce** (*Work Breakdown Structure*). Pokud je proces nově vytvořený, je v tabulce na kartě pouze kořenový uzel, představující celý proces. Vyvoláním kontextového menu nad tímto uzlem a zvolením položky „*New Child*“ lze do procesu přidat fázi (*Phase*), iteraci (*Iteration*), aktivitu (*Activity*), deskriptor úkolu[19] (*Task deskriptor*) nebo milník (*Milestone*). Tímto způsobem lze přidávat další a další prvky do struktury procesu využitím voleb „*New Child*“ a „*New Sibling*“ z kontextového menu podle toho, zda je třeba nový prvek přidat na stejnou nebo nižší úroveň vzhledem ke zvolenému elementu[20].

Nově vytvořené fáze, iterace, aktivity a milníky lze po jejich pojmenování upravovat v pohledu *Properties View* (viz obrázky v přílohách C aE[12]) stejným způsobem jako deskriptory úkolů (viz sekce o deskriptorech)

K zařazení deskriptoru do struktury procesu lze využít techniku *drag and drop*, tedy přesouvání úkolů z hierarchie knihovny (pohledu *Library View*) na požadované místo ve struktuře procesu. Stejným způsobem lze přidávat i fáze, iterace a milníky jiného procesu nebo celé procesní vzory (*Capability Patterns*). V některých případech je uživatel po jejich přidání nucen vybrat jednu z následujících tří možností kopírování.

- ***Extend***– vloží kopii prvku závislou na svém zdroji, tzn., že v současném umístění jí nelze upravit nebo změnit její vnořené části ani jejich popisy a každá změna zdrojového elementu se promítne do této jeho kopie. To se týká i diagramů aktivit (viz sekce *Diagramy* dále v tomto oddílu).
- ***Copy***– vloží kopii prvku nezávislou na zdroji, tzn., že ji lze volně upravovat, ale každá změna, kterou uživatel provede na zdroji a má být aplikována i v kopii, musí být provedena ručně.
- ***Deep Copy***– možnost kopírování více využívaná v perspektivě *Tailoring* (viz*3.3*). V postupu sestavování procesu jí není potřeba.

[19] Přidávání deskriptorů úkolů tímto způsobem není doporučeno pro nové uživatele. Vhodnější způsob je popsán dále v textu.
[20] Ze zjevných důvodů menu nad kořenovým uzlem nedisponuje položkou „*New Sibling*“, stejně jako menu nad deskriptorem úkolu nemá položku „*New Child*“.

U kopírování pomocí možnosti „*Extend*“ budou neupravitelné části procesu v tabulce jeho struktury (resp. jejich názvy) označeny zeleným, kurzívním písmem. Do jejich nadřazeného segmentu procesu (fáze, iterace, atd.) je možné přidat další prvek. Ten ale může být umístěn před soubor zeleně označených sousedních prvků nebo za něj, nikoli mezi ně.

Pozn.: Technikou *drag and drop* nelze přesouvat prvky do struktury procesu z balíků obsahu ani ze složky **Procesy**. Proto je nutné přesouvat elementy ze standardní a vlastních kategorií z pohledu *Library V*iew nebo z pohledu *Configuration View* (viz obrázek v příloze C[12]).

Pro přesun položek ve struktuře procesu lze opět využíttechniku*drag and drop*. U přesunu deskriptorů funguje tato technika však pouze v rámci aktivity nebo jiného nejbližšího nadřazeného celku.Dalším, nicméně ne vždy použitelným, způsobem jsou volby „*Move Up*“ a „*Move Down*“ pro posun prvku v rámci nejbližšího nadřazeného celku (aktivity, iterace, fáze, atd.) a volby „*Indent*“ a „*Outdent*“ pro zařazení prvku do nebližšího s ním souřadného elementu, resp. o úroveň výš (to je na stejnou úroveň s jeho momentálně nadřazeným prvkem).

Po sestavení procesu a přepnutí do jedné z dalších karet jeho editace je možné pozorovat, že role a výsledky práce (resp. jejich deskriptory), které mají vazbu na úkoly, jejichž deskriptory byly zařazeny do procesu, byly do struktury rovněž automaticky přidány. Proto se uživatel o ostatní karty editace procesu zabývající se rolemi a výsledky práce (viz předcházející sekce *Ostatní karty editace procesu*) nemusí starat.

Synchronizace prvků metody a procesů

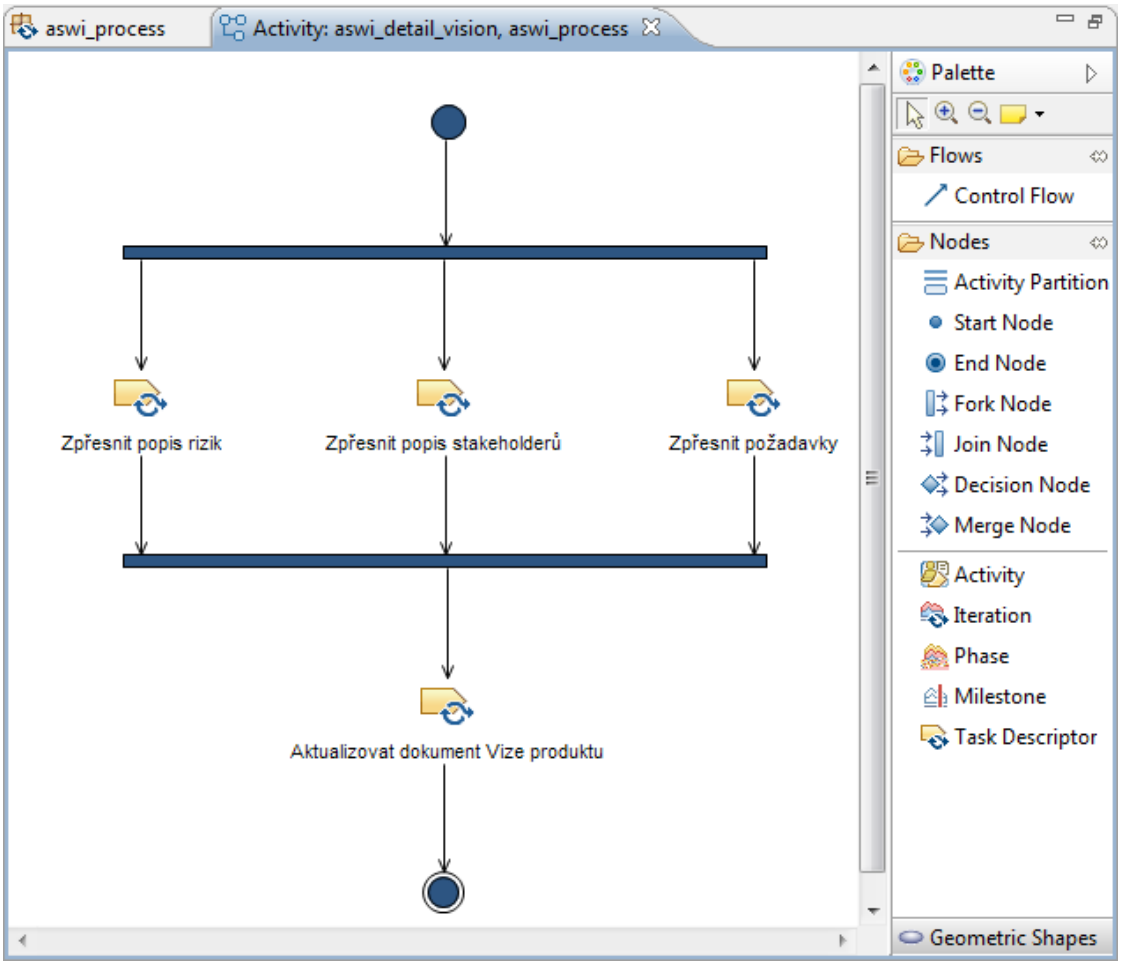
Pokud dojde k úpravě popisu úkolu, výsledku práce či role po zařazení jeho deskriptoru do struktury procesu, změna se v deskriptorech automaticky neprojeví. Je proto vhodné po každé takové změně (nebo jejich větším počtu) využít volbu „*Default Synchronization from Method Content*“ z kontextového menu nad patřičnými deskriptory, jejich nadřazenými položkami nebo celým procesem. Ta promítne provedené změny elementů do deskriptorů.

Příjemnou vlastností této operace je, že popisy samotných deskriptorů, které byly odlišeny od jejich vzorů, zůstanou nezměněny. Nezmění se tak upřesněné názvy a popisy deskriptorů, ani například stavy přidané k výsledkům práce.

Diagramy

RMC (resp. jeho vývojáři) si uvědomuje, že slovní popisy prvků procesu a jejich provázání odkazy mnohdy nestačí. Daleko větší informační hodnotu mají v mnoha případech obrázky a diagramy. A proto jsou při modelování procesů v RMC dostupné hned tři typy snadno generovatelných diagramů.

Prvním a zřejmě nejdůležitějším z nich je **diagram aktivit** (resp. *workflow diagram*) zmiňovaný obecně již v oddíle *2.3.1*. Ten je v RMC možné vytvořit pro každý prvek struktury procesu, mající v sobě vnořené další elementy, tzn. proces samotný, procesní vzor, fázi, iteraci a aktivitu.



Obrázek 5.8 – Editor diagramu aktivity

Jeho editace je přístupná pod položkou „*Open Activity Diagram*“ v kontextovém menu požadovaného prvku struktury procesu na jedné z jeho karet.K úpravě diagramu lze použít prostředky z nabídnuté palety viz obrázek *5.8*.

Jednotlivé objekty v modelu není nutné zarovnávat a uspořádávat ručně. Po jejich propojení šipkami stačí pouze v kontextovém menu nad prázdnou plochou plátna zvolit položku „*Arrange All*“, a editor sám uspořádá prvky v digramu tak, jak je schopen nejlépe v závislosti na jejich propojení.

Druhým diagramem v RMC je **diagram detailu aktivity** v podobě, v jaké byl demonstrován při jeho teoretickém popisu v oddíle *2.3.2*.

Tento diagram je v RMC přístupný pouze pro aktivity. K jeho zobrazení slouží položka „*Diagrams* > *Open Activity Detail Diagram*“ v kontextovém menu nad aktivitou. Diagram je

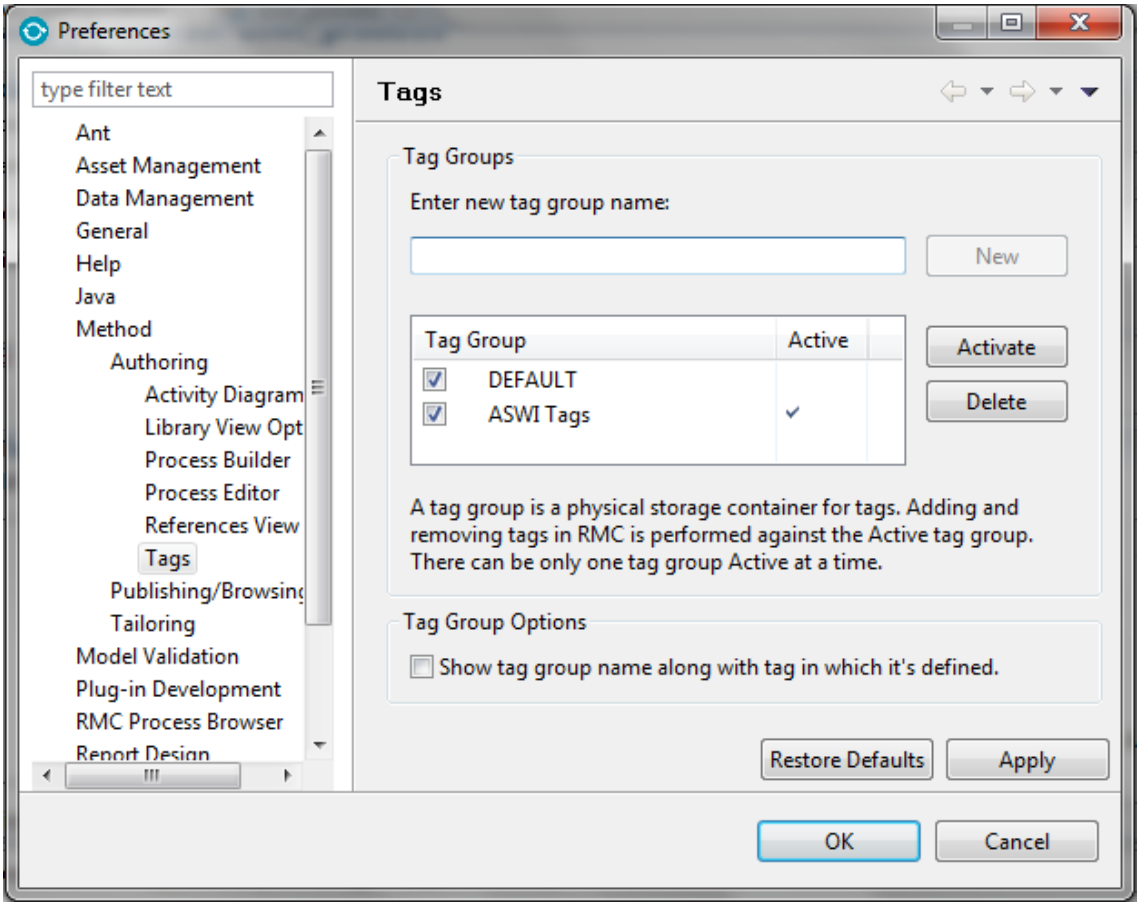
automaticky generován zároveň s přidáváním deskriptorů úkolů a jeho úprava tak není nutná, a je také silně nedoporučovaná novým uživatelům RMC.

Třetím typem diagramu je **digram závislosti výsledků práce**, který zobrazuje vzájemné vztahy artefaktů v rámci části procesu, k níž náleží. Jeho editaci lze zpřístupnit volbou „*Diagrams* > *Open Work Product Dependency Diagram*“ z kontextového menu nad vybraným uzlem struktury procesu (s výjimkou deskriptorů úkolů).

5.1.8 Značky a jejich správa

Již několikrát se v tomto textu objevil pojem značky (*Tags*) v kontextu popisů prvků procesu a jejich kategorizace. Značky jsou vlastně klíčovými slovy, která mohou být připojena k téměř jakémukoli elementu knihovny. V tomto oddíle je popsána manipulace s množinou těchto značek a jejich využití. Samotná nápověda RMC definuje značky následovně.

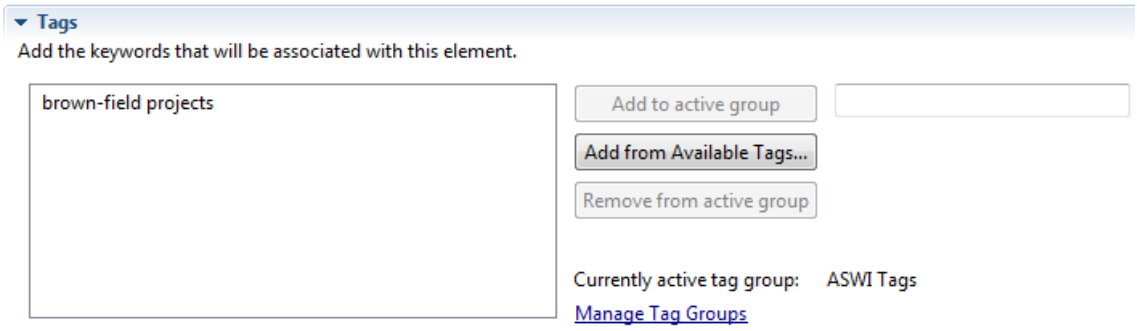
„Tags are keywords that can be assigned to elements within a method library. You can then search, retrieve, and filter information based on tags.“[17]



Obrázek 5.9 – Okno pro správu skupin značek

Po otevření jakéhokoli prvku popisu procesu pro editaci se na kartě **Popis** nachází sekce „*Tags*“ s odkazem „*Manage Tag Groups*“ pro správu skupin značek, určení jejich množiny, která bude

užívána v procesu. Předdefinována je prázdná skupina *DEFAULT*, nicméně doporučuje se založit skupinu vlastní a tu také označit za tzv. aktivní, což znamená, že přidávání nových značek a jejich mazání bude realizování právě nad touto skupinou.Obrázek *5.9* ukazuje dialogové okno pro správu skupin značek.



Obrázek 5.10– Sekce "Tags" karty Popis

Na samotné kartě **Popis**je pak možné přidávat jednotlivé značky k editovanému elementu knihovny, vytvářet nové značky a přidávat je do skupiny. Sekci „*Tags*“ s jejími ovládacími prvky prezentuje obrázek *5.10*.

Pozn.: Při přidávání značek k elementu pomocí tlačítka „*Add from Available Tags*“ je nutné v textovém poli nejprve zadat řetězec „**“ pro zobrazení všech dostupných značek.

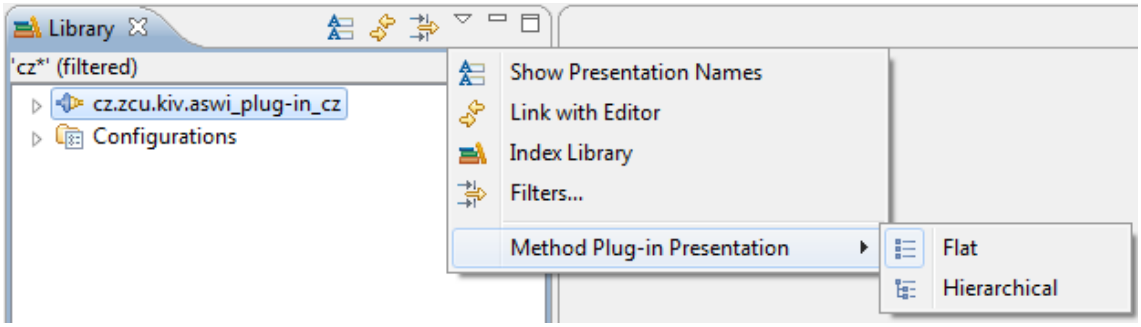
Značky je možné využít k automatickému seskupování elementů do vlastních kategorií pomocí dotazů (viz *5.1.5–Vlastní kategorie*). Dalšími příklady využití značek je jejich zapojení při hledání v knihovně nebo filtrování jejího obsahu a další pokročilé techniky práce s RMC (např. při využívání nástroje *Process Builder*), jež nejsou obsahem této práce a bližší informace o nich je možné najít v [17].

5.1.9 Užitečné tipy

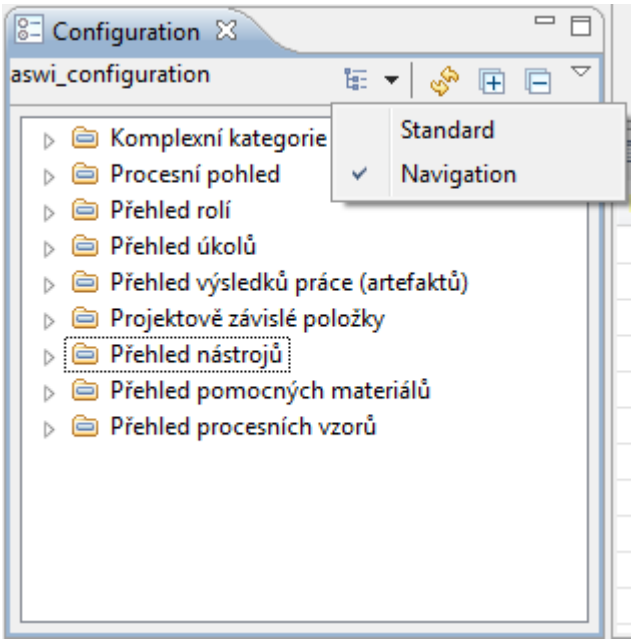
V tomto oddíle jsou uvedeny některé další tipy a rady pro práci s RMC, které mohou usnadnit práci novým uživatelům tohoto nástroje.

Využití filtrů

V zobrazení struktury knihovny (v pohledu *Library View*) i v mnoha dialogových oknech při vytváření relací mezi jednotlivými elementy popisu procesu existuje možnost filtrovat nabízený obsah standardním užitím masek se zástupnými znaky „*“ a „?“. Filtry fungují podle toho, zda je momentálně nastaveno zobrazování identifikačních nebo prezentačních názvů pomocí volby v pohledu *Library View* (viz následující sekce*Možnosti zobrazení struktury knihovny v pohledu Library View*).



Obrázek 5.11– Užitečné volby zobrazení pohledu Library View



Obrázek 5.12– Pohled Configuration View

Možnosti zobrazení struktury knihovny v pohledu Library View

Zobrazení struktury knihovny v pohledu *Library View* má několik prostředků pro její zpřehlednění. Ty jsou demonstrovány na obrázku *5.11* a jejich význam od shora dolů je následující:

- Přepínání mezi **zobrazováním identifikačních** („*Name*“) **a prezentačních názvů**(„*Presentation name*“) elementů knihovny (má vliv i na fungování filtrů – viz předchozí sekce *Využití filtrů*).
- **Propojení s editorem** – stisknutí nastaví pohled *Library View* tak, aby ukazoval pozici elementuaktuálně otevřenéhopro editacive struktuře knihovny.
- **Filtry** (viz předchozí sekce *Využití filtrů*)
- **Zobrazení plug-inů**

- ***Ploché*** – zobrazuje každý plug-in na samostatném řádku včetně jeho zařazení v hierarchii balíků připojené pomocí tečkové notace k jeho názvu.
- ***Hierarchické*** – zobrazuje kompletní hierarchii balíků plug-inů ve stromové struktuře.

Náhled konfigurace v pohledu Configuration View

V pohledu *Configuration View* (viz obrázky v příloze C[12] a obrázek *5.12*) lze nastavit typ zobrazení konfigurace označený jako „*Navigation*“. Pohled pak poskytuje náhled struktury konfigurace tak, jak bude publikována (viz *5.2.1*). Uzly nejvyšší úrovně v hierarchii pak představují jednotlivé pohledy konfigurace (viz *5.1.6–Karta Pohledy*).

Stručný popis položek v seznamech

Pod seznamy spřízněných elementů na kartách editace popisu jednotlivých prvků (např. seznamy kategorií na kartě **Kategorie** u většiny elementů – viz *5.1.3*) se většinou nachází neupravitelné pole „*Brief description*“, které zobrazuje obsah stejnojmenného pole na kartě **Popis** (viz*5.1.3*) označeného elementu v seznamu. To může uživateli pomoci identifikovat konkrétní prvek tehdy, pokud k tomu nestačí jeho název.

Poznámka o autorských právech

K účelu publikace konfigurace lze vytvořit speciální pomocný materiál (*Guidance*; viz*3.4.3*) typu **Podpůrný materiál** (*Supporting material*; viz tabulka *3.1*), který vloží do zápatí všech stránek HTML nebo PDF dokumentu publikované konfigurace klasickou poznámku o autorských právech.

Stačí do pole „*Main description*“ na kartě **Popis** (viz *5.1.3*)vytvořeného prvku typu **Podpůrný materiál** vložit požadovaný text poznámky. Tento podpůrný materiál je poté třeba připojit k plug-inu do pole „*Copyright*“ na kartě **Popis**.

Glosář

Pokud v plug-inu uživatel vytvoří pomocné materiály (*Guidance*; viz *3.4.3*) typu **Definice pojmu** (*Term definition*; viz tabulka *3.1*), je z nich možné automaticky generovat glosář publikované formy konfigurace. Stačí využít patřičnou volbu na jedné z karet editace konfigurace (viz *5.1.6*) nebo při samotném průběhu operace publikování (viz *5.2.1*).

Kontrola pravopisu

Nástroj RMC disponuje i funkcí kontroly pravopisu, ale protože knihovna ani RMC nejsou vybaveny lokalizací pro Českou republiku, je tato funkce využitelná pouze pro popisy v anglickém jazyce. Funkce je přístupná pod volbou „*Spell Check…*“ v kontextových menu.

5.2 Generování výstupních dokumentů z RMC konfigurace procesu

RMC má několik výstupních formátů, které lze primárně rozdělit do dvou skupin: publikace konfigurace a exporty. Nejvýznamnějšími zástupci obou skupin a postupem jejich provedení se zabývá tento oddíl.

5.2.1 Publikování konfigurace

Publikovaná forma konfigurace s popisem jednoho či více procesů je hlavním výstupem celého nástroje RMC a její generování je jedním z předních účelů této diplomové práce. Proces publikace konfigurace lze zahájit zvolením položky „*Configuration* > *Publish*...“ z hlavního menu uživatelského rozhraní RMC.

Program následně otevře průvodce publikováním konfigurace, resp. sekvenci po sobě následujících dialogových oken, mezi kterými se uživatel naviguje obvyklým způsobem– pomocí tlačítek „*Next* >“ a „< *Back*“.

Výstupem publikování konfigurace mohou být HTML stránky, PDF dokument nebo dokument aplikace *Microsoft® Word®*. U HTML stránek navíc existuje možnost vybrat jednu z nabízených technologií užitých k jejich generování.

Publikování konfigurace v jakémkoli formátu může trvat několik minut, nebo i desítek minut a některé konkrétní volby tohoto procesu jsou zmíněny v oddíle *6.2.5*.

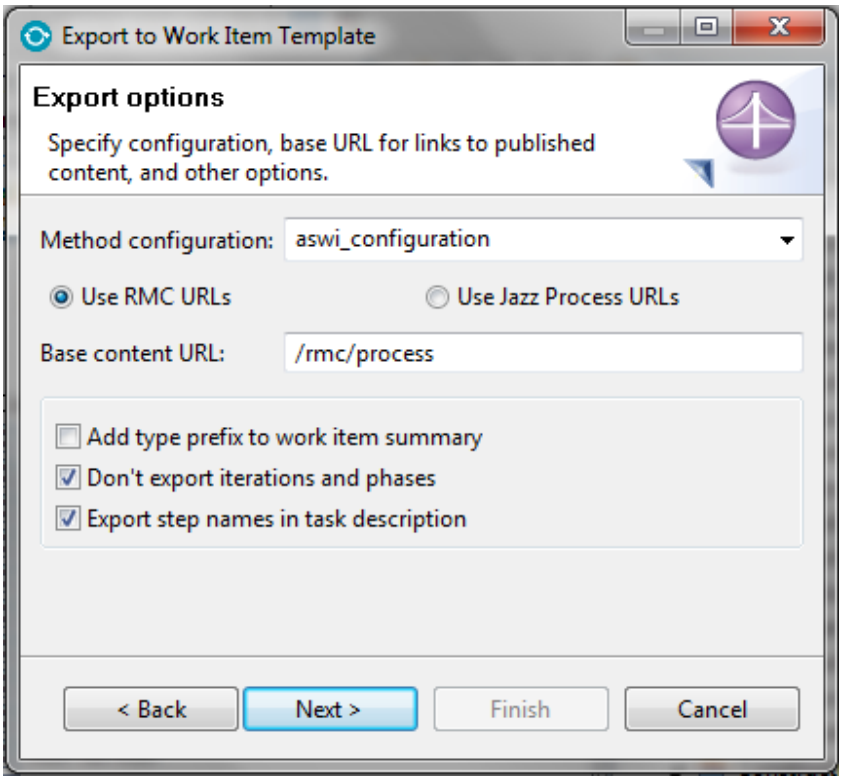
5.2.2 Export šablon pracovních položek

RMC disponuje možností exportu knihovny nebo jejích částí (nikoli publikací konfigurace) hned v několika formátech, jejichž specifika zde nebudou uváděna. Následuje pouze jejich přehled:

- archiv,
- HTML dokument,
- šablona projektu pro *IBM Rational Portfolio Manager*,
- konfigurace knihovny,
- šablona plug-inu,
- skupina plug-inů
- projekt Microsoft,
- skupina praktik,
- možnosti (nastavení),
- sada značek,
- množina týmových projektů,
- projekt pro *WebSphere Business Modeler*,

- XML dokument,
- šablony pracovních položek.

Pro tuto diplomovou práci je nejdůležitější export tzv. *šablon pracovních položek* (*Work Item Templates*). Ty mají formu XML souborů určité struktury a jsou použitelné pro import do systému pro řízení projektu *IBM Rational Team Concert* (viz *4.4* a *5.3*). Tato operace převodu šablon mezi RMC a RTC je hlavním předmětem této diplomové práce.



Obrázek 5.13 – Dialogové okno exportu šablon pracovních položek

Pro export šablon pracovních položek slouží volba „*Export to Work Item Template…*“ z kontextového menu nad některým z prvků struktury procesu na kartě**Struktura rozkladu práce** (*Work Breakdown Structure*; viz *5.1.7*). Následně spuštěný průvodce obsahuje hlavně dialogové okno na obrázku *5.13*, jehož konkrétní nastavení budou popsána v oddíle *6.3.1*.

Pozn.: Exportovány budou jen elementy struktury procesu označené na kartě **Struktura rozkladu práce** (*Work Breakdown Structure*; viz *5.1.7*) zaškrtávacím políčkem ve sloupci „*Planned*“.

5.3 Postup importu šablon pracovních položek do RTC

V oddíle výše byl popsán postup exportu šablon pracovních položek z popisu procesu v RMC. Ty slouží především jako přenosové medium mezi RMC popisem procesu a RTC šablonou jeho procesu, viz následující citát.

„The RMC/RTC integration allows us to modify and publish our process for viewing in a web browser, then create tasks in RTC that match the tasks in the process published from RMC.“[19]

Postup operace spojené s importem šablon pracovních položek RMC do RTC byl předmětem i diplomové práce Radka Kohúta na *Masarykově univerzitě v Brně* vypracované v roce 2011 (viz [18]). Ta konkrétně popisovala model procesu *V-modelu* (viz *2.4.1*) a následný import šablon jeho pracovních položek do RTC verze *2.0.0.2*.

„Exportem šablony procesu V-model z RTC 2.0.0.2, následným importem do RTC 3.0 a odzkoušením je otestováno, že šablona procesu V-model je plně kompatibilní s RTC 3.0 a není tedy nutné provádět žádné dodatečné úpravy.“[18]

Postup importu v diplomové práci z *Masarykovy univerzity* se však prokázal jako nepoužitelný pro verzi *RTC3.0.1.2*, která je v současné době užívána na *Katedře informatiky a výpočetní techniky Fakulty Aplikovaných věd* na ZČU. Tato práce sice zmiňuje kompatibilitu s RTC verze *3* (viz citace výše), ovšem ta se týká pouze přenosu kompletní šablony projektu převedené do této verze RTC z verze *2*, ne samotného postupu připojení šablon pracovních položek do projektu nebo jeho šablony v RTCverze *3*.

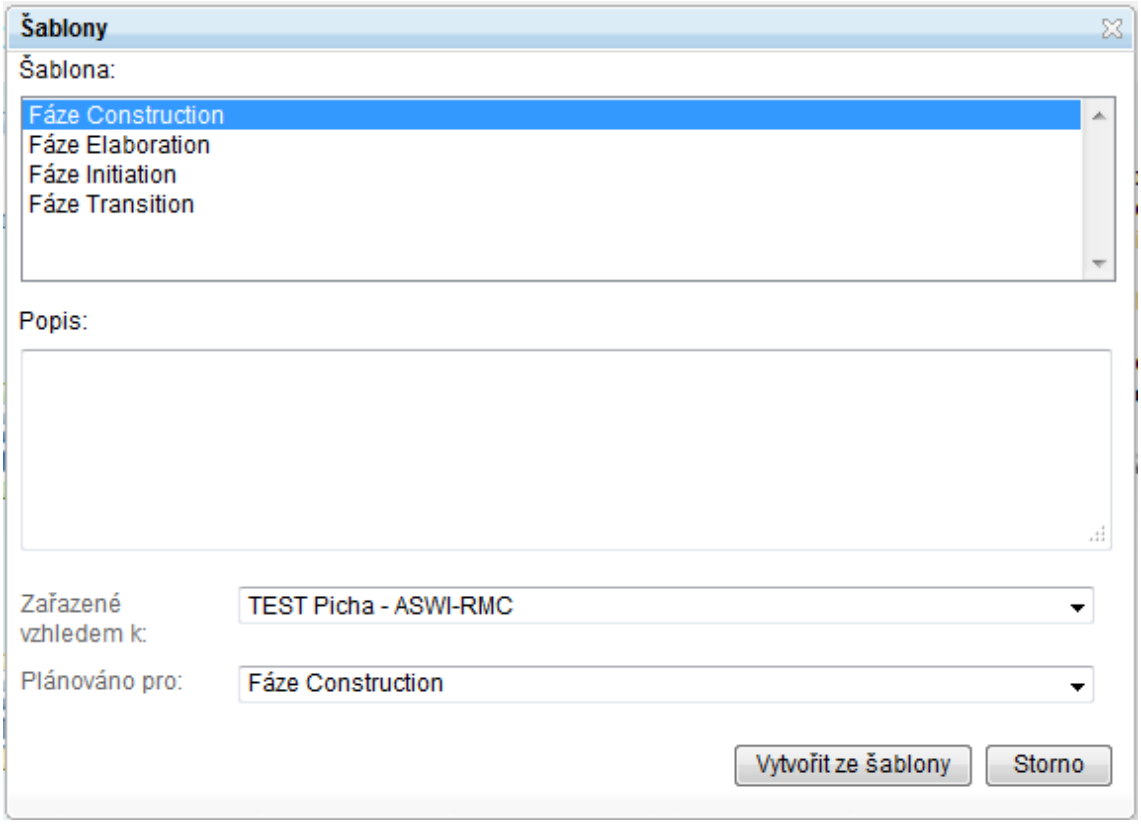
Zde popsaný postup byl prováděn na serveru *RTC 3.0.1.2* a pomocí klientské aplikace RTC verze *3.0.1.5* integrované v IDE *Eclipse*.Postup byl do velké míry převzat z článku [19].

K připojení šablon pracovních položek je nejprve nutné vytvořit novou oblast projektu v RTC a nastavit všechny její požadované aspekty[21] (např. role a jejich oprávnění, možnosti plánování, typy pracovních položek, přístupné priority, atd.). Časovouosu projektu a iteraceje vhodné realizovat tak, aby byly přímo mapovatelné na pracovní položky, jejichž šablony byly exportovány z RMC.

Na kartě **Odkazy** editace oblasti projektu v klientské aplikaci RTC v sekci **Přílohy**je dále nutné připojit XML soubory se šablonami exportovanými z RMC. Ty se následně zobrazí na kartě **Konfigurace procesu** v sekci **Sdílené šablony** po zvolení „*Konfigurace projektu > Konfigurační data > Pracovní položky >Šablony*“v sekci **Konfigurace** (viz obrázek v příloze D[12]).

Dále stačí v uživatelském rozhraní serverové části RTCvybrat příslušnou oblast projektu a v menu položku „*Pracovní položky >Vytvořit ze šablony*“ a připojit šablony k jednotlivým iteracím (resp. částem časové osy projektu).

[21] Nastavení oblasti projektu je nutné k její plné funkčnosti a použitelnosti. Tato nastavení nejsou importovatelná z modelu procesu v *RMC* a bez nich je samotný import šablon pracovních položek bezvýznamný.



Obrázek 5.14– Dialogové okno implementace šablon pracovních položek v RTC

Na obrázku *5.14* je dialogové okno pro realizaci této vazby.

Pro získání šablony celé oblasti projektu je poté nutnév klientské aplikaci RTC využít volbu „*Extrahovat šablonu procesu*“ z kontextového menu oblasti, která uloží šablonu na aktuální RTC server, a následně volbu„*Exportovat*“z kontextového menu nad šablonou samotnou. Tuto šablonu je později možné importovat na jiný server RTC.

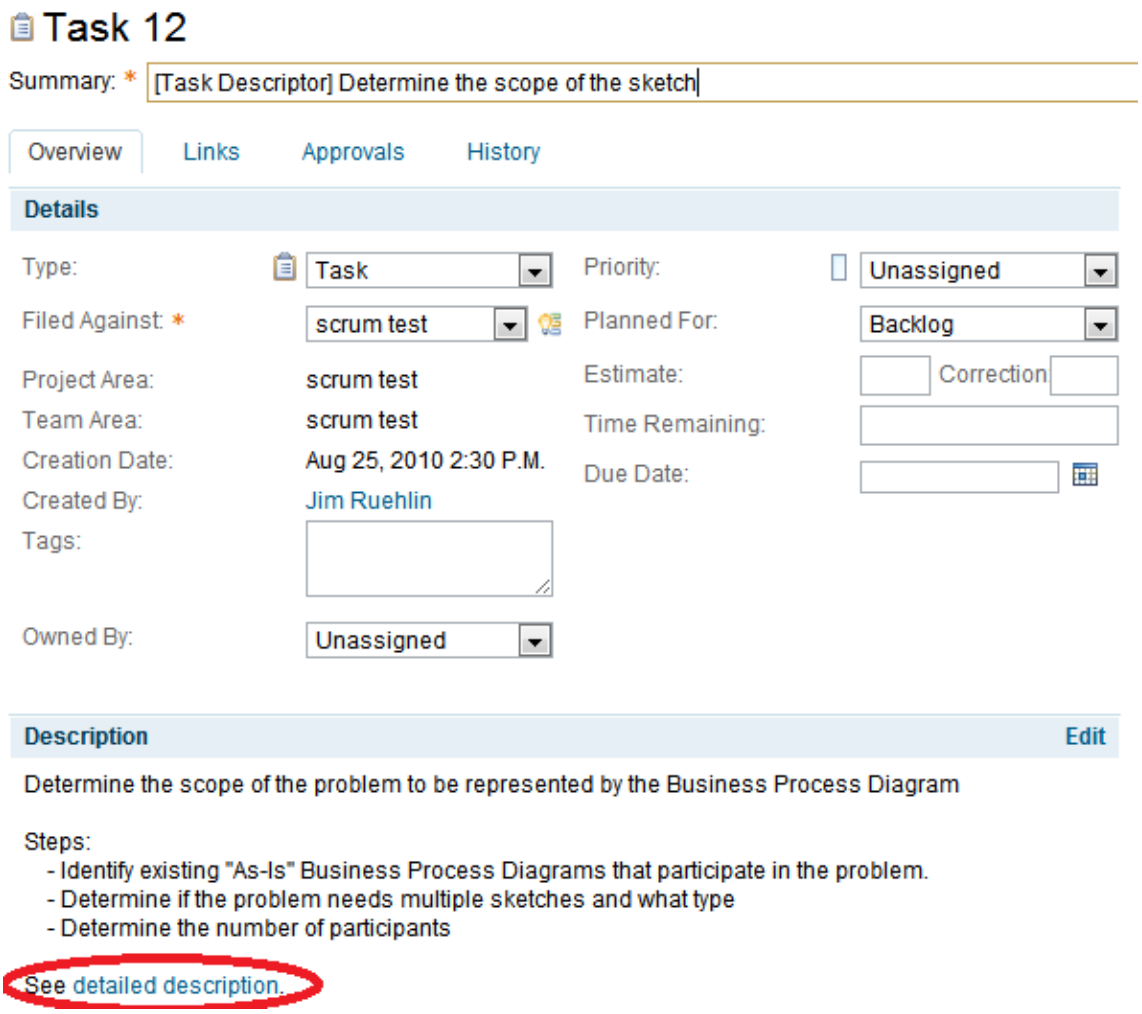
Pozn.: Pokud chcete šablonami pracovních položek rozšířit již existující šablonu celého procesu, vytvořte nejdříve oblast projektu, implementujte na ní šablonu, ke které chcete šablony z RMC přidat, a dále postupujte stejně, jak je popsáno výše.

5.4 Propojení tiketů RTC a HTML dokumentu konfigurace

Konverze mezi RMC a RTC umožňuje dokonce provázat pracovní položky (tikety) v RTC vygenerované z jejich importovaných šablon s HTML dokumentem s publikovanou konfigurací z RMC. Odkazy, které k tomu slouží, se nacházejí na konci popisu pracovních položek při zobrazení jejich detailů a jsou označeny „*See detailed description*“.

Pro aktivaci odkazů stačí pouze publikovat konfiguraci RMC podle postupu popsaného v oddíle *5.2.1* ana posledním dialogovém okně průvodce publikováním zvolitmožnost „*Java EE web application*“. Ta zapříčiní generování aktivních HTML stránek do souboru formátu WAR nebo

EAR. Tento soubor stačí nahrát do složky „*webapps*“ na příslušný serveru *Tomcat*, na němž běží instance RTC serveru (viz [19]).



Obrázek 5.15 – Odkaz na HTML formu konfigurace RMC v pracovní položce RTC[22]

Obrázek *5.15* ukazuje umístění propojovacího odkazu v pracovní položce RTC.

5.5 Nedostatky a možná vylepšení RMC

V tomto oddílu jsou shrnuty drobné nedostatky a návrhy na vylepšení nástroje RMC sebrané v rámci jeho užívání při realizace této diplomové práce.

1) Umožnit **přidání popisu k relacím** mezi jednotlivými prvky popisu procesu (resp. elementy knihovny). Například, proč je u konkrétního úkolu uvedena ta či ona role jako další vykonavatel (*Additional performer*; viz 5.1.5–*Úkol*), nebo proč je daný výsledek práce volitelným vstupem (*Optional input*) určitého úkolu. V současnosti je tyto informace nutné zanést do slovního popisu prvku.

[22] Obrázek převzat z [19].

2) **Přidat** pole **„*Optional output*“** k popisu úkolu (viz *3.5.5*) pro zachycení možných vedlejších produktů činnosti v rámci konkrétního úkolu. To je nyní možné pouze u deskriptorů úkolů.
3) Umožnit využití funkce ***drag and drop* pro přesuny** prvků mezi balíky obsahu (*Content Packages*; viz *3.4.1* a *5.1.5*) knihovny.
4) **Sjednotit terminologii přesunu** položek ze současného „*Move*“ u balíků obsahu a „*Reassign*“ u vlastních kategorií (*Custom Categories*; viz *5.1.5*).
5) Přidat **možnost exkluzivních** vlastních **kategorií** tak, aby bylo například možné jednorázově přidat do kategorie všechny úkoly, které nepatří do jedné nebo více kategorií jiných.
6) Umožnit **vykonání dotazů** u vlastních kategorií (*Custom Categories*; viz *5.1.5*), aby bylo jejich užití možné i **na** již **vytvořenéelementy**, nejen na ty, které jsou založeny až po sestavení dotazu.
7) V kartě **Náhled** jednotlivých elementů zobrazovat i **popisy přejaté** pomocí některé z možností proměnlivosti obsahu (*Content variability*; viz *5.1.3– Provázání obsahu elementů popisu*) ze vzoru elementu. V současnosti v rámci úspory času při jeho generování náhled zobrazuje pouze popisy elementu jemu skutečně vlastní.
8) Zpřístupnit funkci ***drag and drop* pro přesun deskriptorů** úkolu **v** rámci struktury **procesu** i za hranice jejich nadřazeného prvku (aktivity, fáze, iterace, atd.; viz *5.1.7– Skládání procesu*).
9) **Odstranit** nedostatek, díky kterému je při hledání značky, která se má připojit k elementu knihovny, **nutné** zadat řetězec „******“ pro nabídnutí všech dostupných značek (viz *5.1.8*).
10) **Zvýraznit** ve struktuře procesu na kartě **Struktura rozkladu práce** ty **deskriptory** úkolů, které byly **upraveny oproti** svým **vzorům** (tj. úkolům samotným; viz *5.1.7– Skládání procesu*).

6. Proces vývoje software v rámci předmětu KIV/ASWI

Předmět *Pokročilé softwarové inženýrství* (ASWI) je v současné době vyučován na *Katedře informatiky a výpočetní techniky* (KIV) *Fakulty aplikovaných věd* (FAV) na *Západočeské univerzitě v Plzni* (ZČU). V rámci tohoto předmětu studenti nabývají znalostí o procesu vývoje software, jeho metodikách a vzorech tak, jak jsou aplikovány v praxi. V praktické části předmětu tvoří studenti malé vývojové týmy, kterým je na semestr zadán projekt z reálného světa.

Pro tyto projekty je definován proces vývoje a jeho popis v nástroji RMC (viz kapitola 3) je hlavním předmětem praktické části této diplomové práce. Motivací zkoumání možností RMC a následného převodu částí modelu procesu z RMC do RTC (viz *5.2.2* a *5.3*) právě na tomto procesu je fakt, že v projektech v rámci předmětu část týmů používá právě nástroj RTC (viz*4.4*) pro jejich řízení. Dalším důvodem je, že hlavní výstupy této práce, kterými jsou kompletní popis procesu vytvořený v RMC ve formátu HTML (viz *5.2.1*) a šablona pro tyto projekty v RTC s importovanými šablonami pracovních položek, mají usnadnit studentům předmětu uchopení a orientaci v tomto procesu.

Tato kapitola popisuje jak samotný proces ASWI, tak strukturu jeho modelu vytvořeného v RMC a podobu jeho výsledných šablon pro RTC.

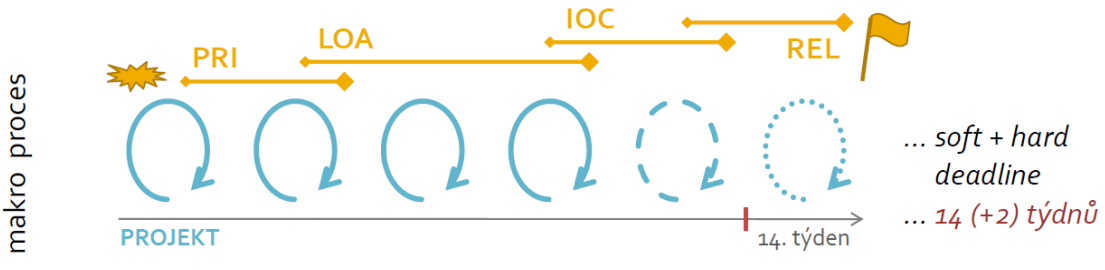
6.1 Struktura procesu

„Softwarový proces pro studentské projekty ASWI je iterativní, agilně orientovaný proces pro řízení tvorby malých až středně velkých softwarových systému, který je založen na metodice Scrum doplněné o některé pedagogicky významné momenty — zejména koncept fází či milníků procesu definovaných Boehmem a použití softwarových nástrojů pro plánování a monitorování procesu.“[20]

V rámci práce na projektech jsou studenti nejprve rozděleni do malých vývojových týmů (obvykle o 2-4 členech), jejichž strukturu a rozdělení pravomocí si určují studenti sami, a následně jsou jim přidělena témata projektů od zadavatelů (obvykle) externích vzhledem ke KIV. Zákazníky jsou nejčastěji např. katedry jiných fakult ZČU, instituce města Plzně nebo některé společnosti v Plzni působící. Od chvíle obdržení zadání mají studenti průběh projektů zcela ve své pravomoci a výsledky prezentují přímo zákazníkovi. Přednášející, resp. cvičící předmětu funguje jen jako dozor (jinak řečeno *mentor*) nad procesní a formální stránkou projektů.

Proces je iterativní a přírůstkový a je rozdělen na čtyři fáze ukončené milníky, jež jsou mírně upravené oproti jejich vzorům z metodiky RUP (viz tabulka *2.1*).

- Fáze **Initiation** – milník **Project Initialized** (PRI),
- Fáze **Elaboration** – milník **Lifecycle Objectives a Architecture** (LOA),
- Fáze **Construction** – milník **Initial Operational Capability** (IOC),
- Fáze **Transition** – milník **Product Release**(REL).



Obrázek 6.1– Struktura makro procesu ASWI[23]

Obsah fází je víceméně stejný jako u metodiky RUP (viz *2.4.2*) s přihlédnutím k rozsahu a časovému rozmezí projektů. Výjimkou jsou první dvě fáze. Strukturu makro procesu ASWI (to je rozdělení na fáze označené zkratkami milníků a iterace v rámci 14týdenního semestru) demonstruje obrátek *6.1*.

Ve fázi **Initiation** je kladen důraz hlavně na zahájení samotných projektů. Fáze je více rozdělena na dvě části (většinou každá o jedné iteraci). V první z nich probíhají hlavně přípravně kroky na samotný projekt jako je ustanovení týmu, přidělení zadání, získání přístupu a porozumění nástrojům pro řízení projektů a končí prvním osobním kontaktem se zákazníkem. Ve druhé je pak kladen důraz na získání základních informací o rozsahu a obsahu projektu a nárocích zákazníka na funkčnost a další parametry finálního produktu, získání a instalaci potřebných prostředků pro vývoj a sestavení hrubého plánu projektu. Finálním výstupem fáze je prvotní a ne příliš detailní verze dokumentu **Vize produktu**.

Fáze **Elaboration** se pak soustředí na zpřesnění požadavků a dalších základních aspektů projektu a tím i dokumentu **Vize**. Dále pak na návrh a ověření možných architektonických řešení, výběr jednoho konkrétního, zdokumentování této architektury prostřednictvím modelů a popisů, případně i implementace základu architektury a její první testování. Hlavním výstupním artefaktem je dokument **Architektura**.

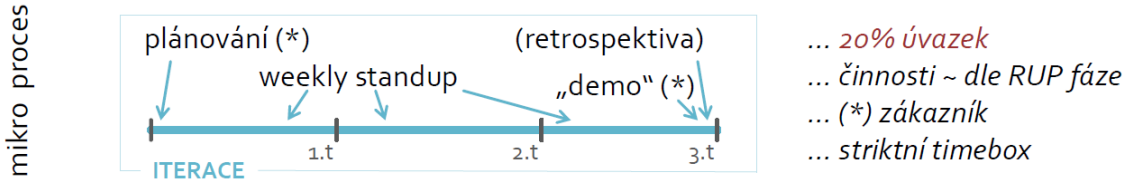
Fáze **Construction** se soustředí na samotnou implementaci aplikace, testování a popř. zahájení práce na dokumentaci produktu.Během fáze by měl tým vystavět tzv. **beta verzi produktu**. Ta obsahuje zhruba 90% funkčnosti dostatečně ověřené testy. Dále pak její částí mohou být i další artefakty související s mimofunkčními požadavky zákazníka na finální produkt.

[23] Obrázek převzat z [20] a mírně upraven.

Ve fázi**Transition** pak týmy dokončují implementaci, testování a dokumentování výsledku své práce a předávají **celý systém** zákazníkovi. Po předání, resp. nasazení vyvíjené aplikace do produkčního prostředí tým provede retrospektivu celého projektu, v níž shrne získané zkušenosti a vyvstalé problémy a jejich řešení v průběhu práce. Poté následuje finální schůzka s mentorem, který následně práci týmu hodnotí na základě informací z revizí iterací i celého projektu a spokojenosti zákazníka.

V rámci fází je proces dále dělen do iterací o rozsahu obvykle 1-3 týdnů. Každá iterace končí schůzkou se zákazníkem. Na té týmy prezentují dosažené výsledky a domlouvají se zástupci zákazníka postup dalšího průběhu projektu.

Na konci každé iterace jsou týmy také vedeny k provedení tzv. retrospektivy, neboli schůzky týmu, kde jsou zhodnoceny silné a slabé stránky postupu v uběhlé iteraci za účelem zvýšení efektivity v dalším vývoji. Navíc na konci první, poslední a každé sudé iterace se tým schází s *mentorem* (akademickým pracovníkem), kterého informuje o dosavadním průběhu projektu.



Obrázek 6.2 – Struktura mikro procesu ASWI

Struktura iterace na třítýdenním příkladu bez revize jejího průběhu s *mentorem* (neboli mikroproces ASWI) je na obrázku *6.2*[23].

Další interní schůzkou v rámci týmu je tzv. *weekly standup* realizovaná každý týden. Hlavním účelem je synchronizace práce a povědomí o stavu projektu mezi jednotlivými členy týmu.

Dalším aspektem ASWI procesu je zapojení několika softwarových nástrojů pro řízení a dokumentování pokroku projektů v jejich průběhu. Jedná se především o systémy pro správu změna verzí (SCM nástroje; viz *4.1*), konkrétně o jednu z možností RTC (viz *4.4*) nebo *Redmine* + *Subversion* (viz *4.2*). Dalším pomocným nástrojem je systém *StudentWiki @KIV* s *wiki* stránkami jednotlivých projektů, kam studenti zaznamenávají průběh jednotlivých iterací a další data o projektech.

Celý proces se soustředí hlavně na:

- **identifikaci** a eliminaci **rizik** v raných fázích projektu,
- brzké **stanovení architektonického řešení** a implementace a ověření jejího základ,
- rychlou a pružnou **reakci na změny** v požadavcích a dalších aspektech projektu,

- průběžné **testování**,
- častou **interakci** se zákazníkem,
- **modelování** architektonických aspektů projektů,
- **získávání zkušeností** z dosavadního vývoje **a** jejich **okamžitou aplikaci** na budoucí průběh projektu.

Další informace o procesu jsou uvedeny v článku, jehož autorem je současný garant a přednášející předmětu ASWIDoc. Ing. Přemysl Brada, MSc., Ph.D. vydaném v roce 2010 (viz [20]). Nicméně od doby publikace článku doznal proces jisté změny, a proto ne všechny informace v něm uvedené jsou aktuální a relevantní.

6.1.1 Vazby procesu ASWI na definované metodiky

Jak již vyplývá z předcházejícího textu, je proces ASWI postaven na základu některých iterativních a agilních metodik a následně upraven pro své konkrétní potřeby a parametry. Konkrétně se jedná především o metodiky *Rational Unified Process* (viz*2.4.2*) a *Scrum* (viz *2.4.3*). Vazby ASWI procesu vývoje software na definované metodiky jsou popsány v tomto oddíle.

Z metodiky RUP přebírá ASWI proces **iterativní charakter**, rozdělení na **čtyři fáze** ukončené milníky a zhruba i jejich obsah. Dále jsou převzaty principy brzké **stabilizace architektury**, artefakty **Vize produktu**, dokument **Architektura** a **beta verze** produktu.

Z metodiky Scrumje převzat **agilní charakter** celého procesu. Konkrétními přebranými praktikami jsou *Daily Scrum* převedený do formy **weekly standupů**, **retrospektivy**, flexibilní **reakce na změny**, průběžná **úprava postupu** v závislosti na dosavadních zkušenostech, princip **interního rozdělení rolí** v rámci vývojového týmu, role *Scrum Mastera* formou **mentora** a víceméně i **backlog**, který je veden v nástrojích pro řízení změn.

Dalšími praktikami, které jsou v procesu ASWI často aplikovány, jsou například ***refactoring*** (úprava formy artefaktů bez změny jejich obsahu), ***timeboxing*** (striktní časové vymezení konkrétní činnosti) a ***vývoj řízený funkčností*** (*use case driven development*). V určitém podílu projektů jsou pak užívány i některé koncepty a konkrétní formy příslušných artefaktů jako ***model a scénáře případů užití*** (*use case model*, resp. *scenarios*), ***user stories*** (krátké a výstižné popisy funkčních požadavků), ***proof-of-concept*** (praktický důkaz kandidátního architektonického řešení) nebo ***Dokument specifikace požadavků***.

V současném akademickém roce (2012/2013) byl proces upraven, respektive byl část původní fáze *Inception* přesunut do fáze *Elaboration*, tak aby přechod studentů (kteří se ve většině případů setkávají s iterativním týmovým vývojem podle definovaného procesu poprvé) do

samotného vývoje byl poněkud hladší a snazší. Konkrétní změna fází oproti těm z metodiky RUP je popsána již v předchozím textu. Inspirace pro tuto změnu vzešla z metodiky *Disciplined Agile Delivery* (viz *2.4.3*).

6.2 Implementace procesu KIV/ASWI v RMC

Následující oddíl popisuje konkrétní model procesu ASWI a jeho částí v nástroji RMC (viz kapitola 3). Tento popis procesu je jedním z hlavních výstupů této diplomové práce.

Všechny části popisu jsou centralizovány v plug-inu (viz *3.4.1*)*ASWI plug-in* (*cz.zcu.kiv.aswi_plug-in_cz*) a konfiguraci (viz *3.4.5*)*Konfigurace pro předmět KIV/ASWI*. O tyto prvky je rozšířena standardní knihovna *RMClib.7.5.rup* a jejich vývoj probíhal ve verzi *RMC 7.5.1* integrovaném v IDE *Eclipse*. Texty popisů jsou v češtině z důvodu primárně výukového účelu modelu procesu. Zároveň identifikační názvy („*Name*“; viz *5.1.3–Karta Popis*) všech vytvořených elementů začínají prefixem „*aswi_*“ pro vyloučení možnosti konfliktů s knihovními prvky a jejich snadnou filtraci a vyhledávání.

Jako pomocné materiály (*Guidance*; viz *3.4.3*) v popisu procesu byly použity převážně zdroje a dokumenty ze stránky předmětu KIV/ASWI na portále *Courseware ZČU* (viz [21]).

*Pozn.:*V tomto oddíle nejsou podrobně rozepsány všechny elementy *ASWI plug-inu* vzhledem k jejich velkému počtu. Tabulky všech elementů se základními údaji o nich lze nalézt v přílohách F – K. Elementy jsou v textu zmiňovány formou svého prezentačního jména („*Presentation name*“; viz *5.1.3–Karta Popis*) a jejich identifikátory („*Name*“) jsou za nimi v závorkách. Pro detailní informace o popisu viz samotnou knihovnu v RMC nebo publikovanou formu konfigurace.

6.2.1 Obsah metody ASWI plug-inu

V tomto oddíle jsou popsány jednotlivé balíky obsahu a kategorie v rámci *ASWI plug-inu* a jejich význam, obsah a další segmentace.

Balíky obsahu

Struktura *ASWI plug-inu* obsahuje celkem 12 balíků obsahu (viz *3.4.1*).

- **Copyright balík (*aswi_copyright_package*)** – obsahuje pouze poznámku o autorských právech.
- **Balík obecných pomocných materiálů(*aswi_general_guidance_package*)** – obsahuje balíky praktik a konceptů a další pomocné materiály (viz *3.4.3*) náležící k procesu jako celku.
    - **Balík konceptů (*aswi_concepts_package*)** – obsahuje pomocné materiály typu Koncept (viz *3.4.3–Pomocné materiály*) vytvořené pro popis procesu.

  - **Balík praktik (*aswi_practices_package*)** – obsahuje pomocné materiály typu Praktika (viz *3.4.3–Pomocné materiály*) vytvořené pro popis procesu.
- **Balík procesních elementů (*aswi_process_package*)** – obsahuje úkoly, výsledky práce a pomocné materiály (všechny viz *3.4.3*) zařaditelné na konkrétní místo ve struktuře projektu.
  - **Balík fází (*aswi_phases_package*)** – obsahuje úkoly, výsledky práce a pomocné materiály náležící k jedné konkrétní fázi procesu.
    - **Obsah fáze Construction (*aswi_construction_package*)** – obsahuje úkoly, výsledky práce a pomocné materiály náležící fázi *Construction*.
    - **Obsah fáze Elaboration (*aswi_elaboration_package*)** – obsahuje úkoly, výsledky práce a pomocné materiály náležící fázi *Elaboration*.
    - **Obsah fáze Initiation (*aswi_initiation_package*)** – obsahuje úkoly, výsledky práce a pomocné materiály náležící fázi *Initiation*.
    - **Obsah fáze Transition (*aswi_transition_package*)** – obsahuje úkoly, výsledky práce a pomocné materiály náležící fázi *Transition*.
  - **Průběžné elementy (*aswi_ongoing_package*)** – obsahuje úkoly, výsledky práce a pomocné materiály užívanév procesu opakovaně nebo průběžně (obvykle v iteracích).
- **Balík rolí (*aswi_roles_package*)** – obsahuje role (viz *3.4.3*Pomocné materiály) definované pro ASWI proces.

Standardní kategorie

ASWI plug-in obsahuje 3 sady rolí a 3 kategorie nástrojů[24] (oboje viz *3.4.2–Standardní kategorie*).

- **Sady rolí (*Role Sets*)**
  - **Akademický dozor (*aswi_academic_roles*)** – obsahuje pouze roli vyučujících předmětu ASWI v rámci procesu.
  - **Role ve vývojovém týmu (*aswi_team_roles*)** – obsahuje role v rámci vývojového týmu.
  - **Zástupci zákazníka (*aswi_customer_roles*)** – obsahuje role zastávané zástupci zadavatelů projektů.
- **Nástroje (*Tools*)**

[24] Oproti definovanému systému RMC, kde jsou standardní kategorie typu **Nástroje** určeny pro reprezentaci skutečných nástrojů a zapouzdřují sady pomocných materiálů typů **Návod pro nástroj** (*Tool Mentor*), jsou v *ASWI plug-inu* kategorie **Nástroje** užité jako opravdové skupiny nástrojů se stejným účelem a *Tool Mentor* položky představují konkrétní nástroje.

- **Nástroje pro řízení změn (*aswi_change_mgmt_tools*) –** obsahuje popisy nástrojů *Jira*, *Flyspray*, *Redmine* a *RTC*.
- **Publikační nástroje (*aswi_web_page_tools*) –** obsahuje popis nástroje *StudentWiki @KIV*.
- **Verzovací nástroje (*aswi_config_tools*) –** obsahuje popis nástrojů *RTC* a *Subversion*.

Vlastní kategorie

V *ASWI plug-inu* je definováno celkem 40 vlastních kategorií (viz *3.4.2*). Jejich kompletní přehled je v tabulce v příloze J. Zde jsou uvedeny jen ty v nejvyšších úrovních jejich hierarchie.

- **Komplexní kategorie (*aswi_complex_category*)** – obsahuje všechny další kategorie.
  - **Procesní pohled (*aswi_process_category*)** – obsahuje samotný proces a základní pomocný dokument k němu.
  - **Přehled rolí (*aswi_roles_category*)** – obsahuje standardní kategorie rolí a souhrnnou kategorii všech rolí.
  - **Přehled úkolů (*aswi_tasks_category*)**–obsahuje kategorie pro rozřazení úkolů podle důležitosti a disciplín, kategorii obsahující všechny úkoly představující schůzky a souhrnnou kategorii všech úkolů.
  - **Přehled výsledků práce (*aswi_work_products_category*)** – obsahuje kategorie pro rozřazení výsledků práce podle důležitosti, disciplíny, druhu a úrovně formality a souhrnnou kategorii všech výsledků práce.
  - **Projektově závislé položky (*aswi_project_dependent_elements_category*)** – obsahuje rozřazení elementů podle specifického typu nebo tématu projektu (např. *brown-field* projekty, projekty s databází, atd.).
  - **Přehled nástrojů (*aswi_tools_category*)** – obsahuje standardní kategorie nástrojů.
  - **Přehled pomocných materiálů (*aswi_guidance_category*)** – obsahuje rozřazení pomocných materiálů na příklady, šablony, návody, praktiky, koncepty a obecné.
  - **Přehled procesních vzorů (*aswi_capability_patterns_category*)** – obsahuje všechny procesní vzory (viz *3.4.4*) v plug-inu a verzi procesu AWSI pro export šablon pracovních položek.

Specifickými případy jsou dvě kategorie s názvem **Project Assesment** (***aswi_project_assesment_discipline***, resp. ***aswi_project_assesment_domain***) v nadřízených kategoriích **Úkoly podle disciplín** (***aswi_tasks_by_discipline***), resp. **Výsledky práce podle disciplín** (***aswi_work_products_by_discipline***). Těm jsou patřičně změněny i ikony tak, aby

z pohledu čtenáře publikovaných dokumentů *Konfigurace pro předmět KIV/ASWI* byly konzistentní se standardními kategoriemi z knihovny (S tím rozdílem, že jejich popis je v českém jazyce).

6.2.2 Procesy

V tomto oddíle jsou zevrubně popsány procesy a procesní vzory v *ASWI plug-inu* (viz *3.4.4*).

Procesní vzory

V *ASWI plug-inu*jsou vytvořeny 3 procesní vzory (*Capability Patterns*). Kompletní struktura vzoru *Iterace* je rozepsána v tabulce v příloze K, části a.

- **Fáze ASWI procesů (*aswi_phases*)** – obsahuje pouze fáze a milníky ASWI procesu.
- **Iterace (*aswi_iteration*)** – Obsahuje strukturu aktivit a deskriptorů úkolů společnou pro iterace všech fází, a to včetně užití dalšího procesního vzoru – **Týden**.
- **Týden (*aswi_week*)** – obsahuje aktivity a úkoly prováděné v ASWI procesu každý týden. Momentálně obsahuje pouze jednu aktivitu „*Provést každotýdenní úkoly*“, v níž je úkol „*Provést weekly standup*“.

Procesy životního cyklu

ASWI plug-in obsahuje 2 celkové procesy životního cyklu (*Delivery Processes*).

- **Exportovaný proces ASWI (*aswi_proces_export*)** – proces se strukturou užívanou pouze pro export šablon pracovních položek. Ten obsahuje pouze nejpodstatnější a projektově nezávislé úkoly (resp. deskriptory), které je možné využít přímo jako pracovní položky po importu šablon do RTC. Kompletní struktura procesu je rozepsána v tabulce v příloze K, části b.
- **Proces vývoje software v ASWI (*aswi_process*)** – komplexní popis procesu ASWI s využitím všech úkolů a výsledků práce obsažených v plug-inu (včetně těch projektově závislých), který slouží jako podpůrný materiál pro studenty předmětu. Kompletní struktura vzoru *Iterace* je rozepsána v tabulce v příloze K, části c.

6.2.3 Užité reference na knihovní elementy RMC

V tomto oddíle jsou zmíněny vazby mezi vytvořenými a knihovními elementy popisu procesů. Malé využití knihovních prvků je důsledkem volby českého jazyka pro popis procesu a faktu, že veškeré elementy ve standardní knihovně jsou popsány v jazyce anglickém.

Standardní kategorie

Popis ASWI procesu využívá standardní kategorie pro seskupování úkolů a výsledků práce z původní knihovny distribuované s RMC. Jedná se tedy konkrétně o množinu disciplín, domén a druhů výsledků práce (viz *3.4.2–Standardní kategorie*). Kategorie **Úkoly podle disciplín**

(***aswi_tasks_by_discipline***) přímo rozšiřuje (typem *Extends* provázání obsahu; viz *5.1.3*) knihovní kategorie **Disciplines** (*disciplines*) z plug_inu *core.base_rup*.

Tabulka 6.1– Elementy původní knihovny připojené k prvkům ASWI plug-inu

Element původní knihovny			Element ASWI plug-inu	
Název	**Typ**	**Součást plug-inu**	**Název**	**Typ**
CSPS Use Case Specification – Elaboration Phase	Příklad	core.informal_resources	Scénáře případů užití	Koncept
CSPS Use Case Specification – Inception Phase				
Software Requirements Specification	Směrnice	core.base_rup	Dokument specifikace požadavků	Výsledek práce
			Vytvořit Dokument specifikace požadavků	Úkol
Use Case Specification (Informal)	Šablona	core.informal_resources	Popis požadavků	Výsledek práce
Use-Case Diagram	Směrnice	core.base_rup	Use case digram	Koncept
Use-Case Model				

Pomocné materiály

K některým elementům *ASWI plug-inu* jsou připojeny odkazy na pomocné materiály z původního obsahu knihovny. Nejčastěji se jedná o popisy možné konkrétní podoby některých artefaktů nebo jejich šablony. Konkrétní přehled těchto pomocných materiálů zachycuje tabulka *6.1*.

6.2.4 Značky definované v ASWI plug-inu

V rámci práce na popisu ASWI procesu byly v nástroji RMC definovány i některé značky (*Tags*). Ty jsou zařazeny do skupiny **ASWI Tags**. Konkrétní značky odpovídají kategoriím projektově závislých elementů popisu. Konkrétně jde o následující značky:

- **brown-field projects** – slouží pro označení specifických úkolů a výsledků práce spojených s *brown-field* projekty (tzn. projekty rozšiřující funkčnost již existujících softwarových aplikací),
- **database project**– slouží pro označení specifických úkolů a výsledků práce spojených s projekty, jejichž částí realizace je i databáze,
- **detached tech support**– slouží pro označení specifických úkolů a výsledků práce spojených s projekty, v jejichž rámci je role technického správce výsledného produktu odloučena od role kontaktní osoby zákazníka,
- **production environment project**– slouží pro označení specifických úkolů a výsledků práce spojených s projekty, jejichž výsledný produkt je určen pro stálý běh v produkčním prostředí (tzn., že se nejedná o pouhou *standalone* aplikaci předávanou zákazníkovi na paměťovém mediu),
- **UI project**– slouží pro označení specifických úkolů a výsledků práce spojených s projekty, jejichž součástí je i uživatelské rozhraní.

6.2.5 Konfigurace pro předmět KIV/ASWI

Předmětem tohoto oddílu je popis konkrétních nastavení *Konfigurace pro předmět KIV/ASWI* užívaných pro její publikování.

Karta Výběr plug-inů a balíků

Na této kartě (*Plug-in and Package Selection*) jsou v poli**Obsah** (*Content*) specifikovány*ASWI plug-in* a kategorie z plug-inu *core.base_concepts* jako jediné hlavní části konfigurace.

Karta Pohledy

Na této kartě (*Views*) jsou specifikovány pohledy konfigurace. Jako tyto pohledy jsou zvoleny **Komplexní kategorie** (označená jako výchozí) a všechny kategorie jim přímo podřízené (viz *6.2.1–Vlastní kategorie*). Podřízené kategorie jsou samozřejmě i součástí **Komplexní kategorie** a tím dochází ke zdvojení jejich zařazení do pohledů. To má za následek v HTML dokumentu generovaném z konfigurace možnost zobrazení buď všech kategorií najednou, nebo zúžený pohled na jednu konkrétní z nich (viz obrázek *5.5*).

Karta Obecné možnosti publikování

Na této kartě (*Publish General Options*) je zvolena možnost „***Publish the entire configuration*“**. To znamená, že do procesu publikace jsou zařazeny všechny části plug-inů a balíků specifikovaných na kartě **Výběr plug-inů a balíků** (viz sekce *Karta Výběr plug-inů a balíků* výše v tomto oddíle). V sekci *Tags* jsou v prvním seznamu označeny skupiny značek *DEFAULT* a *ASWI Tags*. Ve třetím seznamu jsou pak zařazeny pouze *ASWI Tags* jako značky, které se mají zobrazovat v publikovaných dokumentech.

Karta Možnosti publikování HTML

Na této kartě (*Publish HTML Options*) jsou zvoleny následující možnosti a nastavení:

- Je nastavena hodnota v poli „***Title***“, která přestavuje titul publikovaných HTML stránek.
- Hodnota pole „***Feedback URL***“ je sice nastavena, ale na původní hodnotu zadanou samotným RMC.
- V poli „***Skin***“ je nastavena hodnota „***RMC***“, která označuje jednu z výchozích šablon vzhledů HTML stránek distribuovaných spolu s nástrojem.
- **Publish banner** – volba je nastavena, ale bez konkrétní hodnoty, tudíž mázáhlaví stránek podobu definovanou šablonou vzhledu nastavenou v předchozím poli.
- **Publish background** - volba je nastavena, ale bez konkrétní hodnoty, tudíž má pozadí stránek podobu definovanou šablonou vzhledu nastavenou v poli „*Skin*“.
- **Publish aktivity detail diagrams that have not been manuály created** – volba zapříčiní publikování i těch *diagramů detailů aktivit* (viz *5.1.7–Diagramy*), které nebyli ručně vytvořeny (tj. těch automaticky generovaných).
- **Show relationship sub-folders in navigation trees** – volba má za důsledek zobrazení podsložek vztahů v navigačních stromech HTML stránek.
- V poli „***Default tab for aktivity pages***“ je nastavena hodnota „***Work Breakdown Structure***“. To znamená, že při zobrazení stránky procesu nebo nějaké části jeho struktury bude jako výchozí zobrazena její kata **Struktura rozkladu práce**.

Karta Možnosti publikování dokumentu

Na této kartě (*Publish Doc Options*) jsou zvoleny následující možnosti a nastavení:

- V poli „***Report library***“ je nastavena hodnota podle návodu v RMC (viz [16]):

<kořenová_složka_instalace_RMC>\RMC75\rmc\report\Configuration(simple).rptlibrary

- **Publish cover page** – díky volbě je ke generovanému dokumentu připojena titulní strana
- V poli „**Theme**“ je nastavena hodnota „***RMC***“. Význam je analogický k poli „*Skin*“ z předchozí karty (viz předchozí sekce *Karta Možnosti publikování HTML* tohoto oddílu).

6.3 Specifika převodu popisu ASWI procesu mezi RMC a RTC

V tomto oddíle jsou popsány specifické volby použité při exportu a importu šablon pracovních položek mezi RMC a RTC a základní údaje o výstupních souborech. Všechny popsané postupy

byly prováděny na instanci serverové části RTC instalované na hardwarovém vybavení KIVverze *3.0.1.2* a klientské aplikaci *3.0.1.5*.

6.3.1 Export šablon pracovních položek z RMC

Postup operace exportu šablon pracovních položek z nástroje RMC byl prováděn podle postupu popsaného v oddíle *5.2.2*. Exportovány byly šablony pro jednotlivé fáze procesu **Exportovaný proces ASWI** (***aswi_process_export***; viz *6.2.2–Procesy životního cyklu*). Export po fázích byl zvolen kvůli možnosti jejich namapování na části časové osy v RTC a proměnlivému (a tudíž nejistému) počtu iterací v rámci jednotlivých fází napříč konkrétními projekty v předmětu KIV/ASWI.

Pro všechny exporty bylo zvolené jednotné nastavení v dialogovém okně na obrázku *5.13*. Konkrétní nastavení a jejich důvody byly následující:

- **Use RMC URLs** – nastavení této možnosti oproti její alternativě („*Use Jazz Process URLs*“) je doporučeno ve zdrojích [18] a [19].
- **Add prefix type to work item summary** – volba nebyla v nastavení exportů použita, protože výsledné prefixy s označením typu konkrétní položky (např. „*[Task Descriptor]*“) by mohly být v RTC pro studenty matoucí.
- **Don’t export iterations and phases** – volba byla v nastavení exportů použita, aby nedocházelo k vytváření obalujících (nadřazených) pracovních položek v RTC. Cílem převodu do RTC byla pouze množina jednoduchých pracovních položek.
- **Export step names in task description** – volba byla v nastavení použita, aby byly v popisu generovaných pracovních položek zahrnuty i názvy jejich jednotlivých kroků a tím se zvýšila jejich informační hodnota.

Tabulka 6.2– Údaje o exportovaných šablonách pracovních položek procesu ASWI

Exportovaná pracovní položka (část procesu)	Identifikátor (*Template ID*)	Název (*Template display name*)	Název výsledného souboru
Initiation	aswi_initiation	Fáze Initiation	initiation.xml
Elaboration	aswi_ elaboration	Fáze Elaboration	elaboration.xml
Construction	aswi_ construction	Fáze Construction	construction.xml
Transition	aswi_transition	Fáze Transition	transition.xml

Identifikátory, názvy a soubory jednotlivých exportovaných šablon popisuje tabulka *6.2*.

Pozn.: Zároveň byl podle postupu v oddíle *5.4* exportována HTML podoba konfigurace ASWI procesu ve formátu WAR souboru pro provázání pracovních položek RTC a jejich popisu v tomto HTML dokumentu. Tento soubor byl nahrán do složky *webapps* na severu RTC v prostředí KIV.

6.3.2 Import šablon do RTC a vytvoření šablony oblasti projektu

Importem šablon pracovních položek v souborech z tabulky *6.2* byla obohacena šablona oblasti projektu *KIV/ASWI Proces lholy 02.02.11* vytvořená v roce 2011 Ing. Lukášem Holým pro účely zakládání oblastí projektů ASWI v instanci RTC instalované na hardwarovém vybavení KIV.

Pro připojení šablon pracovních položek byla nejprve vytvořena pomocná oblast projektu, která implementovala výše zmíněnou šablonu. Do té byly postupem popsaným v oddíle *5.3* připojeny šablony pracovních položek procesu ASWI z RMC. Následně byla extrahována šablona procesu (viz *5.3*) z této oblasti projektu a v ní byla upravena časová osa tak, aby obsahovala pouze čtyři fáze definované v procesu ASWI.

Finální šablona nese prozatímní název „*ASWI – RMC test*“ a její exportovaná podoba ve formě jak složky souborů, tak archivu se nachází stejně jako XML soubory se šablonami pracovních položek na CD nosiči přiloženém k této diplomové práci.

7. Dosažené výsledky a zhodnocení

Tato kapitola shrnuje a zhodnocuje výsledky, jichž bylo v rámci realizace této diplomové práce dosaženo. Dále pak jsou v kapitole zahrnuty návrhy na údržbu, rozšiřování a další využití vytvořeného RMC popisu procesu ASWI i ostatních znalostí ohledně převodu modelu procesu do nástroje RTC.

7.1 Přínosy a výsledky práce

Tento oddíl popisuje konkrétní výsledky této diplomové práce a znalosti získané v průběhu její realizace.

7.1.1 Teoretické znalosti

Prvním přínosem práce bylo získání širšího a detailnějšího povědomím jejího autora o principech, praktikách a definovaných metodikách procesů užívaných v praxi při vývoji softwarových produktů (viz kapitola 2).

Těchto vědomostí bylo využito k návrhu vylepšení procesu předmětu ASWI, jakož i obsahu jeho přednášek. Tento vedlejší produkt práce byl předán garantovi předmětu a je také připojen k práci samotné na CD nosiči přiloženém k tomuto textu.

7.1.2 Popis procesu KIV/ASWI

Další částí práce bylo prozkoumání možností a funkcí nástroje *IBM®Rational Method Composer®* v rozsahu a míře detailu, daném obsahem a časovým rozpětím této diplomové práce (viz kapitola 3). Byly zjištěny jeho silné stránky, a také jeho mírné nedostatky, které byly popsány v oddíle *5.5*.

Hlavním produktem této části práce je popis procesu vývoje software v rámci předmětu KIV/ASWIv RMC (viz kapitola 6). Jeho beta verze *0.9* publikovaná ve formě HTML dokumentu je již nyní přístupná studentům předmětu KIV/ASWI, kteří byli požádáni o zpětnou vazbu ohledně odhalených chyb a nedostatků.

Studenti vznesli několik námitek a připomínek. Následuje jejich seznam a zdůvodnění, popř. kroky provedené k jejich odstranění.

1. **Přílišná míra provázání jednotlivých prvků** – Tabyla způsobena nastavením konkrétní volby při publikování HTML dokumentu ASWI konfigurace. Ta zapříčinila, že na každé stránce konkrétního úkolu nebo výsledku práce byly v sekci *Product Usage* seznamy všech jeho deskriptorů užitých v procesu. Těch je, ale obvykle mnoho a popis se tak stává nepřehledným a mnohdy rekurzivním. Nastavení bylo upraveno a k tomuto efektu již nedochází.

2. **Přílišná míra detailu popisu** (mnoho úkolů není realizováno v konkrétních projektech) – To je dáno faktem, že vytvořený popis procesu má pokrývat všechny projekty obecně a některé prvky v něm jsou tudíž pro konkrétní projekty volitelné, nepovinné, či dokonce nevhodné[25]. Popis slouží jako ilustrace často vykonávaných činností a často produkovaných artefaktů napříč všemi typy projektů tak, aby bylo i pro některé specifické případy znázorněno jejich umístění ve struktuře procesu a vazby na ostatní elementy jeho popisu. Vytvoření specializovaných verzí popisu procesu pro jednotlivé typy projektů je jedním z návrhů rozšíření výsledků této diplomové práce vznesených v oddíle *7.2*.
3. **Dvojakost jazyka** – Jednotlivá návěští finálních HTML dokumentů daných šablonou jejich vzhledu přímo z RMC (viz *6.2.5–Karta Možnosti publikování HTML*) a popisy užitých knihovních elementů (viz *6.2.3*) jsou v anglickém jazyce. Tento fakt byl autoru i vedoucímu práce dopředu znám, jak je uvedeno v oddíle *6.2*. Nicméně tvorba anglické verze popisu je jedním z návrhů jeho rozšíření v oddíle *7.2*.
4. **Mnoho odkazů, málo textu**– Popis procesu byl vytvářen jako jeho první verze. Proto byl důraz kladen především na jeho strukturu a vazby mezi jednotlivými prvky, a méně na jejich obsáhlé textové popisy. Důsledkem toho je nepoměr zastoupení odkazů a textů na jednotlivých stránkách HTML dokumentu ve prospěch odkazů. Texty se dají (a zřejmě budou) dále rozšiřovat, jak je i navrženo v oddíle *7.2*[26].

Většina dalších připomínek pramenila z nevědomosti studentů o celkovém fungování RMC (co je ovlivnitelné a co ne) nebo souvisí s pokročilými funkcemi jeho užívání (např. forma pro chytré telefony, pořadí jednotlivých sekcí popisu na konkrétních HTML stránkách, apod.). Některé připomínky měly i důvod v chybné interpretaci účelu popisu nebo některých jeho konkrétních aspektů (např. čísla v hranatých závorkách za iteracemi označují jejich počet v rámci fáze průměrný a obvyklý, nikoli povinný nebo závazný).

Celkově se ale studenti ve velké většině případů shodli, že popis je srozumitelný a obsahuje veškeré potřebné prvky odpovídající obsahu předmětu KIV/ASWI a projektům realizovaným v jeho rámci.

Struktura celého popisu byla podrobně konzultována s garantem předmětu p. Bradou, aby co nejvíce vyhovovala jeho představám. Textová část popisů byla zkontrolována Bc. Pavlem Kraftem, absolventem předmětu KIV/ASWI, který celý popis zhodnotil jako dobře realizovaný a potenciálně velmi přínosný pro budoucí studenty předmětu. Finální verze popisu *1.0*, která je

[25] Pokud projekt neobsahuje práci s databází, je zbytečné vytvářet její model nebo skript.
[26] Stejným případem je i návrh jednoho ze studentů na připojení více pomocných materiálů ve formě konkrétních příkladů artefaktů.

obsažena na CD přiloženému k tomuto dokumentu bude v dohledné době prezentována a zpřístupněna studentům předmětu.

7.1.3 Šablony procesu KIV/ASWI a jejich využití v projektech

V následující části práce byla prověřena možnost převodu částí modelu procesu ASWI z RMC do nástroje pro řízení projektů *IBM®Rational Team Concert®*, který je používán v rámci praktické části předmětu KIV/ASWI (viz oddíl *4.4*).

Hlavními výstupy této činnosti v rámci práce jsou šablony pracovních položek RMC a šablona projektů pro RTC (viz *6.3*). Tyto šablony, resp. jejich části, byly pro účely kontroly a zhodnocení poskytnuty jednomu ze současných studentských týmů v předmětu KIV/ASWI. To by mělo přinést zpětnou vazbu ohledně jejich použitelnosti a relevance vzhledem k reálným projektům. Protože ale časový rámec projektů přesahuje termín odevzdání této diplomové práce, není tato zpětná vazba zahrnuta v tomto textu a bude prezentována až během obhajoby práce.

7.1.4 Návody postupů a další vedlejší produkty

Vedlejším produktem realizace práce je kromě návrhů na úpravu obsahu a procesu předmětu KIV/ASWI(zmíněných v oddíle *7.1.1*) rovněž množina návrhů pro udržování a rozšíření samotného popisu ASWI procesu shrnutá v oddíle *7.2*.

Navíc celý text kapitoly 5 o modelování procesu v RMCa postupu importu šablon pracovních položek do RTC a jejich následné zapojení do šablon celých projektů může sloužit jako návod pro nové uživatele RMC a případné zájemce o tématiku převodu RMC modelu procesu do RTC. Obdobný souhrnný návod pro RMC v českém jazyce doposud neexistoval a zdroje v anglickém jazyce se většinou soustředí na velmi úzce specifická témata.

7.1.5 Znalostní báze a její možnosti využití na KIV

V neposlední řadě je dosaženým výsledkem vytvoření znalostní báze pro práci s nástrojem RMC a pro postup převodu modelu procesů z RMC do RTC na *Katedře informatiky a výpočetní techniky*.

V rámci rozšíření a udržení těchto vědomostí na KIV je plánován krátký seminář, během něhož předá autor této diplomové práce základní informace získané v průběhu její realizace několika členům katedry.

Těchto znalostí lze využít v kooperačních projektech KIV s jinými institucemi či soukromými subjekty. Jeden takový projekt konkrétně pro pražskou pobočku společnosti *IBM* je naplánován k zahájení již v dohledné době.

Za zmínku rovněž stojí, že diplomová práce bude prezentována na Studentské vědecké konference pořádané na Fakultě aplikovaných věd ZČU 23. května 2013.

7.2 Udržování a budoucí rozšíření modelu

Tento oddíl shrnuje možná rozšíření popisu procesu ASWI v nástroji RMC vytvořeného v rámci této diplomové práce, jakož i dalších znalostí získaných v průběhu její realizace.

7.2.1 Aktualizace modelu podle procesu

Model v RMC popisuje aktuální podobu ASWI procesu. Při jakékoli změně jeho struktury nebo jiných aspektů, které by vytvořily konflikt mezi modelem a reálným procesem, je nutné tyto změny do modelu zanést.

7.2.2 Rozšíření modelu

I vzhledem k současné podobě procesu ASWI jeho popis v nástroji RMC vytvořený v rámci této diplomové práce není zdaleka tak detailní, jak by mohl vzhledem k obsahu informací v jednotlivých elementech RMC být. V tomto oddíle jsou uvedeny některé návrhy na další rozšíření popisu, resp. *ASWI plug-inu*.

Praktiky

V *ASWI plug-inu* je vytvořeno několik pomocných materiálů typu **Praktika** (*Practice*), které popisují jednotlivé praktiky nebo metodiky, které ASWI proces do sebe zahrnuje a aplikuje. Jejich popis je momentálně velmi stručný, jelikož jejich širší využití bylo po dohodě s vedoucím této diplomové práce vyřazeno z jejího obsahu a může tak být předmětem některého z budoucích projektů na KIV. Je možné jak rozšířit množinu praktik, tak jejich popisy, a také je jich možné využít pro aktivní propojení odkazy mezi HTML formou publikované konfigurace a přednáškami předmětu KIV/ASWI.

Více pomocných materiálů

Do *ASWI plug-inu* by bylo snadno možné přidat více pomocných materiálů, zejména pak příkladů jednotlivých artefaktů objevujících se v procesu.

Anglická verze

Pro účely prezentace modelu např. na mezinárodních konferencích by bylo vhodné vytvořit jeho verzi v anglickém jazyce, např. kopírováním plug-inu a překladem jeho textových popisů. Tato operace by byla sice poměrně jednoduchá, ale vzhledem k rozsáhlosti popisu časově náročná a může tak být využita jako zadání některého z budoucích individuálních projektů studentů na KIV.

Verze procesu pro konkrétní typy projektů

Současný proces, tak jak je modelován v ASWI plug-inu, popisuje souhrnně všechny projekty nezávisle na jejich typu a obsahuje jen nejpodstatnější projektové závislé položky pro demonstraci jejich umístění ve struktuře procesu (např. skript pro databázi, nasazení systému do produkčního prostředí, původní verze systému, prototypy uživatelských prostředí, atd.). V budoucnu by proto bylo vhodné vytvořit verze procesů pro jednotlivé konkrétní typy projektů a jejich šablony pro RTC. Možným prvním krokem by mohlo být vytvoření oddělených procesů pro *green-field* a *brown-field* projekty.

7.2.3 Hlubší využití možností RMC

Kromě prohloubení samotných popisů (viz *7.2.2*), které jsou aktuálně součástí *ASWI plug-inu*, je rovněž možné k budoucím úpravám popisu procesu v RMC využít pokročilejších možností a funkcí tohoto nástroje, než jaké byly aplikovány v rámci této diplomové práce. Některé dosud objevené, ale pro časový i obsahový rámec práce dosud nevyužité, možnosti jsou zmíněny v tomto oddíle.

Glosář

Elegantní a snadno využitelnou funkcí RMC je vytvoření glosáře (viz *5.1.9*), který je následně možné připojit k publikované verzi konfigurace. Práce na tomto rozšíření by de facto zahrnovala pouze definování množiny zásadních pojmů v rámci ASWI procesu a vytvoření odpovídajících elementů typu **Definice pojmu** (*Term Definition*) v RMC.

Pokročilé možnosti formátování textů

Některá z polí karet **Popis** jednotlivých elementů plug-inu jsou vybavena funkcemi pro pokročilé formátování textů. To zahrnuje následující možnosti:

- několik definovaných stylů textu (pro nadpisy, apod.),
- formátování písma (velikost, váha, kurzíva, podtržení, horní a dolní index, atd.),
- číslování a odrážky,
- odsazování textu,
- aktivní odkazy,
- tabulky,
- obrázky, atd.

Rovněž je snadno (pomocí funkce *drag and drop*) možné odkazovat se v popisech na ostatní elementy knihovny.

Diagram závislosti výsledků práce

Kromě dvou typů diagramu aktivit používaných v této diplomové práci je možné též vytvářet diagramy závislostí výsledků práce. Tento fakt zmiňuje již oddíl *5.1.7–Diagramy*.

Karta Odhad

Karta **Odhad** (viz *5.1.5 – Úkol*) u popisu úkolů umožňuje přidat údaje o jejich odhadované době trvání. V budoucnu by bylo možné sbírat a agregovat data o době trvání alespoň několika nejzásadnějších úkolů napříč jednotlivými studentskými projekty a tyto statistické údaje poté zanést do této karty popisu úkolů.

Perspektiva Tailoring

Také využití perspektivy *Tailoring* (viz *3.3*) pro modelování struktury konkrétních projektů na základě popisu procesu je možností, která stojí za zvážení a hlubší průzkum v budoucím vývoji popisu procesu ASWI vytvořeného v rámci této práce. Popřípadě by tato možnost mohla být prozkoumána a využita v jiných projektech na KIV, v nichž by byl používán nástroj RMC.

7.2.4 Automatizace převodu z RMC do RTC

Poslední možností využití vědomostí nabytých v průběhu realizace této diplomové práce by mohly být budoucí projekt zaměřený na automatizaci převodu šablon procesu (resp. pracovních položek) z RMC přímo do nástroje RTC. To může zahrnovat vývoj různých skriptů a dalších prostředků automatizace. Nejvyšší formou nástroje pro zjednodušení převodu šablon mezi oběma nástroji je pak zřejmě plug-in IDE *Eclipse*, do něhož jsou oba produkty integrovány.

8. Závěr

Závěrečná kapitola tohoto textu shrnuje celou diplomovou práci a demonstruje splnění jednotlivých bodů zadání.

Prvním bodem byla studie iterativních metodik vedení softwarových projektů s důrazem na *IBM®Rational Unified Process®* a *Scrum*. To bylo splněno v kapitole 2 tohoto textu, který popisuje kromě jmenovaných metodik i ostatní iterativní a agilní metodiky, a také vývojově starší sekvenční přístupy.

Druhým bodem zadání bylo prozkoumání zástupců pokročilých nástrojů pro popis softwarových procesů a řízení jednotlivých projektů. Zvoleny byly produkty *IBM® Rational Method Composer®* a výše zmíněný *Rational Team Concert*. Zdůvodnění jejich výběru a výsledky jejich zkoumání a tím i splnění tohoto bodu zadání jsou popsány v kapitolách 3, 4 a 5.

V rámci splnění dalšího bodu práce, kterým byla analýza a návrh formy popisu konkrétního procesu, který bude využitelný jako výukový materiál, byl prozkoumán a popsán proces vývoje software užívaný v praktické části předmětu *Pokročilé softwarové inženýrství*. Ten je momentálně vyučován na *Katedře informatiky a výpočetní technikyFakulty aplikovaných vědZápadočeské univerzity v Plzni* a byl tak ideálním kandidátem pro tyto analýzy a popis. K volbě tohoto procesu přispěl i fakt, že jeho garantem je i vedoucí této diplomové práce Doc. Ing. Přemysl Brada MSc., PhD. Výsledky dosažené v tomto bodě práce jsou popsány v první části kapitolách 5 a6.

Dalším bodem zadání bylo vytvoření popisu zvoleného procesu v nástroji RMC a její následné využití pro generování šablony konkrétních projektů v nástroji RTC. Tento popis procesu a postup převodu jeho částí z RMC do RTC je popsán v kapitole 6. Zároveň jsou model procesu, šablony pracovních položek z něj generovaných a konečná šablona projektů z RTC, která šablony pracovních položek využívá, hlavními výstupy této diplomové práce vedle tohoto dokumentu a jsou součástí nosiče CD přiloženého k této práci.

Plnění posledního bodu zadání práce, čili ověření výstupních znalostí a produktů na konkrétních projektech, je momentálně ještě v průběhu z důvodů popsaných v kapitole 7. V této kapitole byly také shrnuty všechny hlavní i vedlejší výstupy a dosažené výsledky této diplomové práce.

Diplomová práce je nyní kompletní a splňuje všechny body zadání v takové míře, v jaké to bylo v době tvorby tohoto textu možné, a tím je připravena k obhajobě. Její realizace byla pro autora nadmíru přínosná a její výstupy a výsledky mají velký potenciál využití jak pro studenty předmětu KIV/ASWI, tak pro *Katedru informatiky a výpočetní techniky*.

Seznam zkratek

ALM – Řízení životního cyklu aplikace (Application Lifecycle Management)

BPMN – Business Process Model and Notation

ASWI – Pokročilé softwarové inženýrství (Advanced Software Engineering)

CCM – Řízení verzí a změn (Configuration and Change Management)

DAD – Disciplined Agile Delivery

EAR – Enterprise Archive

EPF – Eclipse Process Framework

EUP – Enterprise Unified Process

FAV – Fakulta aplikovaných věd

HTML – Hypertextový značkovací jazyk(HyperText Markup Language)

IDE – Integrované vývojové prostředí (Integrated Development Environment)

IOC–Initial Operational Capability

IT – Informační technologie (Information Technology)

KIV – Katedra informatiky a výpočetní techniky

LCA– LifeCycle Architecture

LCO – LifeCycle Objectives

LOA– LifeCycle Objectives and Architecture

OpenUP – Open Unified Process

PDF – Přenositelný formát dokumentů (Portable Document Format)

PR– Pruduct Release

PŘI – Project Initialized

REL – Pruduct Release

RMC – Rational Method Composer

RTC – Rational TeamConcert

RUP – Rational Unified Process

SCM – řízení konfigurací systému (System Configuration Management)

UI – Uživatelské rozhraní (User Interface)

UML – Jednotný modelovací jazyk (Unified Modeling Language)

UP – Unified Process

UPEDU–Unified Process for Education

WAR – Web application Archive

XML – Extensible Markup Language

ZČU – Západočeská univerzita

Literatura

[1] S. W. Ambler a M. Holitza, Agile for Dummies, Hoboken, NJ: John Wiley & Sons, Inc., 2012.

[2] B. W. Boehm, „A Spiral Model of Software Development and Enhancement,“ *ACM SIGSOFT Software Engineering Notes Volume 11 Issue 4*, pp. 14-24, Srpen 1986.

[3] P. Kroll a P. Kruchten, The Rational Unified Process Made Easy: A Practitioner's Guide to the RUP, Stoughton, Massachusetts, USA: Pearson Education, Inc., 2006.

[4] Eclipse Foundation, „Eclipse Process Framework Wiki - OpenUP,“ 2012. [Online]. Dostupné na: http://epf.eclipse.org/wikis/openup/. [Přístup získán 29 Duben 2013].

[5] S. W. Ambler, J. Nalbone a M. J. Vizdos, The Enterprise Unified Process: Extending the Rational Unified Process, Crawfordsville, Indiana: Pearson Education, Inc., 2005.

[6] École Polytechnique de Montréal, „UPEDU - Home,“ 2011. [Online]. Dostupné na: http://www.upedu.org/. [Přístup získán 29 Duben 2013].

[7] Agilní asociace, „Agilní asociace - O nás,“ 2009. [Online]. Dostupné na: http://agilniasociace.cz/. [Přístup získán 30 Duben 2013].

[8] K. Schwaber, Agile Project Management with Scrum, Microsoft Press, 2004.

[9] S. Ambler a M. Lines, „Disciplined Agile Delivery: An agile process decision framework for the enterprise - Intro to DAD,“ [Online]. Dostupné na: http://disciplinedagiledelivery.com/. [Přístup získán 30 Duben 2013].

[10] IBM Corporation, „IBM - Rational Method Composer,“ [Online]. Dostupné na: http://www-03.ibm.com/software/products/cz/cs/rmc/. [Přístup získán 1 Květen 2013].

[11] E. Chromík, „Oborový projekt - Popis procesu ASWI v Rational Method Composer,“ Západočeská univerzita v Plzni, Plzeň, 2011.

[12] rommanasoftware.com, „Rommana: Integrated Application Lifecycle Manager - ALM Tools,“ 2011. [Online]. Dostupné na: http://www.rommanasoftware.com/alm-tools.php. [Přístup získán 6 Květen 2013].

[13] J.-P. Lang, „Redmine - Overview,“ 2013. [Online]. Dostupné na: www.redmine.org.

[Přístup získán 29 Duben 2013].

[14] Atlassian, „Issues & Project Tracking Software | Atlassian Jira - Overview,“ 2013. [Online]. Dostupné na: http://www.atlassian.com/software/jira/overview. [Přístup získán 29 Duben 2013].

[15] IBM Corporation, „Jazz Community Site - Rational Team Concert,“ IBM Corporation, [Online]. Dostupné na: https://jazz.net/products/rational-team-concert/. [Přístup získán 4 Květen 2013].

[16] IBM Corporation, „Rational Team Concert - Help“.

[17] IBM Corporation, „Rational Method Composer - Help,“ 2010.

[18] R. Kohút, „Diplomová práce - Implementace V-modelu v Rational Team Concertu,“ Masarykova univerzita, Fakulta informatiky, Brno, 2011.

[19] J. Ruehlin, „JazzPractices | Process and practices on the IBM Jazz platform - Enacting Processes with Method Composer and Team Concert,“ 25 Srpen 2005. [Online]. Dostupné na: http://jazzpractices.wordpress.com/2010/08/25/enacting-processes-with-method-composer-and-team-concert/. [Přístup získán 29 Duben 2013].

[20] P. Brada, „Agilní proces pro výuku softwarového inženýrství,“ v *Sborník konference Objekty 2010*, Ostrava, 2010.

[21] P. Brada, „Portál ZČU > Courseware > KIV > ASWI,“ Západočeská univerzita v Plzni, 2007. [Online]. Dostupné na: https://portal.zcu.cz/wps/portal/. [Přístup získán 5 Květen 2013].

[22] W. W. Royce, „Managing the Development of Large Software Systems,“ *Proceedings of IEEE WESCON*, pp. 1-9, Srpen 1970.

[23] S. W. Ambler, „Enterprise Unified Process (EUP) - Home,“ Ambysoft Inc., 2012. [Online]. Dostupné na: http://enterpriseunifiedprocess.com/. [Přístup získán 11 Květen 2013].

[24] K. Blittner, „IBM - Driving Iterative Development With Use Cases,“ 25 Květen 2004. [Online]. Dostupné na: http://www.ibm.com/developerworks/rational/library/4029.html. [Přístup získán 11 Květen 2013].

[25] Cycoda Limited, „Cycoda - The Rational Unified Process,“ Cycoda Limited, 2010. [Online]. Dostupné na: http://www.cycoda.com/html/prog-rup.html. [Přístup získán 11 Květen 2013].

[26] ISTQB GUIDE, „ISTQ Exam Certification .com - What is V-model- advantages, disadvantages and when to use it?,“ [Online]. Dostupné na: http://istqbexamcertification.com/what-is-v-model-advantages-disadvantages-and-when-to-use-it/. [Přístup získán 11 Květen 2013].

Přílohy

A. Obsah CD

Zde je rozepsán obsah CD nosiče přiloženého k této diplomové práci.

- **ASWI notes** – obsahuje poznámky a návrhy na úpravu ASWI procesu a obsahu přednášek sesbírané v průběhu realizace práce.
- **Export**
  - ***Work Item Templates*** – obsahuje soubory šablon pracovních položek exportované z RMC.
  - ***RTC*** – obsahuje šablonu ASWI procesu exportovanou z RTC ve formátu složky a archivu.
- **lib7.5rup_aswi_plug-in1.0** – obsahuje knihovnu RMC rozšířenou o kompletní ASWI plug-in a konfiguraci ASWI procesu.
- **Publish**
  - ***HTML*** – obsahuje publikovanou konfiguraci ASWI procesu ve formátu HTML stránek.
  - ***PDF*** – obsahuje publikovanou konfiguraci ASWI procesu ve formátu PDF dokumentu.
- **Text –** obsahuje tento dokument ve formátech PDF a Microsoft Word.
- **READ_ME.txt** – obsahuje tento rozpis obsahu CD nosiče.

B. Obrázky k metodikám procesů

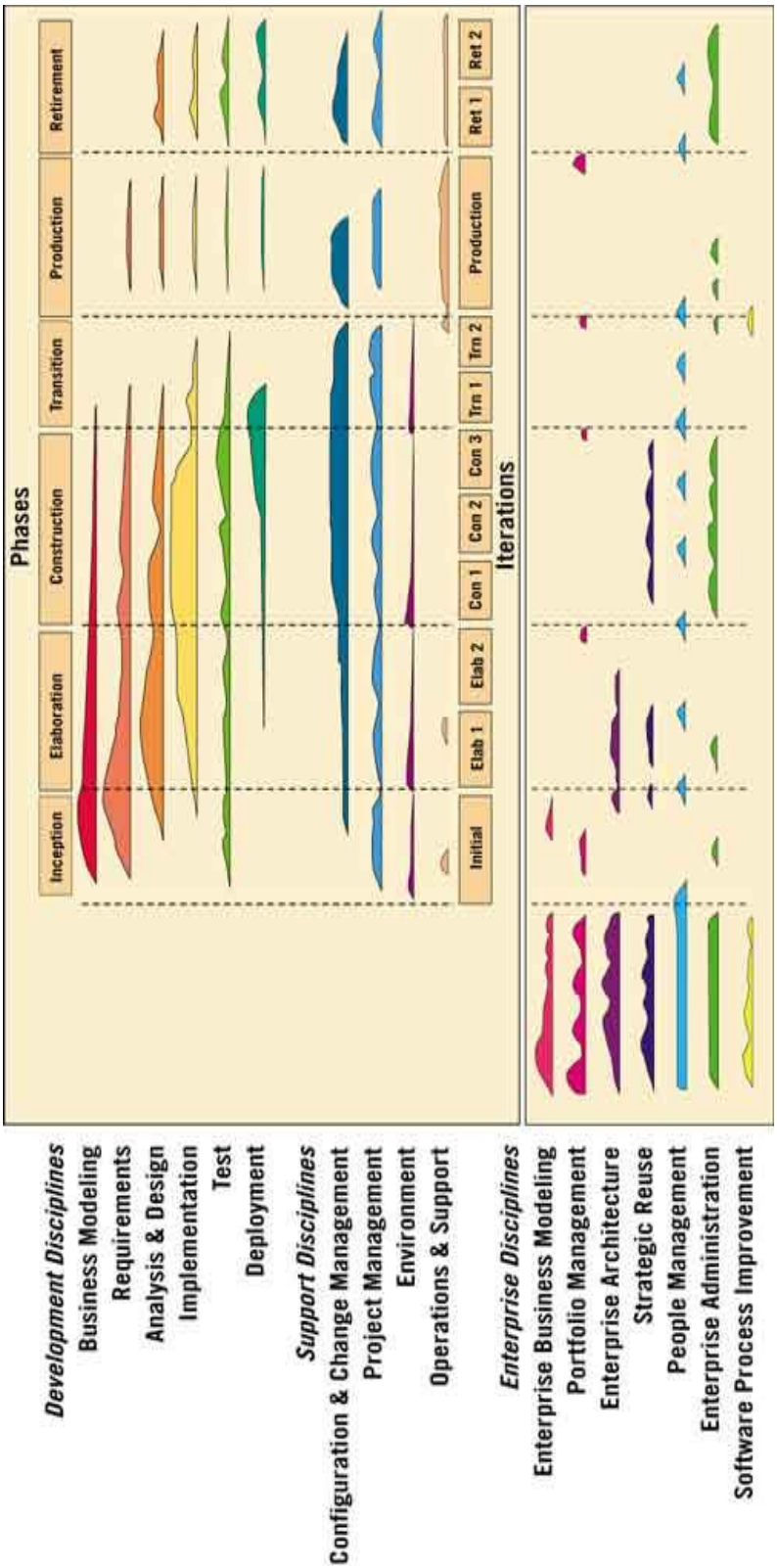
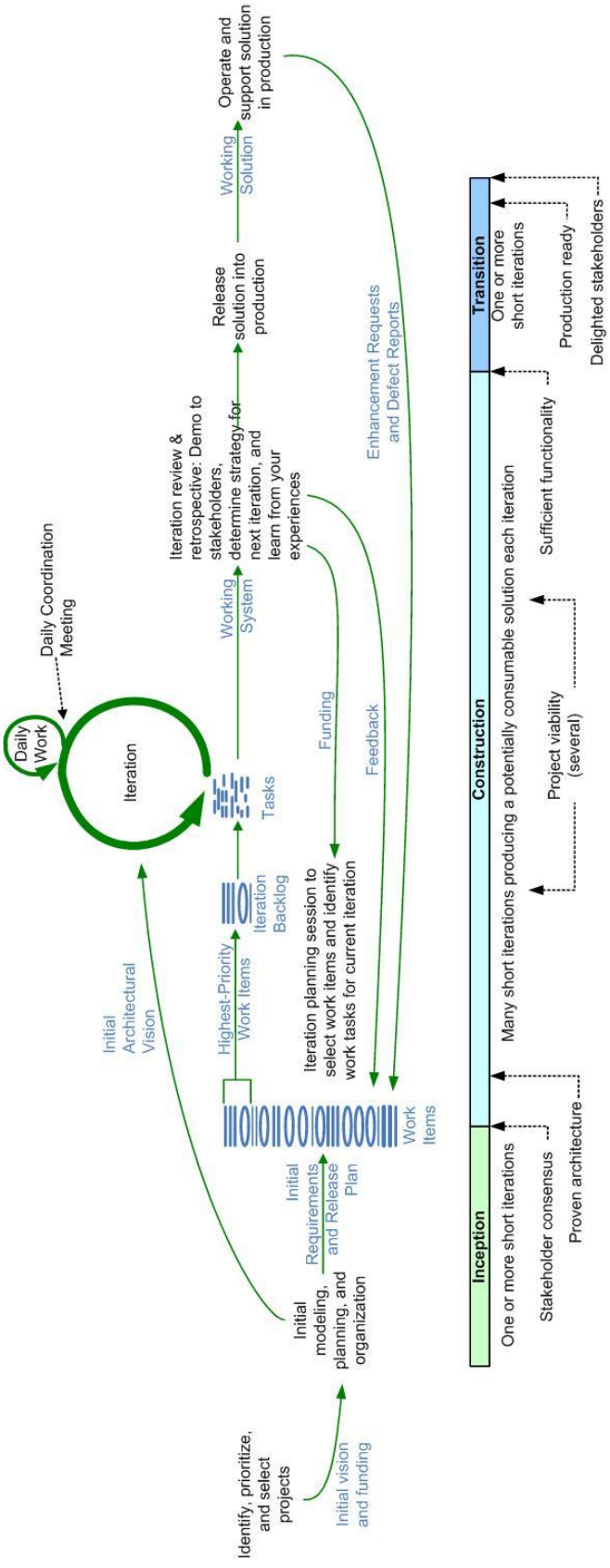


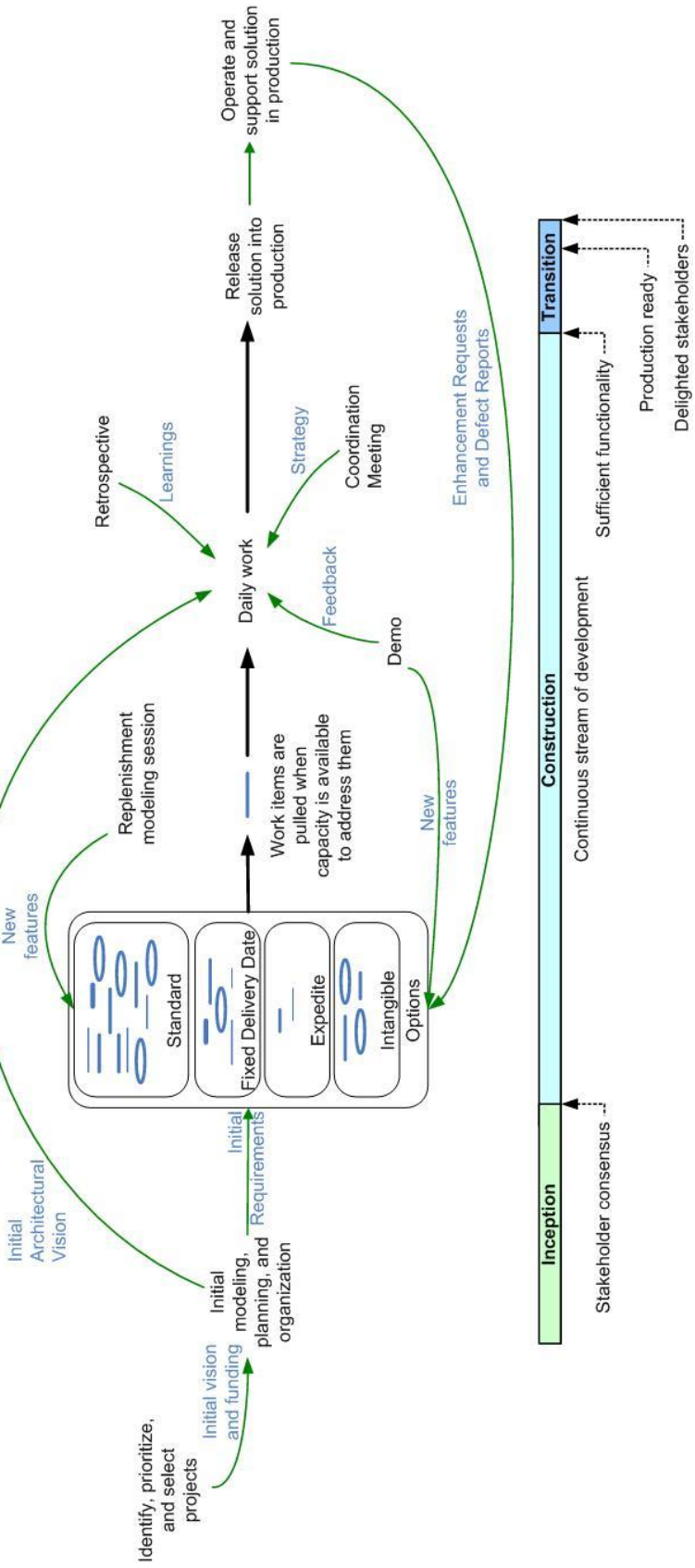
Schéma vývoje software podle EUP[27]

[27] Obrázek převzat z [23].



Základní schéma vývoje software podle DAD[28]

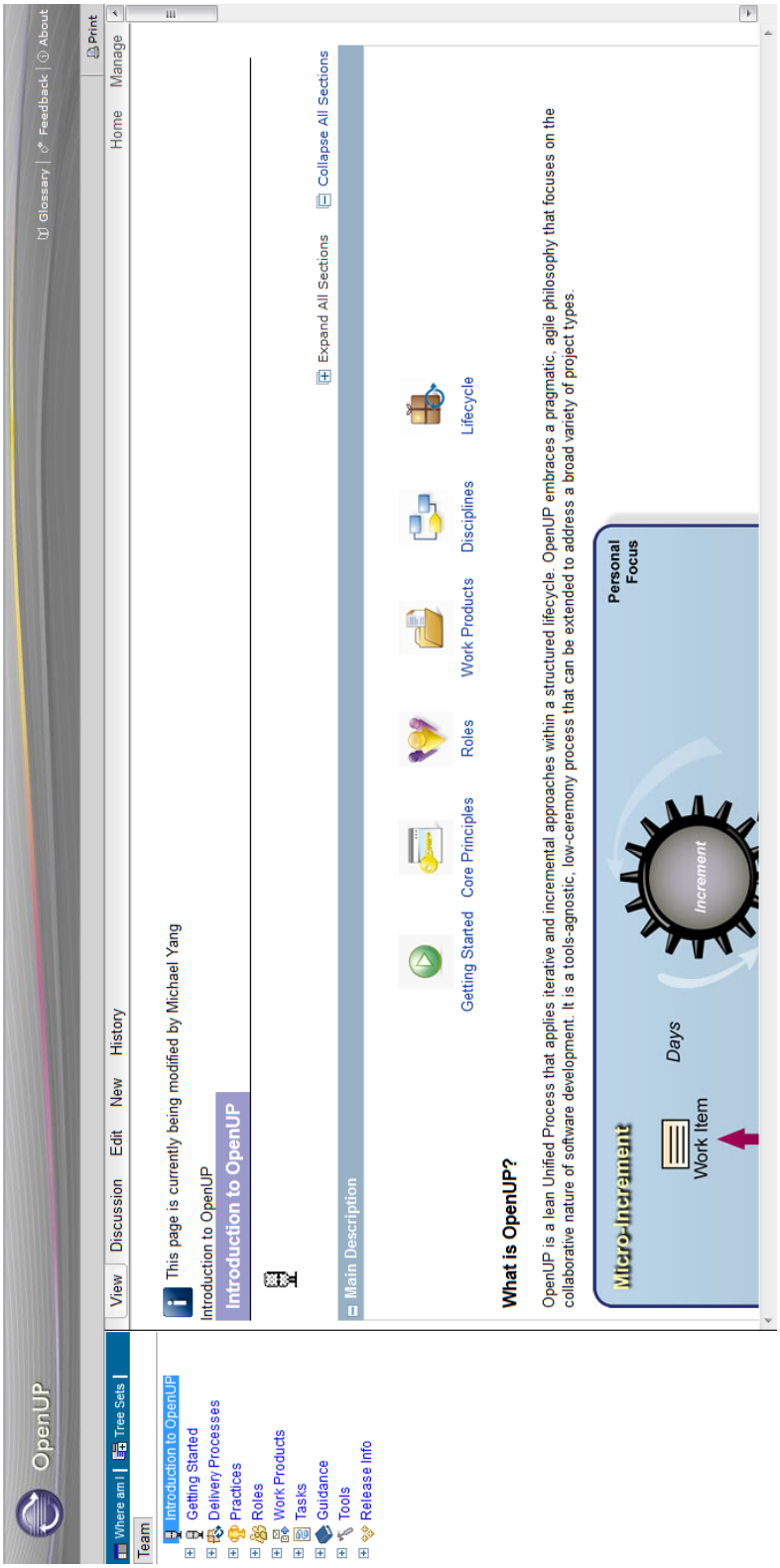
[28] Obrázek převzat z [9].



Pokročilé schéma vývoje software podle DAD[29]

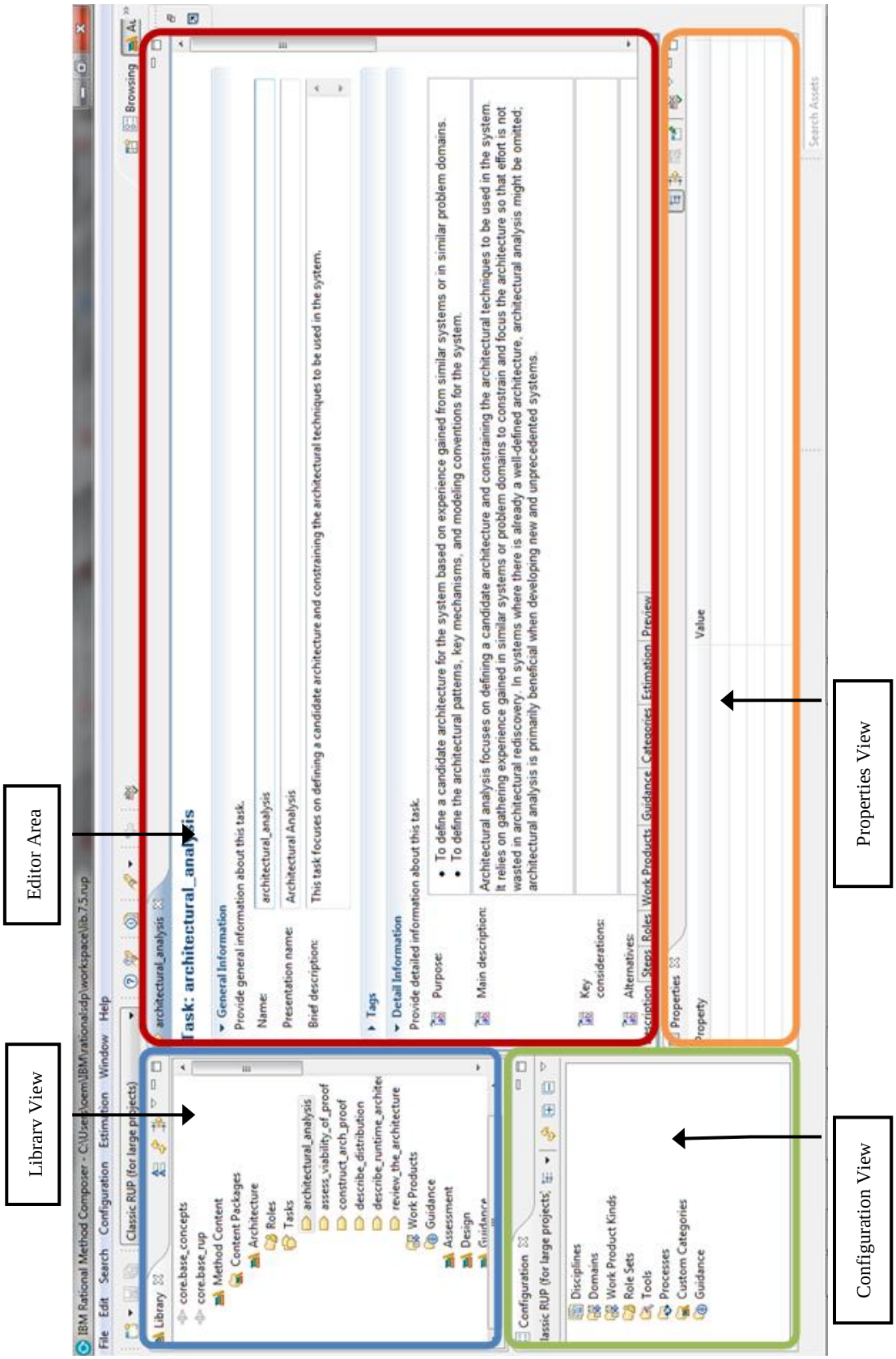
[29] Obrázek převzat z [9].

C. Obrázky k nástrojům pro modelování procesu



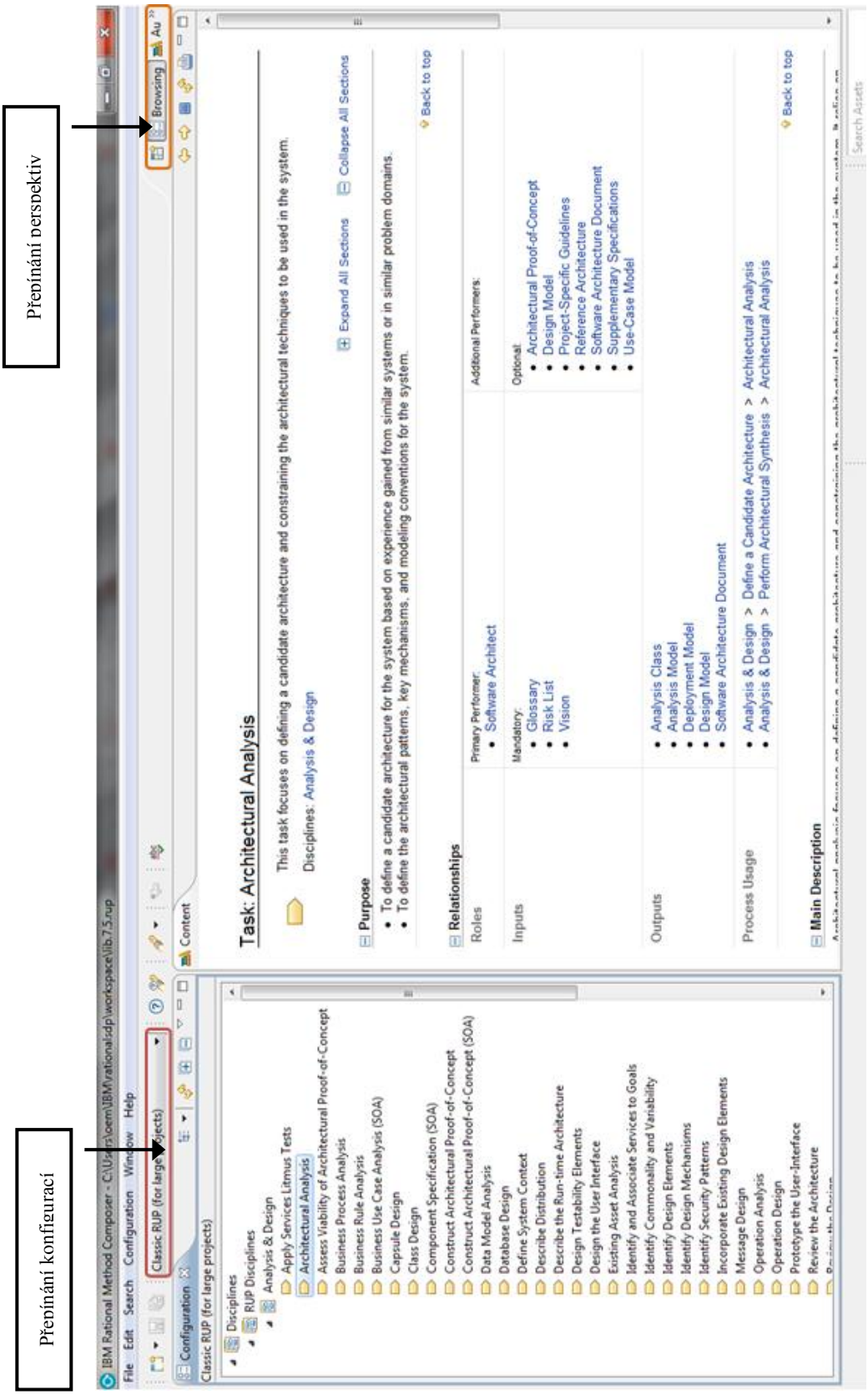
Ukázka popisu OpenUP metodiky vytvořeného v nástroje EPF Composer[30]

[30] Obrázek převzat z [4].



Ukázka perspektivy Authoring uživatelského prostředí RMC[31]

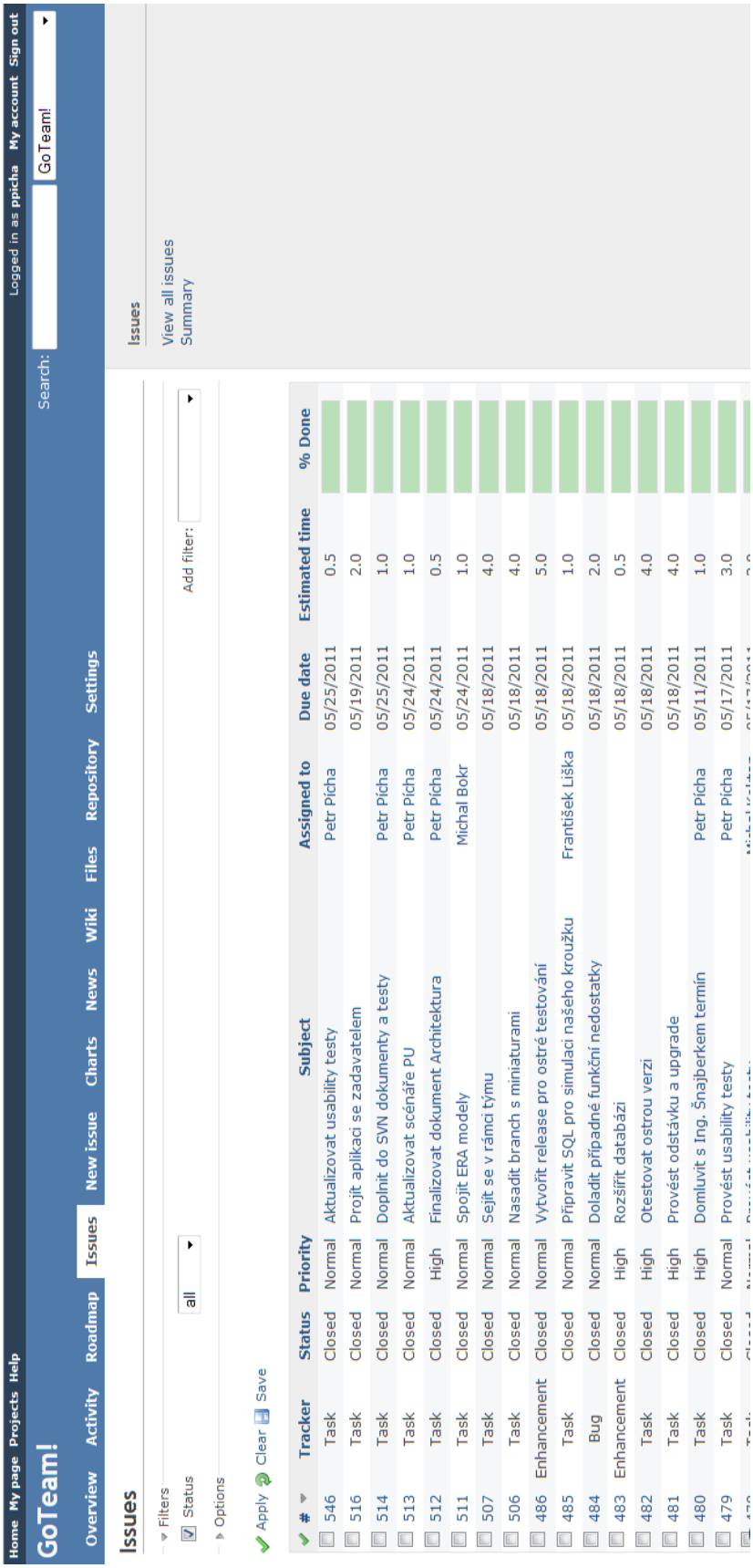
[31] Obrázek neukazuje standardní rozložení perspektivy při jejím prvním otevření, ale spíše její pro vývoj nejvhodnější nastavení upravené autorem této diplomové práce.



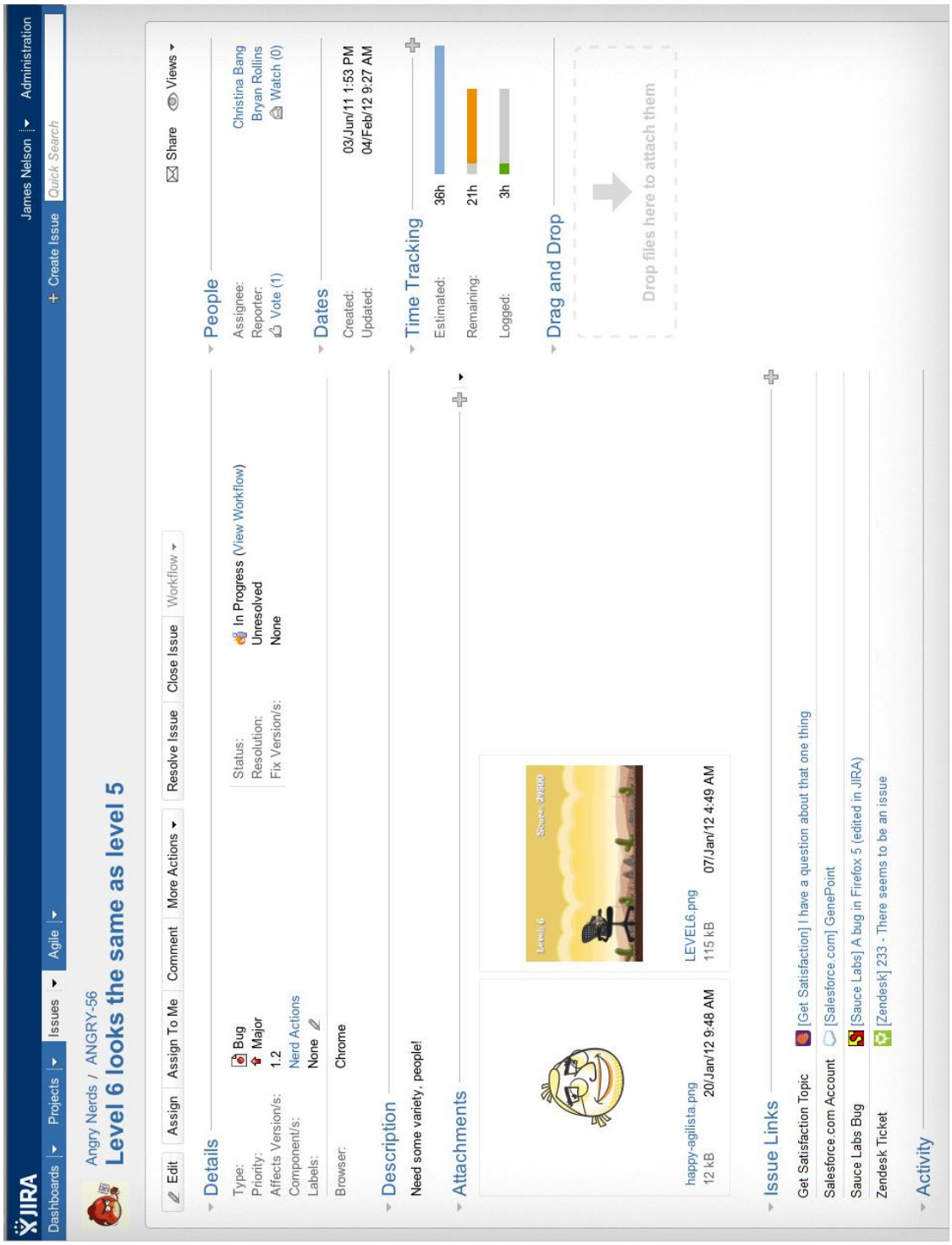
Ukázka perspektivy Browsing uživatelského rozhraní RMC[32]

[32] Na obrázku označené části jsou obecné pro všechny perspektivy, nikoli specifické pro perspektivu Browsing.

D. Obrázky k nástrojům řízení projektu



Ukázka uživatelského rozhraní nástroje Redmine

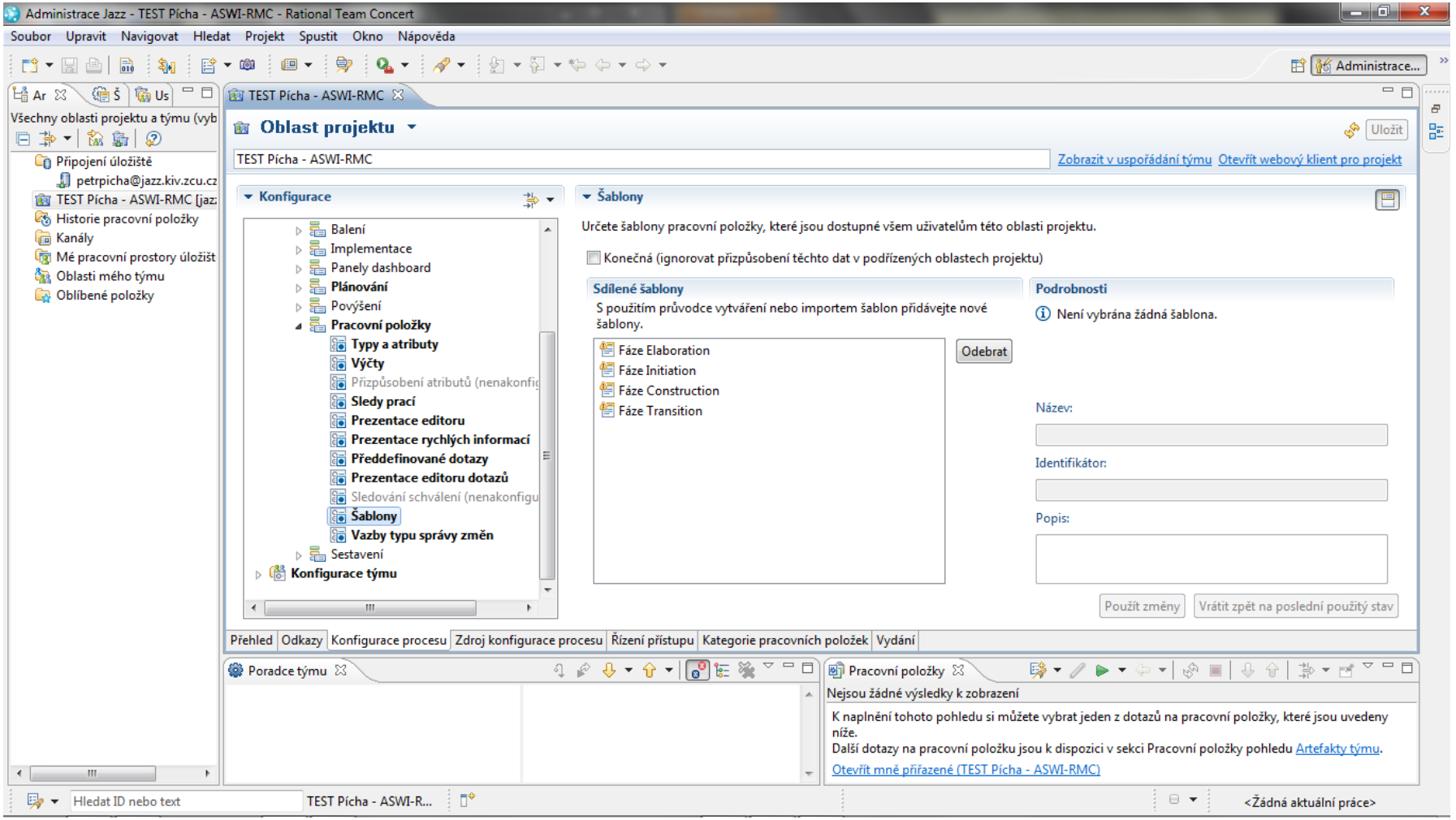


Ukázka uživatelského prostředí nástroje Atlassian Jira[33]

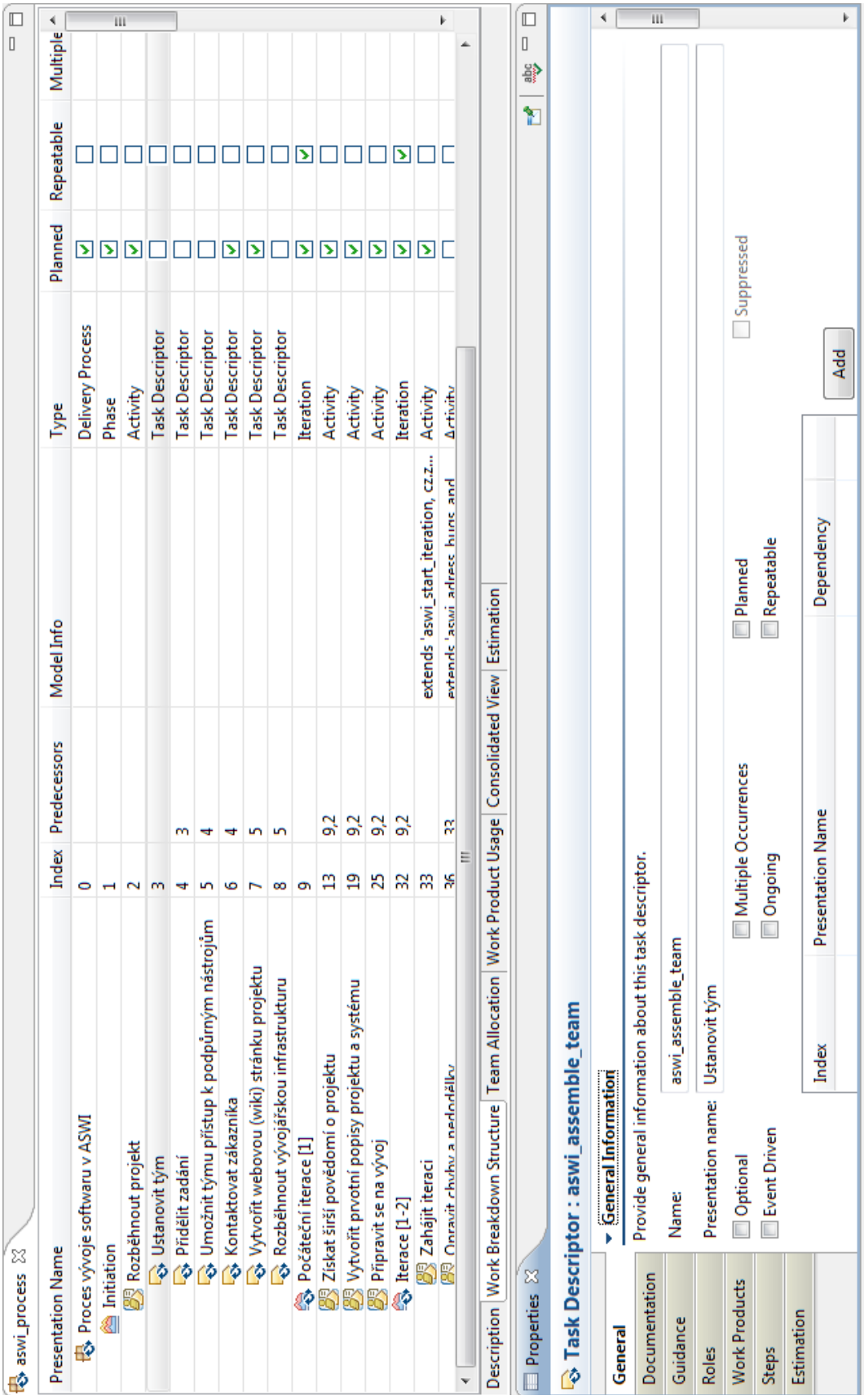
[33] Obrázek převzat z [14].

Ukázka uživatelského rozhraní serverové části RTC[34]

[34] Obrázek převzat z [15].


Administrace Jazz - TEST Pícha - ASWI-RMC - Rational Team Concert
Soubor Upravit Navigovat Hledat Projekt Spustit Okno Nápověda
Administrace...
Připojení úložiště
petrpicha@jazz.kiv.zcu.cz
TEST Pícha - ASWI-RMC [jaz
Historie pracovní položky
Kanály
Mé pracovní prostory úložišt
Oblasti mého týmu
Oblíbené položky
TEST Pícha - ASWI-RMC
Oblast projektu
Uložit
TEST Pícha - ASWI-RMC
Zobrazit v uspořádání týmu
Otevřít webový klient pro projekt
Konfigurace
Balení
Implementace
Panely dashboard
Plánování
Povýšení
Pracovní položky
Typy a atributy
Výčty
Přizpůsobení atributů (nenakonfi
Sledy prací
Prezentace editoru
Prezentace rychlých informací
Předdefinované dotazy
Prezentace editoru dotazů
Sledování schválení (nenakonfigu
Šablony
Vazby typu správy změn
Sestavení
Konfigurace týmu
Šablony
Určete šablony pracovní položky, které jsou dostupné všem uživatelům této oblasti projektu.
Konečná (ignorovat přizpůsobení těchto dat v podřízených oblastech projektu)
Sdílené šablony
S použitím průvodce vytváření nebo importem šablon přidávejte nové šablony.
Fáze Elaboration
Fáze Initiation
Fáze Construction
Fáze Transition
Odebrat
Podrobnosti
Není vybrána žádná šablona.
Název:
Identifikátor:
Popis:
Použít změny
Vrátit zpět na poslední použitý stav
Přehled
Odkazy
Konfigurace procesu
Zdroj konfigurace procesu
Řízení přístupu
Kategorie pracovních položek
Vydání
Poradce týmu
Pracovní položky
Nejsou žádné výsledky k zobrazení
K naplnění tohoto pohledu si můžete vybrat jeden z dotazů na pracovní položky, které jsou uvedeny níže.
Další dotazy na pracovní položku jsou k dispozici v sekci Pracovní položky pohledu Artefakty týmu.
Otevřít mně přiřazené (TEST Pícha - ASWI-RMC)
Hledat ID nebo text
TEST Pícha - ASWI-R...
<Žádná aktuální práce>

E. Obrázky k návodu pro práci s RMC



Editace deskriptoru v pohledu Properties View

F. Tabulka rolí v ASWI plug-inu

Následující tabulka obsahuje všechny role definované v ASWI plug-inu a jejich základní vlastnosti.

Prezentační název	Sada rolí	Vlastní kategorie	Balík obsahu
Člen týmu	Role ve vývojovém týmu	CC4	Balík rolí
Další zástupce zákazníka	Zástupci zákazníka	CC4	Balík rolí
Kontaktní osoba	Zástupci zákazníka	CC4	Balík rolí
Mentor	Akademický dozor	CC4	Balík rolí
Technický pracovník	Zástupci zákazníka	CC4	Balík rolí
Vedoucí týmu	Role ve vývojovém týmu	CC4	Balík rolí
Vývojový tým	Role ve vývojovém týmu	CC4	Balík rolí

Legenda:

- **Vlastní kategorie** – užívá jejich identifikátory z tabulky v příloze J

G. Tabulka úkolů v ASWI plug-inu

Následující tabulka obsahuje všechny úkoly definované v ASWI plug-inu a jejich základní vlastnosti.

ID	Prezentační název	Disciplíny	Vlastní kategorie	Balík obsahu
T1	**Archivovat původní verzi**	CCM	CC9, CC13, CC28, CC28	Obsah fáze Initiation
T2	**Dodělat zbytky z předchozího vývoje**	CCM, PM	CC8, CC13	Průběžné elementy
T3	**Dohodnout požadavky**	R	CC7, CC13	Obsah fáze Initiation
T4	**Finalizovat dokument Architektura**	CCM	CC8, CC13	Obsah fáze Construction
T5	**Finalizovat vizi**	AD, CCM	CC7, CC13	Obsah fáze Elaboration
T6	**Implementovat funkčnost**	CCM, I, T	CC7, CC13	Obsah fáze Construction
T7	**Implementovat spustitelnou architekturu**	AD, I, T	CC8, CC13	Obsah fáze Elaboration
T8	**Implementovat testy**	I, T	CC7, CC13	Obsah fáze Construction
T9	**Kontaktovat technického správce produktu**	AD, E, R	CCC7, CC12, CC13	Obsah fáze Initiation
T10	**Kontaktovat zákazníka**	PM	CC7, CC13	Obsah fáze Initiation
T11	**Najít a popsat aktéry**	AD, R	CC8, CC13	Obsah fáze Elaboration
T12	**Naplánovat iteraci**	CCM, PM	CC7, CC13	Průběžné elementy
T13	**Naplánovat nasazení**	D, PM	CC7, CC13	Obsah fáze Transition
T14	**Naplánovat projekt**	PM	CC7, CC13	Obsah fáze Initiation
T15	**Nasadit systém**	D	CC7, CC13	Obsah fáze Transition
T16	**Nastavit vývojové prostředí**	E	CC8, CC13	Obsah fáze Initiation
T17	**Ohodnotit iteraci**	PM	CC11, CC7, CC13	Průběžné elementy
T18	**Ohodnotit projekt**	PM	CC11, CC7, CC13	Obsah fáze Transition
T19	**Ohodnotit předaný produkt**	PM	CC11, CC7, CC13	Obsah fáze Transition
T20	**Opravit chyby v produktu**	CCM, I	CC7, CC13	Průběžné elementy
T21	**Popsat průběh iterace**	PM	CC8, CC12, CC13	Průběžné elementy
T22	**Popsat rizika**	AD, PM	CC8, CC13	Obsah fáze Initiation
T23	**Popsat stakeholdery**	AD	CC8, CC13	Obsah fáze Initiation
T24	**Potvrdit dosažení milníku**	CCM, PM	CC7, CC13	Průběžné elementy
T25	**Potvrdit dosažení milníku IOC**	CCM, PM	CC7, CC13	Obsah fáze Construction
T26	**Potvrdit dosažení milníku LOA**	CCM, PM	CC7, CC13	Obsah fáze Elaboration
T27	**Potvrdit dosažení milníku PRI**	CCM, PM	CC7, CC13	Obsah fáze Initiation
T28	**Potvrdit dosažení milníku REL**	CCM, PM	CC7, CC13	Obsah fáze Transition
T29	**Pročíst dokumentaci současné verze systému**	AD	CC8, CC13, CC28	Obsah fáze Initiation
T30	**Provést refactoring**	CCM, I	CC9, CC13	Průběžné elementy
T31	**Provést retrospektivu iterace**	CCM, PM	CC7, CC13	Průběžné elementy
T32	**Provést weekly standup**	PM	CC8, CC12, CC13	Průběžné elementy
T33	**Přidělit zadání**	PM	CC11, CC7, CC13	Obsah fáze Initiation

T34	**Rozběhnout vývojářskou infrastrukturu**	E	CC7, CC13	Obsah fáze Initiation
T35	**Řídit iteraci**	CCM, PM	CC7, CC13	Průběžné elementy
T36	**Sejít se se zákazníkem**	CCM, PM	CC7, CC12, CC13	Průběžné elementy
T37	**Sepsat retrospektivu projektu**	PM	CC7, CC13	Obsah fáze Transition
T38	**Setkat se se zákazníkem (poprvé)**	PM	CC7, CC12, CC13	Obsah fáze Initiation
T39	**Seznámit se s dosavadní verzí systému**	AD	CC8, CC13, CC28	Obsah fáze Initiation
T40	**Seznámit se s produkčním prostředím**	AD, E	CC7, CC13, CC31	Obsah fáze Initiation
T41	**Spustit testy**	CCM, T	CC7, CC13	Obsah fáze Construction
T42	**Umožnit týmu přístup k podpůrným nástrojům**	E, PM	CC11, CC7, CC13	Obsah fáze Initiation
T43	**Upravit artefakty**	CCM	CC7, CC13	Průběžné elementy
T44	**Upravit plán projektu**	CCM, PM	CC8, CC13	Průběžné elementy
T45	**Ustanovit konvence vývoje**	E	CC8, CC13	Obsah fáze Initiation
T46	**Ustanovit politiku konfiguračního řízení**	CCM, E	CC8, CC13	Obsah fáze Initiation
T47	**Ustanovit tým**	PM	CC7, CC13	Obsah fáze Initiation
T48	**Uzavřít projekt se zákazníkem**	D, PM	CC7, CC13	Obsah fáze Transition
T49	**Vybrat kandidátní architekturu**	AD, I, T	CC7, CC13	Obsah fáze Elaboration
T50	**Vytvářet a označovat stabilní verze**	CCM, I	CC8, CC13	Průběžné elementy
T51	**Vytvořit beta verzi produktu**	D, I, T	CC7, CC13	Obsah fáze Construction
T52	**Vytvořit další dokumenty**	D, PM	CC9, CC13	Obsah fáze Construction
T53	**Vytvořit dokument Architektura**	AD	CC7, CC13	Obsah fáze Elaboration
T54	**Vytvořit Dokument specifikace požadavků**	R	CC9, CC13	Obsah fáze Elaboration
T55	**Vytvořit dokument Vize produktu**	R	CC7, CC13	Obsah fáze Initiation
T56	**Vytvořit ERA model**	AD	CC8, CC13, CC29	Obsah fáze Elaboration
T57	**Vytvořit glosář**	AD	CC9, CC13	Obsah fáze Initiation
T58	**Vytvořit havarijní plán**	D, PM	CC8, CC13	Obsah fáze Transition
T59	**Vytvořit instalační příručku**	D, I	CC8, CC13	Obsah fáze Construction
T60	**Vytvořit model požadavků**	R	CC8, CC13	Obsah fáze Elaboration
T61	**Vytvořit patřičné modely**	AD	CC8, CC13	Obsah fáze Elaboration
T62	**Vytvořit popis požadavků**	AD, R	CC8, CC13	Obsah fáze Elaboration
T63	**Vytvořit produktovou dokumentaci**	D, I	CC8, CC13	Obsah fáze Construction
T64	**Vytvořit prototypy uživatelských rozhraní**	AD, I	CC8, CC13, CC30	Obsah fáze Elaboration
T65	**Vytvořit skript databáze**	I	CC9, CC13, CC29	Obsah fáze Elaboration
T66	**Vytvořit testovací případy**	AD, T	CC8, CC13	Obsah fáze Elaboration
T67	**Vytvořit uživatelskou dokumentaci**	D, I	CC7, CC13	Obsah fáze Construction
T68	**Vytvořit verzi produktu pro zákazníka**	D	CC7, CC13	Obsah fáze Transition
T69	**Vytvořit webovou (wiki) stránku projektu**	PM	CC7, CC13	Obsah fáze Initiation
T70	**Zaškolit uživatele**	D	CC8, CC13	Obsah fáze Transition
T71	**Získat prostředky pro vývoj**	E	CC8, CC13	Obsah fáze Initiation
T72	**Zrevidovat iteraci**	PM	CC7, CC12, CC13	Průběžné elementy

T73	**Zrevidovat projekt s mentorem**	PM	CC7, CC13	Obsah fáze Transition

Legenda:

- **ID** – identifikátor úkolu užitý v tabulkách v příloze K
- **Disciplíny**
  - ***AD*** – Analysis and Design (analýza a design)
  - ***CCM*** – Configuration and Change Management (řízení verzí a změn)
  - ***D***– Deployment (nasazení / vydání)
  - ***E***– Environment (prostředí)
  - ***I*** – Implementation (implementace)
  - ***PM*** – Project Management (řízení projektu)
  - ***R*** – Requirements (požadavky)
  - ***T***– Test (testování)
- **Vlastní kategorie** – užívá jejich identifikátory z tabulky v příloze J

H. Tabulka výsledků práce v ASWI plug-inu

Následující tabulka obsahuje všechny výsledky práce definované v ASWI plug-inu a jejich základní vlastnosti.

Prezentační název	Typ	Domény	Druhy výsledku práce	Vlastní kategorie	Balík obsahu
Archivovaná původní verze produktu	D	CCM	Sol	CC18, CC25, CC26, CC28	Obsah fáze Initiation
Beta verze aplikace	D	D, I, T	Sol	CC16, CC25, CC26	Obsah fáze Construction
Dokument Architektura	D	AD	M, Sp	CC16, CC25, CC26	Obsah fáze Elaboration
Dokument specifikace požadavků	A	R	Sp	CC18, CC24, CC26	Obsah fáze Elaboration
Dokument Vize produktu	D	R	Sp	CC16, CC25, CC26	Obsah fáze Elaboration
ERA model	A	AD	M	CC17, CC24, CC26, CC29	Obsah fáze Elaboration
Finální verze produktu	D	CCM, PM	PD, Sol	CC16, CC25, CC26	Obsah fáze Transition
Funkční vývojářské prostředí	D	E, I	I	CC16, CC25, CC26	Obsah fáze Initiation
Glosář	O	AD	Sp	CC18, CC23, CC26	Obsah fáze Initiation
Havarijní plán	O	D, PM	P	CC17, CC23, CC26	Obsah fáze Transition
Hodnocení iterace	O	PM	A, PD	CC16, CC20, CC23, CC26	Průběžné elementy
Hodnocení projektu	O	PM	A, PD	CC16, CC20, CC23, CC26	Obsah fáze Transition
Hodnocení zákazníka	O	PM	A, PD	CC16, CC20, CC23, CC26	Obsah fáze Transition
Implementace systému	D	I	Sol	CC16, CC25, CC26	Obsah fáze Construction
Implementované testy	A	I, T	Sol, Sp	CC16, CC24, CC26	Obsah fáze Construction
Informace o produkčním prostředí	O	E	I, PD, Sp	CC17, CC23, CC26, CC31	Obsah fáze Initiation
Instalační příručku	A	D, I	Sol, Sp	CC17, CC24, CC26	Obsah fáze Construction
Kandidátní architektura	O	AD, I, T	C, M, Sol	CC16, CC23, CC26	Obsah fáze Elaboration
Konvence používání SCM nástrojů	O	E	Sp	CC17, CC23, CC26	Obsah fáze Initiation

Model požadavků	A	R	M	CC17, CC24, CC26	Obsah fáze Elaboration
Ostatní dokumenty	A	D, PM	PD, Sp	CC18, CC24, CC26	Obsah fáze Construction
Plán iterace	A	PM	P	CC16, CC24, CC26	Průběžné elementy
Plán nasazení	O	D, PM	P	CC16, CC23, CC26	Obsah fáze Transition
Plán projektu	A	PM	P	CC16, CC24, CC26	Průběžné elementy
Popis požadavků	A	AD, R	M, Sp	CC17, CC24, CC26	Obsah fáze Elaboration
Popis rizik	O	AD	Sp	CC16, CC23, CC26	Obsah fáze Initiation
Programová dokumentace	A	D, I	PD, Sol, Sp	CC17, CC24, CC26	Obsah fáze Construction
Prototypy uživatelských rozhraní	A	AD, I	C, M, Sol, Sp	CC17, CC24, CC26, CC30	Obsah fáze Elaboration
Průběžná stabilní verze systému	A	CCM, D, I	Sol	CC17, CC24, CC26	Obsah fáze Construction
Předávací protokol	A	D, PM	PD	CC16, CC24, CC26	Obsah fáze Transition
Přídavné modely	A	AD	M	CC17, CC24, CC26	Obsah fáze Elaboration
Původní verze systému	D	CCM	Sol	CC17, CC25, CC26, CC28	Obsah fáze Initiation
Registrační formulář týmu	A	PM	PD	CC16, CC24, CC26	Obsah fáze Initiation
Retrospektiva projektu	A	PM	PD	CC16, CC24, CC26	Obsah fáze Transition
Seznam aktérů	O	AD, R	PD, Sp	CC17, CC23, CC26	Obsah fáze Elaboration
Seznam funkčních požadavků	O	R	PD, Sp	CC16, CC23, CC26	Obsah fáze Initiation
Seznam mimofunkčních požadavků	O	R	PD, Sp	CC16, CC23, CC26	Obsah fáze Initiation
Seznam požadavků na změny	O	CCM	PD	CC17, CC23, CC26	Průběžné elementy
Seznam stakeholderů	O	AD	PD, Sp	CC16, CC23, CC26	Obsah fáze Initiation
Seznam závad	O	CCM	PD	CC17, CC23, CC26	Průběžné elementy
Skript databáze	A	I	M, Sol	CC18, CC24, CC26, CC29	Obsah fáze Elaboration
Spustitelná architektura	A	AD, I, T	C, Sol	CC17, CC24, CC26	Obsah fáze Elaboration
Stránka projektu	A	PM	PD	CC16, CC24, CC26	Průběžné elementy
Testovací případy	A	AD, T	C, Sp	CC18, CC24, CC26	Obsah fáze Elaboration
Uzávěrka projektu	O	PM	A, PD	CC16, CC20, CC23, CC26	Obsah fáze Transition

Uživatelská dokumentace	A	D, I	Sol, Sp	CC16, CC24, CC26	Obsah fáze Construction
Verze produktu po fázi Construction	D	CCM, PM	Sol	CC16, CC25, CC26	Obsah fáze Construction
Verze produktu po fázi Elaboration	D	CCM, PM	Sol	CC16, CC25, CC26	Obsah fáze Elaboration
Verze produktu po fázi Initiation	D	CCM, PM	Sol	CC16, CC25, CC26	Obsah fáze Initiation
Verze produktu předávaná zákazníkovi	D	D	Sol	CC16, CC25, CC26	Obsah fáze Transition
Výsledky testů	O	T	PD	CC16, CC23, CC26	Obsah fáze Construction
Vývojářská infrastruktura	D	CCM, E, PM	I	CC16, CC25, CC26	Obsah fáze Initiation
Vývojářské konvence	O	E	Sp	CC18, CC23, CC26	Obsah fáze Initiation
Vývojové prostředky	D	E	I	CC16, CC25, CC26	Obsah fáze Initiation
Zadání projektu	O	PM	PD	CC16, CC23, CC26	Obsah fáze Initiation
Záznam retrospektivy iterace	O	CCM, PM	PD	CC17, CC23, CC26	Průběžné elementy
Záznam schůzky	O	CCM, PM, R	PD	CC18, CC23, CC26	Průběžné elementy
Záznam schůzky se zákazníkem	O	CCM, PM, R	PD	CC18, CC23, CC26	Průběžné elementy
Záznam weekly standupu	O	CCM, PM	PD	CC17, CC23, CC26	Průběžné elementy

Legenda:

- **Typ**
  - ***A*** – Artifact (artefakt)
  - ***D*** – Deliverable (předávaná)
  - ***O***– Outcome (výstup)
- **Domény**
  - ***AD*** – Analysis and Design (analýza a nasazení)
  - ***CCM*** – Configuration and Change Management (řízení verzí a změn)
  - ***D***– Deployment (nasazení / vydání)
  - ***E***– Environment (prostředí)
  - ***I*** – Implementation (implementace)
  - ***PM*** – Project Management (řízení projektu)
  - ***R*** – Requirements (požadavky)
  - ***T***– Test (testování)
- **Druhy výsledku práce**
  - ***A*** – Assessment (stanovení / ohodnocení / výměr)
  - ***C***– Concept (koncept)
  - ***I***– Infrastructure (infrastruktura)
  - ***M*** – Model (model)
  - ***P***– Plan (plán)
  - ***PD*** – Project Data (data projektu)
  - ***Sol*** – Solution (řešení)

- ***Sp***– Specification (specifikace)

- **Vlastní kategorie** – užívá jejich identifikátory z tabulky v příloze J

a. Tabulka stavů výsledků práce v ASWI plug-inu

Následující tabulka obsahuje všechny stavy výsledků práce definované v ASWI plug-inu.

Název
100% scénářů
20% scénářů
80% scénářů
90% funkčnosti
Aktualizovaná verze
Běžící v produkčním prostředí
Finalizovaná verze
Funkční
Jedno definitivně vybrané arch. řešení
Nepodepsaný
Podepsaný
Prvotní verze
Soubor možných architektonických řešení
Stručné zadání
Širší popis zadání
Základ

I. Tabulka pomocných materiálů v ASWI plug-inu

Následující tabulka obsahuje všechny pomocné materiály definované v ASWI plug-inu a jejich základní vlastnosti.

Prezentační název	Typ	Vlastní kategorie	Balík obsahu
Agilní vývoj	P	CC38	Balík praktik
ASWI copyright	SM		Copyright balík
Backlog iterace	C	CC39	Balík konceptů
Backlog projektu	C	CC39	Balík konceptů
Brzké adresování rizik	P	CC38	Balík praktik
Dozor akademického mentora	P	CC38	Balík praktik
Flyspray	TM		Průběžné elementy
Hodnocení projektu	P	CC38	Balík praktik
Iterační retrospektiva	P	CC38	Balík praktik
Iterativní vývoj	P	CC38	Balík praktik
Jira	TM		Průběžné elementy
Metodika RUP	P	CC38	Balík praktik
Metodika Scrum	P	CC38	Balík praktik
Modely verzí pro správu konfigurací	W	CC37	Balík obecných pomocných materiálů
Návod pro user stories	W	CC39	Obsah fáze Elaboration
Popis procesu ASWI	W	CC37	Balík obecných pomocných materiálů
Proof Of Concept	C	CC39	Balík konceptů
Příklad dokumentu architektury	E	CC34	Obsah fáze Elaboration
Příklad Flyspray reportu	E	CC34	Průběžné elementy
Příklad modelování business pravidel	E	CC34	Obsah fáze Elaboration
Příklad plánu iterace	E	CC34	Průběžné elementy
Příklad registračního formuláře týmu	E	CC34	Obsah fáze Initiation
Příklad SVN logu	E	CC34	Průběžné elementy
Příklad zápisu z iterační retrospektivy	E	CC34	Průběžné elementy
Přírůstkový vývoj	P	CC38	Balík praktik
Rational Team Concert	TM		Průběžné elementy
Redmine	TM		Průběžné elementy
Redmine manuál	W	CC36	Průběžné elementy
Refactoring	P	CC38	Balík praktik
Retrospektiva projektu	P	CC38	Balík praktik
Revize iterace	P	CC38	Balík praktik
Revize projektu	P	CC38	Balík praktik
Rozdělení na fáze	P	CC38	Balík praktik

RTC manuál	E	CC36	Průběžné elementy
Scénář případu užití	C	CC39	Balík konceptů
Seznam otázek pro iterační retrospektivu	W	CC35	Průběžné elementy
Schůzky se zákazníkem	P	CC38	Balík praktik
Soustavná úprava artefaktů	P	CC38	Balík praktik
StudentWiki @KIV	TM		Průběžné elementy
Subversion	TM		Průběžné elementy
Šablona plánu iterace	T	CC35	Průběžné elementy
Šablona plánu projektu	T	CC35	Průběžné elementy
Šablona předávacího protokolu	T	CC35	Obsah fáze Transition
Timeboxing	P	CC38	Balík praktik
Tutorial pro průběžnou integraci	W	CC37	Balík obecných pomocných materiálů
Ukázka dokumentu Vize	E	CC34	Obsah fáze Initiation
Ukázky diagramů architektury	E	CC34	Obsah fáze Elaboration
Use case diagram	C	CC39	Balík konceptů
User stories	C	CC39	Balík konceptů
Uzavření projektu	P	CC38	Balík praktik
Využití nástrojů pro řízení změn	P	CC38	Balík praktik
Využití nástrojů pro řízení změn a verzí	P	CC38	Balík praktik
Využití verzovacích nástrojů	P	CC38	Balík praktik
Vývoj řízený případy užití	P	CC38	Balík praktik
Weekly standup	P	CC38	Balík praktik
Záznam průběhu projektu	P	CC38	Balík praktik

Legenda:

- **Typ**
  - ***C*** – Concept (koncept)
  - ***E*** – Example (příklad)
  - ***P*** – Practice (praktika)
  - ***SM*** – Supporting Material (podpůrný materiál)
  - ***T*** – Template (šablona)
  - ***TM*** – Tool Mentor (návod k nástroji)
  - ***W*** – Whitepaper (informační dokument)
- **Vlastní kategorie** – užívá jejich identifikátory z tabulky v příloze J

J. Tabulka vlastních kategorií v ASWI plug-inu

Následující tabulka obsahuje všechny vlastní kategorie definované v ASWI plug-inu a jejich základní vlastnosti.

ID	Prezentační název
CC1	**Komplexní kategorie**
CC2	***Procesní pohled***
CC3	***Přehled rolí***
CC4	Všechny role
CC5	***Přehled úkolů***
CC6	Úkoly podle důležitosti
CC7	*Nutné úkoly*
CC8	*Vhodné úkoly*
CC9	*Nepovinné úkoly*
CC10	Úkoly podle disciplín
CC11	*Project Assessment*
CC12	Přehled schůzek
CC13	Všechny úkoly
CC14	***Přehled výsledků práce***
CC15	Výsledky práce podle důležitosti
CC16	*Nutné výsledky práce*
CC17	*Vhodné výsledky práce*
CC18	*Nepovinné výsledky práce*
CC19	Výsledky práce podle disciplín
CC20	*Project Assessment*
CC21	Výsledky práce podle druhu
CC22	Výsledky podle úrovně formality
CC23	*Výstupy*
CC24	*Artefakty*
CC25	*Předávané*
CC26	Všechny výsledky práce
CC27	***Projektově závislé elementy***
CC28	Specifika brown-field projektů
CC29	Specifika databázových systémů
CC30	Specifika systémů s uživatelským rozhraním
CC31	Specifika trvale nasazených systémů
CC32	***Přehled nástrojů***
CC33	***Přehled pomocných materiálů***

CC34	*Příklady*
CC35	*Šablony*
CC36	*Návody pro nástroje*
CC37	*Obecné popisy*
CC38	*Praktiky*
CC39	*Koncepty*
CC40	***Přehled procesních vzorů***

Legenda:

- **ID** – identifikátor vlastní kategorie užitý v tabulkách v přílohách F – I
- **Prezentační název** – zobrazuje hierarchii kategorií, kategorie různých úrovní jsou označeny různými styly písma

K. Struktura procesů a procesních vzorů v ASWI plug-inu

a. Procesní vzor Iterace

Následující tabulka zachycuje strukturu procesního vzoru Iterace v ASWI plug-inu.

Prezentační název	Typ	Úkol	Upraven
Iterace	Capability Pattern	-	-
Zahájit iteraci	Activity	-	-
Sejít se se zákazníkem	Task Descriptor	T36	ano
Naplánovat iteraci	Task Descriptor	T12	ano
Opravit chyby a nedodělky	Activity	-	-
Dodělat zbytky z předchozího vývoje	Task Descriptor	T2	ano
Opravit chyby v produktu	Task Descriptor	T20	ano
Posunout vývoj	Activity	-	-
Upravit plán projektu	Task Descriptor	T44	ano
Upravit artefakty	Task Descriptor	T43	ano
Provést refactoring	Task Descriptor	T30	ano
Vytvářet a označovat stabilní verze	Task Descriptor	T50	ano
Řídit iteraci	Activity	-	-
Řídit iteraci	Task Descriptor	T35	ano
Ukončit iteraci	Activity	-	-
Sejít se se zákazníkem	Task Descriptor	T36	ano
Provést retrospektivu iterace	Task Descriptor	T31	ne
Zrevidovat iteraci	Task Descriptor	T72	ne
Ohodnotit iteraci	Task Descriptor	T17	ne
Popsat průběh iterace	Task Descriptor	T21	ano
Týden	Capability Pattern	-	-
Provést každotýdenní úkoly	Activity	-	-
Provést weekly standup	Task Descriptor	T32	ne

Legenda:

- **Prezentační název** – zobrazuje hierarchii procesu, prvky různých úrovní jsou označeny různými styly písma
- **Úkol** – úkol, jehož kopií je deskriptor, užívá identifikátory úkolů z tabulky v příloze G
- **Upraven** – označuje, zda byl deskriptor úkolu upraven proti jeho vzoru z předešlého sloupce (k vstupům a výstupům byly přidány stavy, byl změněn název, popis nebo množina kroků)

b. Exportovaný proces ASWI

Následující tabulka zachycuje strukturu procesu životního cyklu Exportovaný proces ASWI v ASWI plug-inu.

Prezentační název	Typ	Úkol	Upraven
Exportovaný proces ASWI	Delivery Process	-	-
Initiation	Phase	-	-
Kontaktovat zákazníka	Task Descriptor	T10	ne
Setkat se se zákazníkem (poprvé)	Task Descriptor	T38	ne
Vytvořit webovou (wiki) stránku projektu	Task Descriptor	T69	ne
Naplánovat projekt	Task Descriptor	T14	ne
Vytvořit dokument Vize produktu	Task Descriptor	T55	ne
Iterace	Iteration	-	-
Naplánovat iteraci	Task Descriptor	T12	ne
Sejít se se zákazníkem	Task Descriptor	T36	ne
Provést retrospektivu iterace	Task Descriptor	T31	ne
Zrevidovat iteraci	Task Descriptor	T72	ne
Popsat průběh iterace	Task Descriptor	T21	ne
Týden	Iteration	-	-
Provést weekly standup	Task Descriptor	T32	ne
Potvrdit dosažení milníku PRI	Task Descriptor	T27	ne
Elaboration	Phase	-	-
Vytvořit popis nejzásadnějších požadavků	Task Descriptor	T62	ano
Vytvořit popis většiny požadavků	Task Descriptor	T62	ano
Finalizovat vizi	Task Descriptor	T5	ne
Vybrat architekturu	Task Descriptor	T49	ano
Vytvořit dokument Architektura	Task Descriptor	T53	ne
Iterace[35]	Iteration	-	-
Potvrdit dosažení milníku LOA	Task Descriptor	T26	ne
Construction	Phase	-	-
Vytvořit popis zbývajících požadavků	Task Descriptor	T62	ano
Finalizovat dokument Architektura	Task Descriptor	T4	ne
Implementovat testy	Task Descriptor	T8	ne
Spustit testy	Task Descriptor	T41	ne
Iterace[35]	Iteration	-	-
Vytvořit beta verzi produktu	Task Descriptor	T51	ne
Potvrdit dosažení milníku IOC	Task Descriptor	T25	ne

[35] Obsah iterace je stejný jako ve fázi Initiation.

Transition		Phase	-	-
	Implementovat funkční a beta testy	Task Descriptor	T8	ano
	Spustit testy	Task Descriptor	T41	ne
	Vytvořit produktovou dokumentaci	Task Descriptor	T63	ne
	Vytvořit instalační příručku	Task Descriptor	T59	ne
	Vytvořit uživatelskou dokumentaci	Task Descriptor	T67	ne
	Vytvořit verzi produktu pro zákazníka	Task Descriptor	T68	ne
	Nasadit systém	Task Descriptor	T15	ne
	Iterace[35]	Iteration	-	-
	Potvrdit dosažení milníku REL	Task Descriptor	T28	ne
	Uzavřít projekt se zákazníkem	Task Descriptor	T48	ne
	Sepsat retrospektivu projektu	Task Descriptor	T37	ne
	Zrevidovat projekt s mentorem	Task Descriptor	T73	ne

Legenda:

- **Prezentační název** – zobrazuje hierarchii procesu, prvky různých úrovní jsou označeny různými styly písma
- **Úkol** – úkol, jehož kopií je deskriptor, užívá identifikátory úkolů z tabulky v příloze G
- **Upraven** – označuje, zda byl deskriptor úkolu upraven proti jeho vzoru z předešlého sloupce (k vstupům a výstupům byly přidány stavy, byl změněn název, popis nebo množina kroků)

c. Proces vývoje software v ASWI

Následující tabulka zachycuje strukturu procesu životního cyklu Proces vývoje ASWI v ASWI plug-inu.

Prezentační název	Typ	Úkol	Upraven
Proces vývoje softwaru v ASWI	Delivery Process	-	-
Initiation	Phase	-	-
Rozběhnout projekt	Activity	-	-
Ustanovit tým	Task Descriptor	T47	ne
Přidělit zadání	Task Descriptor	T33	ano
Umožnit týmu přístup k podpůrným nástrojům	Task Descriptor	T42	ano
Kontaktovat zákazníka	Task Descriptor	T10	ano
Vytvořit webovou (wiki) stránku projektu	Task Descriptor	T69	ano
Rozběhnout vývojářskou infrastrukturu	Task Descriptor	T34	ano
Počáteční iterace [1]	Iteration	-	-
Týden[36]	Capability Pattern	-	-
Získat širší povědomí o projektu	Activity	-	-
Setkat se se zákazníkem (poprvé)	Task Descriptor	T38	ano
Kontaktovat technického správce produktu	Task Descriptor	T9	ne
Pročíst dokumentaci současné verze systému	Task Descriptor	T29	ne
Seznámit se s dosavadní verzí systému	Task Descriptor	T39	ne
Seznámit se s produkčním prostředím	Task Descriptor	T40	ne
Vytvořit prvotní popisy projektu a systému	Activity	-	-
Popsat stakeholdery	Task Descriptor	T23	ano
Dohodnout požadavky	Task Descriptor	T3	ano
Popsat rizika	Task Descriptor	T22	ano
Vytvořit dokument Vize produktu	Task Descriptor	T55	ano
Vytvořit glosář	Task Descriptor	T57	ne
Připravit se na vývoj	Activity	-	-
Ustanovit politiku konfiguračního řízení	Task Descriptor	T46	ne
Naplánovat projekt	Task Descriptor	T14	ano
Archivovat původní verzi	Task Descriptor	T1	ne
Získat prostředky pro vývoj	Task Descriptor	T71	ne
Nastavit vývojové prostředí	Task Descriptor	T16	ne
Ustanovit konvence vývoje	Task Descriptor	T45	ano

[36] Struktura tohoto procesního vzoru je popsána v textu práce v oddíle *6.2.2–Procesní vzory* a je stejná jako u procesního vzoru *Iterace* v části a této přílohy.

Iterace [1-2][37]	Iteration	-	-
Ukončit fázi	Activity	-	-
Potvrdit dosažení milníku PRI	Task Descriptor	T24	ano
PRI (Project Initialized) milník	Milestone	-	-
Elaboration	Phase	-	-
Zpřesnit vizi	Activity	-	-
Zpřesnit popis stakeholderů	Task Descriptor	T23	ano
Zpřesnit požadavky	Task Descriptor	T3	ano
Zpřesnit popis rizik	Task Descriptor	T22	ano
Aktualizovat dokument Vize produktu	Task Descriptor	T55	ano
Zpřesnit rozsah a obsah projektu	Activity	-	-
Seznámit se s produkčním prostředím	Task Descriptor	T40	ne
Vytvořit nebo rozšířit glosář	Task Descriptor	T57	ano
Najít a popsat aktéry	Task Descriptor	T11	ne
Vytvořit model požadavků	Task Descriptor	T60	ne
Vytvořit popis klíčových požadavků	Task Descriptor	T62	ano
Stabilizovat požadavky	Activity	-	-
Vytvořit popis většiny požadavků	Task Descriptor	T62	ano
Finalizovat vizi	Task Descriptor	T5	ano
Vytvořit Dokument specifikace požadavků	Task Descriptor	T54	ne
Navrhnout prvotní podobu systému	Activity	-	-
Vytvořit prototypy uživatelských rozhraní	Task Descriptor	T64	ne
Vybrat kandidátní architektury	Task Descriptor	T49	ano
Baselinovat architekturu	Activity	-	-
Vybrat architekturu	Task Descriptor	T49	ano
Vytvořit ERA model	Task Descriptor	T56	ne
Vytvořit patřičné architektonické modely	Task Descriptor	T61	ano
Implementovat spustitelnou architekturu	Task Descriptor	T7	ne
Vytvořit dokument Architektura	Task Descriptor	T53	ano
Připravit se na implementaci	Activity	-	-
Seznámit se s dosavadní verzí systému	Task Descriptor	T39	ne
Nastavit vývojové prostředí	Task Descriptor	T16	ne
Ustanovit konvence vývoje	Task Descriptor	T45	ne
Vytvořit testovací případy	Task Descriptor	T66	ano
Vytvořit skript databáze	Task Descriptor	T65	ne
Iterace [1-2][37]	Iteration	-	-
Ukončit fázi	Activity	-	-

[37] Struktura iterace přesně odpovídá té z procesního vzoru *Iterace* v části a této přílohy.

Potvrdit dosažení milníku LOA	Task Descriptor	T24	ano
LOA (Lifecycle Objectives and Architecture) milník	Milestone	-	-
Construction	Phase	-	-
Dokončit popis požadavků a architektury	Activity	-	-
Vytvořit popis zbylých požadavků	Task Descriptor	T62	ano
Finalizovat dokument Architektura	Task Descriptor	T4	ano
Finalizovat Dokument specifikace požadavků	Task Descriptor	T54	ano
Implementovat systém	Activity	-	-
Vytvořit beta verzi produktu	Task Descriptor	T51	ano
Připravit a aplikovat testy	Activity	-	-
Vytvořit (rozšířit) testovací případy	Task Descriptor	T66	ano
Implementovat testy	Task Descriptor	T8	ne
Spustit testy	Task Descriptor	T41	ne
Zahájit práce na dokumentaci	Activity	-	-
Vytvořit produktovou dokumentaci	Task Descriptor	T63	ano
Vytvořit instalační příručku	Task Descriptor	T59	ano
Vytvořit uživatelskou dokumentaci	Task Descriptor	T67	ano
Vytvořit další dokumenty	Task Descriptor	T52	ano
Iterace [2-n][38]	Iteration	-	-
Posunout vývoj	Activity	-	-
Implementovat funkčnost	Task Descriptor	T6	ano
Ukončit fázi	Activity	-	-
Potvrdit dosažení milníku IOC	Task Descriptor	T24	ano
IOC (Initial Operational Capability) milník	Milestone	-	-
Transition	Phase	-	-
Dokončit vývoj a testování	Activity	-	-
Implementovat funkční a beta testy	Task Descriptor	T8	ano
Spustit testy	Task Descriptor	T41	ano
Vytvořit verzi produktu pro zákazníka	Task Descriptor	T68	ano
Dokončit dokumentaci	Activity	-	-
Vytvořit (dokončit) produktovou dokumentaci	Task Descriptor	T63	ano
Vytvořit (dokončit) instalační příručku	Task Descriptor	T59	ano
Vytvořit (dokončit) uživatelskou dokumentaci	Task Descriptor	T67	ano
Vytvořit (dokončit) další dokumenty	Task Descriptor	T52	ano
Naplánovat a provést nasazení	Activity	-	-
Naplánovat nasazení	Task Descriptor	T13	ne

[38] Struktura iterace odpovídá té z procesního vzoru *Iterace* v části a této přílohy. Přidán je pouze deskriptor úkolu „*Implementovat funkčnost*“ v aktivitě „*Posunout vývoj*“, jak je zde naznačeno.

Vytvořit havarijní plán	Task Descriptor	T58	ne
Nasadit systém	Task Descriptor	T15	ano
Zaškolit uživatele	Task Descriptor	T70	ano
Iterace [1-2][38]	Iteration	-	-
Posunout vývoj	Activity	-	-
Implementovat funkčnost	Task Descriptor	T6	ano
Ukončit fázi	Activity	-	-
Uzavřít projekt se zákazníkem	Task Descriptor	T48	ano
Potvrdit dosažení milníku REL	Task Descriptor	T24	ano
REL (Product Release) milník	Milestone	-	-
Uzavřít projekt	Activity	-	-
Ohodnotit předaný produkt	Task Descriptor	T19	ne
Sepsat retrospektivu projektu	Task Descriptor	T37	ano
Zrevidovat projekt s mentorem	Task Descriptor	T73	ano
Ohodnotit projekt	Task Descriptor	T18	ano

Legenda:

- **Prezentační název** – zobrazuje hierarchii procesu, prvky různých úrovní jsou označeny různými styly písma
- **Úkol** – úkol, jehož kopií je deskriptor, užívá identifikátory úkolů z tabulky v příloze G
- **Upraven** – označuje, zda byl deskriptor úkolu upraven proti jeho vzoru z předešlého sloupce (k vstupům a výstupům byly přidány stavy, byl změněn název, popis nebo množina kroků)
- *Pozn.:* Čísla v hranatých závorkách u iterací označují jejich běžný počet v rámci dané fáze.