VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

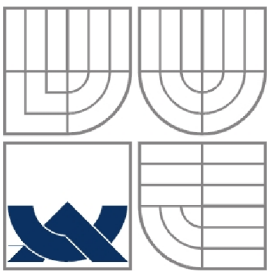
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

SYSTÉM ŘÍZENÍ WORKFLOW

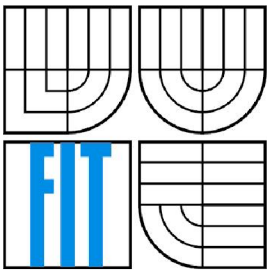
BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE Jan Šimecký
AUTHOR

BRNO 2011



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

SYSTÉM ŘÍZENÍ WORKFLOW

WORKFLOW MANAGEMENT SYSTEM

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR

JAN ŠIMECKÝ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. VOJTĚCH MATES

BRNO 2011

Abstrakt

Tato bakalářská práce je zaměřená na problematiku workflow management systémů, které slouží k automatickému řízení a monitorování podnikových procesů. Je zde popsána související terminologie a referenční model, který přináší mezi jednotlivé výrobce workflow produktů standardizaci. V rámci této práce byl navrhnut a realizován workflow management systém. Jeho klíčovou vlastností je možnost uživatelské editace úloh procesu a podpora rozhodovací logiky větvení procesu na základě hodnot získaných v jeho průběhu.

Klíčová slova

Workflow, podnikové procesy, automatizace podnikových procesů, monitoring podnikových procesů, optimalizace podnikových procesů

Abstract

This Bachelor´s thesis is focused on the area of workflow management systems which serve for automatic control and monitoring business processes. Related terminology and the reference model that brings standardization amongst individual workflow products manufacturers are described in the thesis. A workflow management system was designed in process of writing this thesis. Its key features are the possibility for a user to edit tasks of a process and the support of decision logic of process branching on the basis of the variables acquired during the process.

Keywords

Workflow, business process, automatization of business process, monitoring of business process, optimization of business process

Citace

Šimecký Jan: Systém řízení workflow. Brno, 2011, bakalářská práce, FIT VUT v Brně.

Systém řízení workflow

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Vojtěcha Matese. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Jan Šimecký
16.5.2011

Poděkování

Tímto bych chtěl poděkovat vedoucímu práce Ing. Vojtěchovi Matesovi za jeho velkou ochotou během průběžných konzultací.

Obsah

Úvod

Cílem této bakalářské práce bylo se seznámit s problematikou workflow systémů, které slouží pro řízení pracovních postupů, a následně vytvořit fungující zjednodušený workflow sytém. Workflow systémy přinášejí organizacím dohled a automatické řízení jednotlivých aktivit, které jsou rozdělené na menší celky (úlohy). Spolu s informačními technologiemi nám takto získávají další prostředek pro minimalizaci nákladů při probíhajících prací, a tím i celkové zvýšení produktivity. Workflow systémy se začali objevovat kolem roku 1990 a u nás v České republice se stále tyto principy využívají pouze v malém měřítku.

V 1. kapitole se nacházejí informace o zasazení problematiky workflow mezi informační technologie a je zde také představena organizace WfMC, která se snaží přinášet standardizaci do této problematiky. V 2. kapitole se nacházejí informace o workflow systémech, jejich terminologii a jak tyto systémy fungují. Cílem kapitoly je dále upozornit na to, že zavedení a fungování workflow systémů se skládá mj. z analýzy a optimalizace dosavadně prováděných obchodních činností. 3. kapitola se zabývá referenčním modelem workflow systémů, který by měl zajišťovat kompatibilitu mezi více workflow produkty. V tomto referenčním modelu je definováno celkem pět rozhraní, které mají zajišťovat kompatibilitu mezi jednotlivými funkčními celky. Středem je workflow řídící služba, která zapouzdřuje výkonné jádro a komunikuje přes definovaná rozhraní s nástroji pro definicemi procesů, klientskou aplikací, spouštěnými aplikacemi, dalšími workflow řídícími službami a administrativními nástroji. Ve 4. kapitole se nacházejí informace o použitých implementačních technologiích, informace o jejich historii a další použitá rozšíření daných prostředí. 5. kapitola se věnuje struktuře implementovaného workflow systému. Je zde popsáno, jak jsou uložená jednotlivá data a vazby mezi těmito daty. Dále je zde popsáno fungovaní celého systému na pozadí – generování a přidělování úkolů pro jednotlivé běžící pracovní procesy. V 6. kapitole se nachází seznámení s funkcionalitou, kterou poskytuje uživatelské rozhraní pro jednotlivé části systému. Je zde popsána tvorba zdrojů a jejich způsobů užití, definice jednotlivých uživatelů a uživatelských rolí, správa a samotné definice procesů, seznam úkolů uživatelů a možnosti monitoringu v rámci tohoto systému.

1 Role informačních technologií

Práce ve všech větších organizací a společnostech je již bez podpory informačních technologií nepředstavitelná. Informační technologie pomáhají zaměstnancům při jejich každodenní činnosti. Dle zdroje [2] je ale zapotřebí lépe pochopit, jak nám informační technologie mohou pomoci a jak můžou usnadnit vynaložené pracovní úsilí zaměstnanců, zjednodušit řízení firmy a výroby. Zřízení a prosazení IT ve firmě musí být provázáno se spoluúčastí celé firmy – zaměstnanců, včetně managementu, který musí využívání IT řídit. Je třeba mít také na paměti, že lidé nemají rádi změny, rádi využívají staré zaběhlé postupy a neradi se učí novým. Při zavádění IT je tedy zapotřebí řádně zaměstnancům vysvětlit výhody a přínos informačních technologií. Velmi dobrá je i patřičná motivace. Patřičnou motivací může být například odměna za strávený čas pro účely IT modernizace.

1.1 Informační systém vs. procesní informační systém

Dle zdroje [1] můžeme mluvit o informačním a procesně informačním systému. Informační systém poskytuje uživatelům podporu k běžným úkolům, jako může být například evidence. Procesní informační systém nám přináší navíc plánování aktivit dle specifikovaných definic. Správa takovýchto informačních systému je ale bohužel velmi nákladná a náročná. Při každé změně v pracovních procesech se musí provádět úprava, rekompilace a distribuce takovéhoto informačního systému. Každá takováto úprava může způsobit zanášení nových chyb do celého informačního systému. Principy workflow systémů přinášejí oddělení logických pracovních postupů od aplikační funkcionality. Workflow systémy přinášejí propracované komfortní nástroje pro správu a organizaci pracovních procesů. Pomocí těchto nástrojů lze provádět jednoduché změny bez přepisování zdrojových kódů aplikace, její rekompilace a distribuce. Dělení informačních systému na více typů lze pak nalézt ve zdroji [7].

1.2 WfMC

WfMC (Workflow Managment Coalition) je nezisková mezinárodní organizace, která stojí za vydáním zdroje [3]. WfMC je skupina společností, které se spolu spojili za účelem vytvoření standardů pro workflow systémy. Tyto standardy jsou potřebné z důvodu potřeby spolupráce a kompatibility jednotlivých workflow systémů.

2 Co je to workflow?

Workflow je obvykle označován definicí *„The computerised facilitation or automation of a business process, in whole or part“* [3]. Tento termín v překladu znamená „Počítačová podpora pro zjednodušení nebo automatizaci podnikových procesů v celku nebo z části“.

Dle zdroje [3] se workflow týká automatizace procedur, kde jsou informace, dokumenty a úlohy předávané mezi účastníky za pomocí předem definované množiny pravidel. Tato množina pravidel je vytvořena za účelem dosažení požadovaných obchodních cílů. Workflow proces může být organizován i manuálně, ale z praxe je lepší využít informačních technologii pro automatické zpracování, řízení a monitoring procesů. Životní cyklus jednotlivých procesů se v závislosti na jejich složitostech může pohybovat v rozmezí několika málo minut až po dny a měsíce. Každý proces může být implementován rozdílnými způsoby (stejně jak programy mohou být různě implementované).

Workflow systém automatizuje podnikové procesy, určuje posloupnost procesů a přiděluje těmto procesům zdroje. Každý proces se skládá z více úloh. Jedna tato úloha je nejmenší možný časový celek, který je ohraničený vstupní a výstupní podmínkou. Úlohy jsou mezi sebou logicky propojeny a dohromady tvoří realizaci daného podnikového procesu. Každá úloha, kromě vstupní a výstupní podmínky, musí mít také přiřazené podnikové role (kdo danou činnost z organizační struktury vykonává). Z nadefinovaných procesů se spouštějí jejich instance. Instance procesu představuje reálnou činnost prováděnou na základě definice procesu. V jeden okamžik může v systému existovat několik souběžně běžících instancí jednoho procesu.

2.1 Businness Process Reengineering

Dle zdroje [2] je workflow často spojován s BPR (Businness Process Reengineering), který zajišťuje průzkum, analýzu podnikových procesů a modelaci těchto procesů. Tyto modely lze následně snadno implementovat ve workflow systémech. Cílem BPR je analyzovat existující procesy ve firmě a určit jejich efektivnost jak pro firmu, tak pro zákazníka.

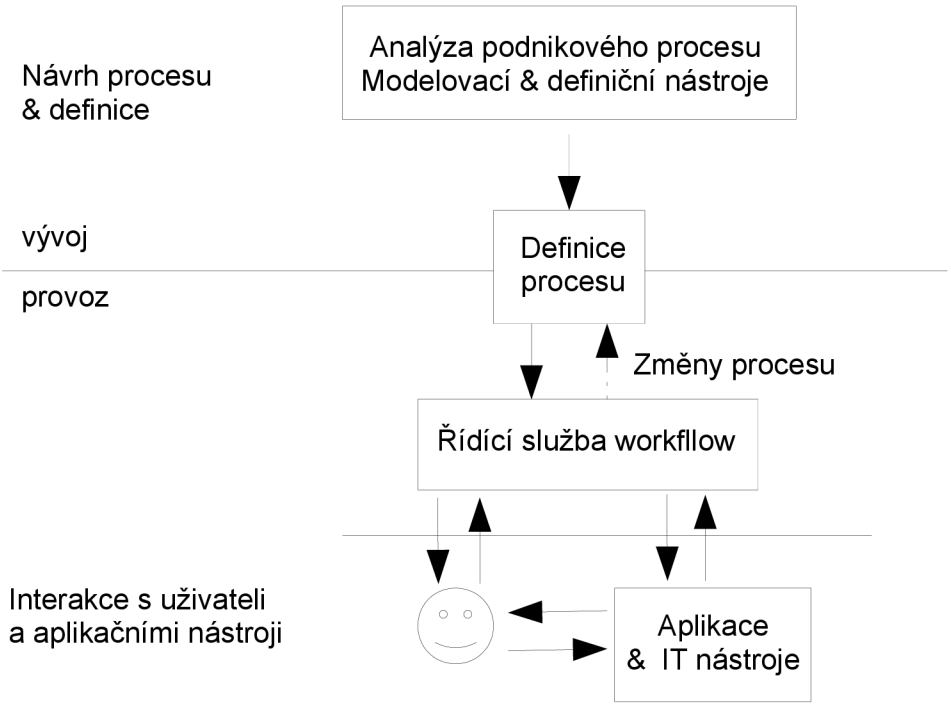
2.2 Důvody zavádění workflow

Dle zdroje [2] je hned několik důvodů, proč zavést systém typu workflow do firmy či organizace. Zavedením standardů pro jednotlivé postupy se výrazně zvyšuje efektivita práce a tím i snižují podnikové náklady. Při zavádění těchto standardů se klade důraz na zjednodušení podnikových procesů, zlepšení organizace a kvality práce. Velikou výhodou je uchovávání pracovních postupů v systému workflow. Tedy při personálních změnách se některé postupy neztrácejí s odchodem zaměstnanců a noví zaměstnanci mají ihned k dispozici hotové pracovní postupy. Další obrovskou

výhodou je, že vedení firmy získává díky workflow systému historii veškeré aktivity firmy a přehled o vykonané práci zaměstnanců.

2.3 Fáze a funkční oblasti workflow systému

Dle zdroje [3] lze na nejvyšší úrovni rozlišovat fázi vývoje a fázi provozu workflow systému. V těchto dvou fázích se nacházejí tři funkční oblastmi. První funkční oblastí v rámci vývojové fáze jsou funkce pro návrh a definici, které mají na starosti definování a modelování workflow procesů a úkolů. Během této operace proběhne překlad podnikového procesu z reálného světa na formální definici, kterou je možné zpracovávat počítačem. V druhé provozní fázi pak následují funkce pro řízení běhu a správu procesů, včetně kontroly řízení jednotlivých úkolů v rámci každého procesu. V této části probíhá celé dění interpretace procesů v rámci výkonného jádra. V třetí funkční oblasti probíhá provozní interakce s uživateli a dalšími IT aplikacemi pro zpracování jednotlivých kroků procesu. Během této části uživatelé zpracovávají informace a jednotlivé činnosti přidělené workflow systémem. Tyto oblasti jsou ilustrované na *obr. 1.* Více informací o modelování procesů a jejich životních cyklech lze získat ve zdroji [8] a [9].

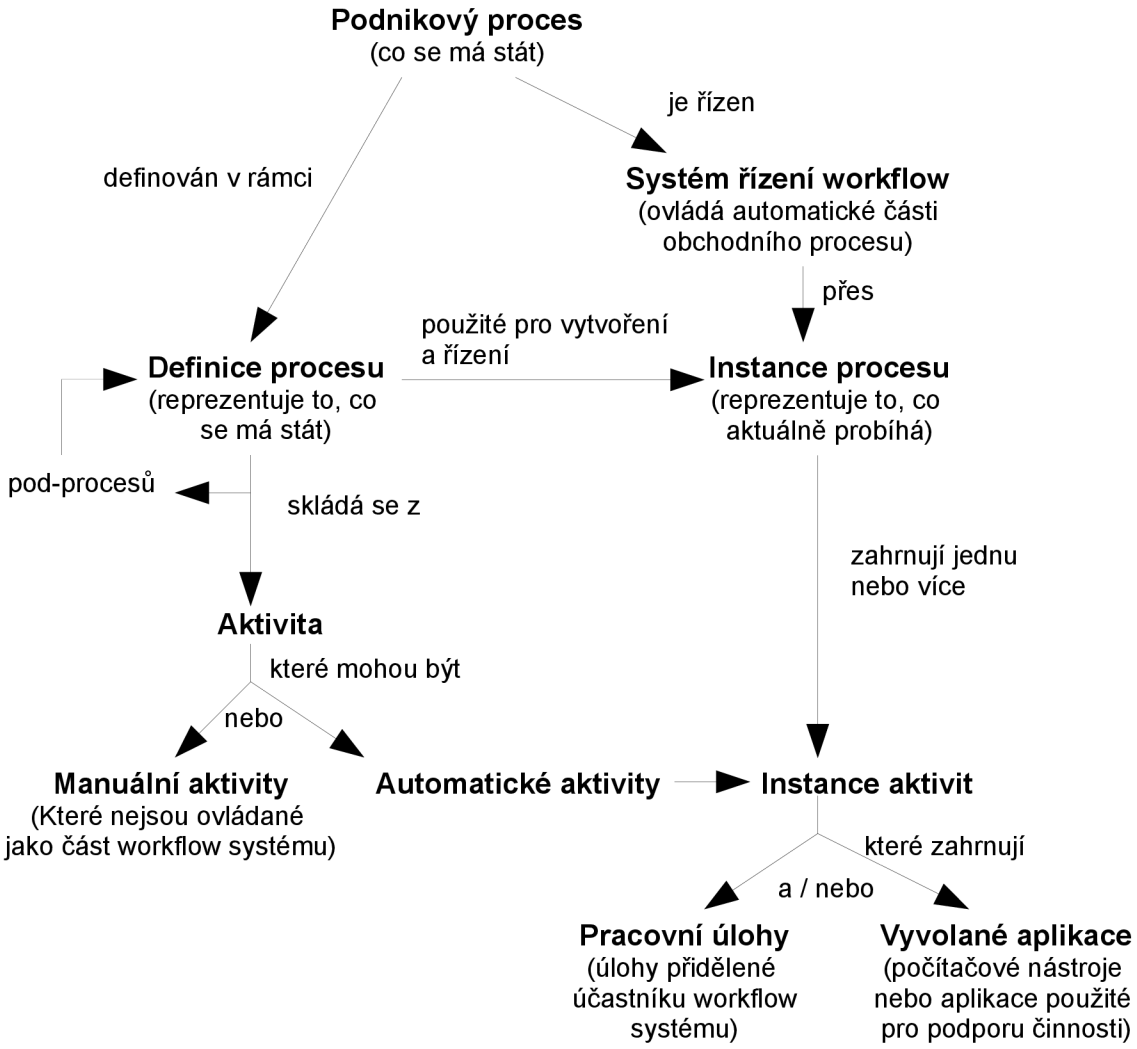


Obr. 1 : Charakteristika workflow systému

2.4 Terminologie workflow

Dle zdroje [1] z důvodu velkého počtu workflow produktů, které vznikaly v 90. letech minulého století, začal vznikat zmatek v terminologii. Proto bylo cílem organizace WfMC vytvořit slovník pojmů pro workflow systémy. V dnešní době ještě všichni dodavatelé workflow systémů nepoužívají

tento slovník, ale téměř ve většině případů je již využíván. Jednotlivé pojmy a vztahy mezi nimi jsou zanesené v *obr. 2.*



Obr. 2 : Vyjádření důležitých terminů

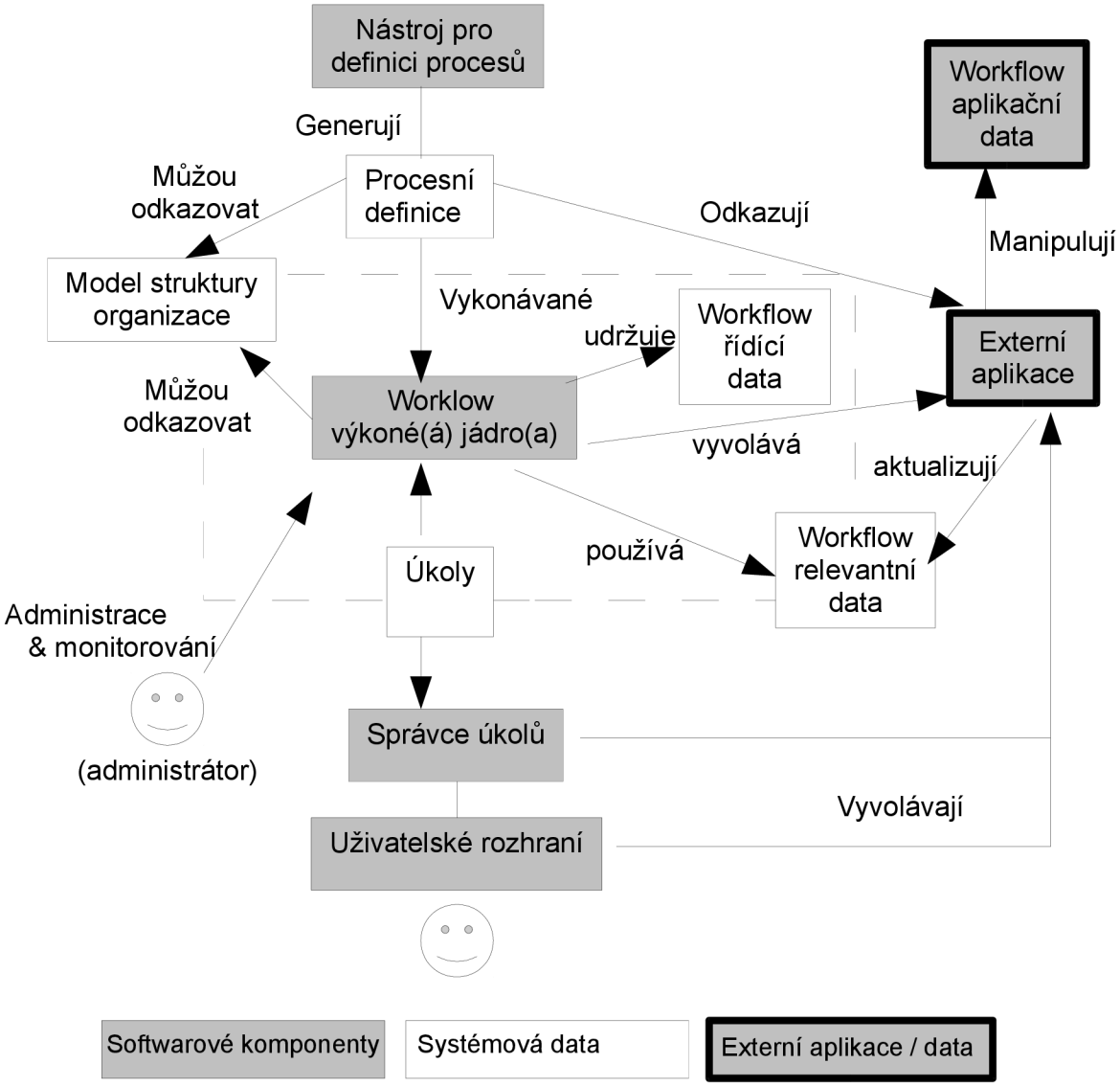
2.5 Funkční komponenty workflow systémů

Zdroj [3] uvádí, že navzdory velkého počtu workflow produktů se ukázalo, že je možné sestavit obecný model jednotlivých funkčních komponent workflow systému. Tento model představuje jednotlivé komponenty a rozhraní mezi těmito komponentami. Tyto funkční komponenty jsou shrnuté v *obr. 3.*

2.5.1 Nástroje pro definici procesů

Procesy je zapotřebí definovat tak, aby je bylo možné později počítačově automaticky zpracovávat. Takováto definice může být provedena buďto vytvořením speciálního jazyka nebo grafickou objektovou reprezentací a nastavením vazeb mezi těmito objekty. Kvalitní nástroje pro modelování

procesů v rámci workflow systému by také měli poskytnout uživateli možnost simulace průběhu pro vytvářenou definici procesu. Procesy mohou být také modelovány v produktu pro analyzování a modelování podnikových operací – za předpokladu kompatibility formátu pro přenos z/do workflow software.



Obr. 3 : Obecný model struktury workflow systému

Samotná definice procesu vyžaduje zadání a následné uložení všech nezbytných informací o procesu v takové podobě, aby mohl být proces zpracován výkonným jádrem. Nástroj tedy musí umožnit definovat a uložit informace o počátečních a koncových podmínkách pro jednotlivé úlohy a pravidla pro přechody mezi nimi.

2.5.2 Výkonné jádro workflow

Hlavní úlohou pro výkonné jádro je interpretovat procesní definice. Tyto definice jsou interpretované v rámci vytvořených instancí procesu. Instance procesu může být zpracována jedním

výkonným jádrem nebo distribuovaně – více jádry. V rámci průběhu instance procesu jsou generovány úlohy pro účastníky systému. Jádro musí tyto aktivity časově plánovat a hlídat jejich termíny plnění. K najití vhodného účastníka využívá výkonné jádro nejčastěji model organizační struktury organizace. Účastníkem systému nemusí být pouze člověk – tedy pro zpracování některých úkolů může být také využita externí aplikace. Workflow jádro si uchovává řídící data, která obsahují interní stavové informace pro jednotlivé instance a jejich aktivity. Výkonné jádro dále využívá relevantní data pro řízení průběhu instance procesu skrz jednotlivé úlohy. Aplikační data výkonnému jádru workflow nejsou přímo přístupná. S těmito daty mohou přímo manipulovat pouze externě vyvolané aplikace.

2.5.3 Úkoly & správce úkolů & uživatelské rozhraní

V případě potřeby interakce uživatele pro pokračování procesu umístí výkonné jádro záznam do seznamu úkolů. O tyto záznamy v seznamu úkolů se stará správce úkolů, který má na starosti interakce mezi výkonným jádrem a účastníky systému. Správce úkolů je tedy odpovědný za pokrok jednotlivých úkolů, které vyžadují pozornost uživatelů. Celý seznam úkolů může být v některých systémech pro uživatele neviditelný a další úkol se dozví až po zpracování předchozího. Některé systémy zobrazují i kompletní seznam úkolů a je zcela na uživateli, který úkol si ze seznamu vybere a bude zpracovávat.

Uživatelské rozhraní, které má úlohu intuitivního ovládání a příjemného vzhledu, může být sloučené přímo se správcem úkolů. Toto rozdělení v obecném schématu je z důvodu případného napojení uživatelského rozhraní na více workflow systémů a sloučení jednotlivých správců úkolů do jednoho uživatelského rozhraní. Pokud uživatel musí úkol splnit za použití lokální aplikace, může být správcem úkolů automaticky spuštěna, nebo si ji uživatel sám spustí z uživatelského rozhraní.

2.5.4 Administrace a monitorování

Pro účely nastavení celého systému je zapotřebí možnost administrátorského přístupu. Administrátor pro základní fungování systému musí zavádět do systému jednotlivé uživatele nebo počítačové stanice. Administrátor může jednotlivé uživatele privilegovat k provádění dalších administrativních úkonů. Administrátor dále definuje jednotlivé dostupné zdroje a nastavuje jejich účel. Administrátor má přístup k procesům a může provádět jejich změny, přerozdělovat alokované zdroje. V rámci monitoringu může administrátor například sledovat průběh jednotlivých instancí procesů a kontrolovat dodržování termínů pro jednotlivé úkoly ze strany uživatelů.

2.6 Typy workflow systémů

Dle zdroje [2] rozlišujeme čtyři typy workflow systémů na základě podnikové hodnoty (podpůrný proces až rozhodující proces) a intenzity opakování procesu. Jedná se o systém produkční, administrativní, kolaborativní a ad-hoc.

2.6.1 Administrativní workflow

Jedná se o velmi často opakované podpůrné procesy pro vyřizování běžné každodenní činnosti. Tyto procesy bývají velmi jednoduché, bez velkého množství alternativních možností. Příkladem může být registrace vozidla, vytvoření objednávky či vyřízení reklamace.

2.6.2 Ad-hoc workflow

Jedná se o nahodile vzniklé procesy, které nejsou předem definované a standardizované. Definice těchto procesů vzniká až v momentu vzniku takovéhoto procesu. Tyto procesy jsou velmi podobné administrativním, ale zpracovávají unikátní situace, jako je například zpracování odpovědí dotazů, zpracování nestandardní reklamace apod. Pro zpracování těchto procesů je vyžadována od workflow produktu snadná definice nových procesů.

2.6.3 Kolaborativní workflow

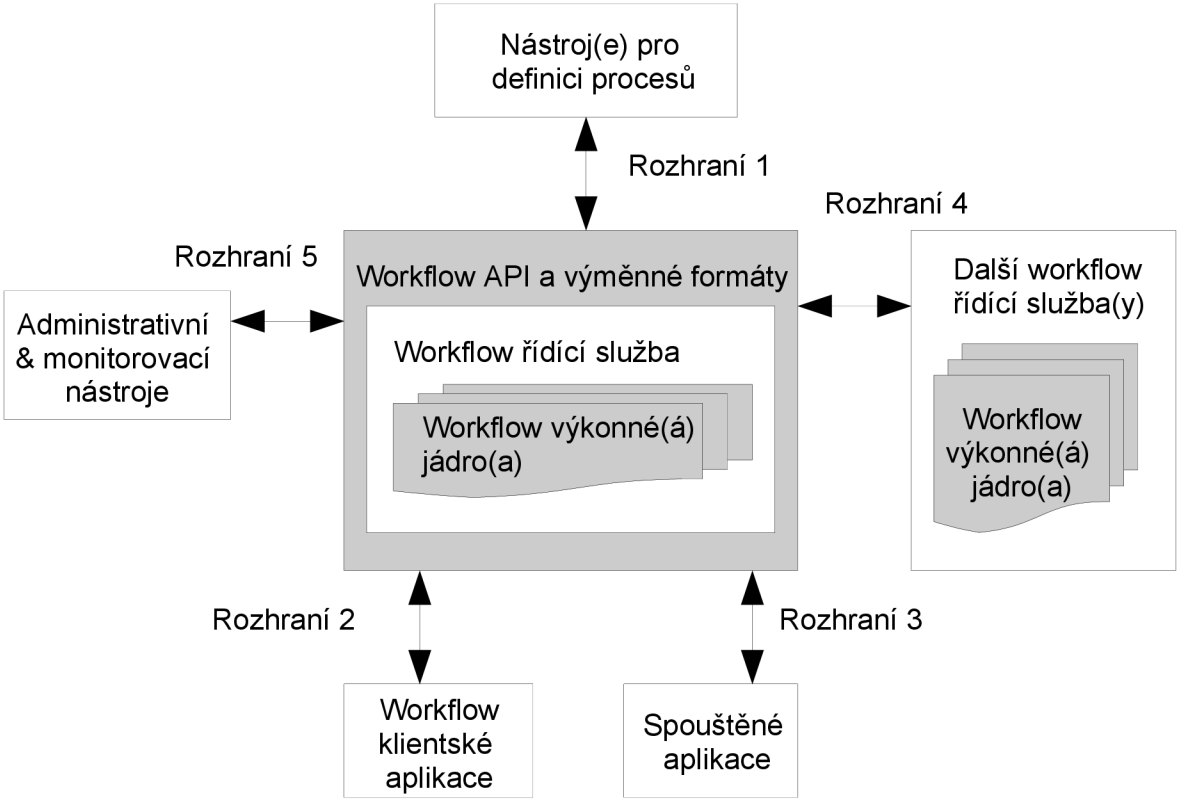
Kolaborativní proces obvykle obsahuje společnou opakovanou činnost mezi více účastníky, která probíhá do doby společného souhlasu. Účastníci si v průběhu tohoto procesu vyměňují dokumenty a poznatky. Příkladem může být tvorba smlouvy, tvorba reklamních materiálu či návrh vzhledu produktu. Tyto procesy jsou dynamické z principu postupné práce na daném procesu.

2.6.4 Produkční workflow

Produkční procesy popisují hlavní podnikové činnosti. Tyto procesy bývají velmi dobře strukturovatelné. Některé struktury mohou být velmi obsáhlé a mohou obsahovat všechny možné alternativní postupy. Zaměstnanci zpracovávají činnosti z těchto procesů na základě jejich profese v rámci organizační struktury. Tyto procesy také zabírají nejvíce času z pracovní doby oproti ostatním typům. Procesy nejsou měněny ze strany uživatelů, ale ze strany specialistů. Změny nebývají časté, a pokud se změny provádějí, tak to většinou souvisí s rozsáhlejšími změnami.

3 Workflow referenční model

Dle zdroje [3] byl Workflow referenční model vytvořen z obecné struktury jednotlivých workflow aplikací. Tento model má na starosti standardizaci komunikace jednotlivých workflow systémů s jejich okolím. Pro dosažení kompatibility mezi workflow produkty muselo být definované rozhraní a formáty zpráv pro výměnu dat. Rozhraní, která tento model popisuje, jsou vyobrazena na *obr. 4*. Rozhraní mezi jednotlivými komponentami a workflow řídící službou se nazývá WAPI (**W**orkflow **API**s and **I**nterchange formats).

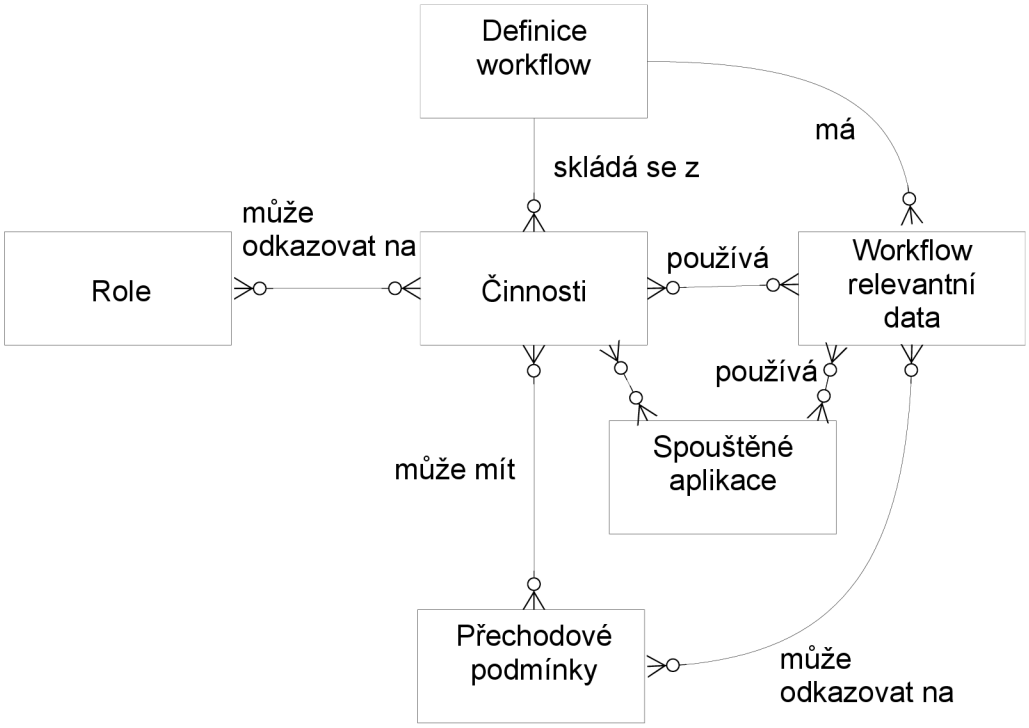


Obr. 4 : Workflow Referenční model

3.1 Nástroje pro definici procesů – rozhraní 1

Díky definovanému API rozhraní je v rámci funkcionality WAPI umožněno rozdělení modelovacích nástrojů pro procesy od prostředí pro interpretaci procesů. Toto dovoluje uživateli si vybrat rozdílné modelovací nástroje a rozdílné workflow interprety procesů, které mají standardizované rozhraní. Díky tomuto rozhraní můžeme výstup z definice jednoho modelovacího nástroje využít ve více workflow produktech.

WfMC definovala základní meta model pro definici procesu, který obsahuje základní objekty procesu a vazby mezi nimi. Jedná se o typické části, které by neměli chybět v definici procesu samotného. Tento model je vyobrazen na *obr. 5*.



Obr. 5 : Základní meta model definice procesu

WfMC předpokládá, že vlastnosti těchto objektů budou specifikovány jako definice workflow (název procesu, číslo verze procesu, podmínky pro start a ukončení procesu, bezpečnostní a kontrolní data), úkoly (název činnosti/úlohy, typ činnosti, vstupní a výstupní podmínky, další plánovací omezení), přechodové podmínky (průchodové podmínky, podmínka vykonání), workflow relevantní data (název dat a cesta, datové typy), role (jméno a organizační jednotka), spouštěné aplikace (obecný typ nebo název, spouštěcí parametry, umístění nebo přístupová cesta).

3.2 Workflow klientská aplikace – rozhraní 2

Seznam úkolů pro zpracování je část workflow produktu, která obsahuje činnosti vyžadující zapojení lidských zdrojů. Seznam úkolů vytváří frontu úkolů pro patřičné uživatele či pro skupiny uživatelů. K seznamu úkolu je v rámci dostupného API možnost využít i dalších komunikačních nástrojů jako je například e-mail.

Funkcionalita API by měla obsahovat možnost vytvoření relace (připojení / odpojení relace mezi zúčastněnými systémy), operace s definicemi procesů (vyhledávací / dotazovací funkce s možnostmi kritérií pro výběr podle názvu definic nebo atributů procesů), řídící funkce procesu (vytváření / spouštění / přerušení procesních instancí, pozastavení / obnovení procesních instancí, provedení změny stavu procesní instance nebo úkolu procesu, přiřazení nebo zjištění atributů procesu, procesní instance nebo úkolu procesu), stavové funkce procesu (otevření / zavření procesu nebo úkolu, nastavování volitelných kritérií pro filtrování, získávání detailů o instancích procesů nebo úkolech, filtrování informací, získání informací o konkrétním procesu nebo úkolu), seznám úkolů /

funkce pro práci se seznamem úkolů (otevření / zavření dotazu pro seznam úkolů, nastavení volitelných kritérií pro filtrování, zjištění položek seznamu úkolů, filtrování informací, oznámení o výběru / přeřazení úkolu / dokončení konkrétního úkolu, nastavení nebo dotázání se na atributy úkolu), funkce pro administraci (změna provozního stavu definice procesu nebo jeho existující instance, změna stavu všech procesů nebo úkolu instance daného specifikovaného typu, přižení atributů všem procesům nebo úkolu instance specifikovaného typu, přerušení všech instancí), funkce na manipulování s daty (načtení / vrácení relevantních či aplikačních dat), spouštění externích aplikací (spouštění externích aplikací z uživatelského rozhraní, jako je například e-mailový klient, webový prohlížeč, editor dokumentů apod.).

3.3 Spouštěné aplikace – rozhraní 3

Workflow systém provádí interakce s různými aplikacemi v rámci zpracování procesní definice. Je ale nereálné vytvořit rozhraní s podporou pro všechny existující aplikace. Z tohoto důvodu bylo specifikované rozhraní, které mohou využívat speciální aplikace přizpůsobené na spolupráci s workflow systémy. Díky tomuto rozhraní lze vytvořit tzv. Agenty. Agent vytváří komunikační bránu mezi workflow systémem a aplikací, pro kterou je agent určen.

Funkcionalita API by měla obsahovat možnost vytvoření relace (připojení / odpojení relace s aplikací nebo agentem), řídící funkce aktivity (výkonné jádro → externí aplikace: spuštění aktivity, pozastavení / obnovení / přerušení aktivity – kde má aplikace dostupné asynchronní rozhraní pro komunikaci, externí aplikace → výkonné jádro: notifikace o dokončení aktivity, signalizace události, zjištění atributu aktivity), funkce pro práci s daty (předání workflow relevantních dat – před aktivitou do aplikace, po aktivitě z aplikace, předání aplikačních dat nebo adresy dat).

3.4 Spolupráce workflow systémů – rozhraní 4

Workflow produkty mají různá zaměření. Některé produkty se mohou zaměřovat na ad-hoc procesy a některé na propracované produkční systémy pro řízení výroby. WfMC navrhla několik scénářů kompatibility a to od procházení jednoduchých úkolů po plnou výměnu definic workflow procesů a dat. Z důvodu velkého množství workflow produktů na trhu se nepředpokládá plná podpora kompatibility pro složité scénáře, avšak předpokládá se podpora pro jednoduché scénáře.

3.4.1 Zřetězení

Tento scénář představuje spojení dvou procesů mezi dvěma rozdílnými workflow systémy, které pracují nezávisle na sobě. Jakmile proběhne proces A, proběhne přesun aktivní instance do procesu B bez další pozdější synchronizace zpět do procesu A.

3.4.2 Hierarchické (vnořený podproces)

Tento scénář dovoluje hierarchické předání řízení aktivní instance pro podřízený podproces, který bude běžet v jiném workflow prostředí. Limit pro hierarchické zanoření do podprocesů není stanoven. Jakmile podproces dokončí svojí činnost, tak se řízení aktivní instance přesouvá zpět do procesu nadřazeného, který daný podproces vyvolal.

3.4.3 Souvislé spojení (Peer-to-Peer)

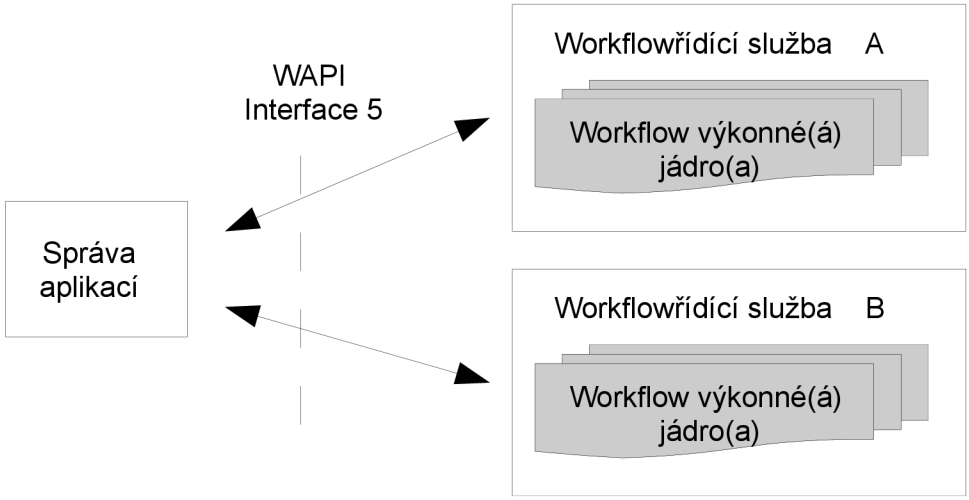
Tento scénář představuje volné přecházení zpracovávání jednotlivých úkolů mezi rozdílnými workflow systémy. Každá úloha takto může být zpracována na jiném workflow jádru a to zcela automaticky bez jakéhokoliv přičinění uživatele nebo administrátora. Tento model tedy vyžaduje propracovanou komunikaci mezi jednotlivými jádry za pomoci specifikovaného API rozhraní.

3.4.4 Paralelní synchronizace

Jedná se o navzájem nezávislé procesy, které mají pouze předem definované synchronizační body. Při dosažení synchronizačního bodu v rámci aktivních instancí dochází ke spouštění příslušných sekvencí.

3.5 Administrativní a monitorovací nástroje – rozhraní 5

Výhoda, kterou přinese specifikované rozhraní pro administraci a monitoring je, že jeden monitorovací nástroj bude možné použít na správu více workflow systémů od různých výrobců. Ilustrace je vyobrazena na *obr. 6.*



Obr. 6 : Rozhraní pro administrativní a monitorovací nástroje

Funkcionalita API by měla obsahovat operace pro správu uživatelů (vytvoření / odstranění / deaktivování / změna oprávnění uživatelů nebo pracovních skupin), operace pro správu uživatelských

rolí (vytvoření / odstranění / změna vztahu mezi rolemi a uživateli, nastavování a rušení vlastností jednotlivých rolí), operace pro audit (dotaz / tisk / start nového / vymazání auditu nebo logu událostí), operace pro kontrolu zdrojů (nastavení / zrušení / upravení zdrojů v procesech a úlohách, nastavení dat o systémových zdrojích, jejich počty a parametry využití), funkce pro dohled nad procesy (měnění operačního stavu workflow definice procesu nebo existující procesní instance, povolení nebo zakázání určitých verzí definice procesu, změna stavu pro všechny procesy nebo aktivní instance specifického typu, nastavení atributů pro všechny procesy nebo aktivní instance specifického typu, přerušení všech instancí procesu), funkce stavu procesu (otevření / zavření procesu nebo úkolu, nastavovaní volitelných kritérií pro filtrování, získávání detailů o instancích procesů nebo úkolech, filtrování informací, získání informací o konkrétním procesu nebo úkolu).

4 Implementace - použité technologie

4.1 PHP

Dle zdroje [4] počátky PHP začali kolem roku 1995, kdy Rasmus Lerddorf vyvinul Perl/CGI skript, který mu umožňoval on-line zjistit počet čtenářů. Návštěvníci jeho stránek se začali dotazovat na tuto funkcionalitu a z toho důvodu dal k dispozici svoji sadu nástrojů pod názvem PHP (Personal Home Page). Postupně začal vydávat další rozšíření funkcionality, které již vyvíjel v jazyce C. Tyto změny byly vydány v roce 1997 pod označením PHP 2.0. Na této verzi se podílelo již mnoho programátorů z celého světa. V roce 1997 došlo ke změně významu zkratky PHP, která až do dnes nyní znamená Hypertext Preprocesor. Díky vysoké popularitě jsme se již dočkali u tohoto skriptovacího jazyka verze PHP 5. Největší výhodou tohoto skriptovacího jazyka je, že se jedná o open-source, takže zde nejsou žádné omezení pro používání, modifikaci a redistribuci.

Tuto platformu jsem zvolil z důvodu, že web je dostupný pro všechny uživatele – tedy samotné rozhraní pro takovýto workflow systém bude dostupný všem účastníkům bez nutnosti instalovat a provozovat speciální aplikaci. Výkonné jádro workflow je napsané jako php skript spouštěný ze shellové konzole serveru, který běží asynchronně vůči uživatelskému rozhraní na pozadí. Díky tomu, že webové rozhraní i výkonné jádro je takto napsané ve stejném jazyce, mohou využívat společné implementace některých části v rámci společných tříd. Příkladem může být třída procesu, která umožňuje vstup procesu z XML uloženého v databázi či z odeslaného modelačního nástroje. Tato třída provádí validaci, interpretaci v rámci instance či výstup pro modelační nástroj či do XML formátu.

4.2 Apache

Dle zdroje [5] je Apache bezpochyby nejoblíbenějším webovým serverem, mj. díky tomu, že je zdarma a poskytuje podporou pro více operačních systémů. Díky tomu, že je šířen kromě binárních souborů i v podobě zdrojových kódů, lze jej po kompilaci provozovat i na různých procesorových architekturách. Apache je snadno rozšiřitelný pomocí dalších modulů, které jednak mohou být součástí distribuce nebo je lze také získat od dalších výrobců. Tento webový server je velmi vhodný pro generování dynamických stránek jak za pomocí CGI-skriptů, tak za pomocí dalších modulů pro skriptovací jazyky.

Apache jsem použil pro přístup uživatelů k webové aplikaci uživatelského rozhraní, která je napsaná ve skriptovacím jazyce PHP.

4.3 MySQL

Dle zdroje [4] MySQL vzniklo původně jako interní firemní projekt. Poprvé se tento projekt pro veřejnost dostal na svět v roce 1995. Z důvodu obrovské popularity jeho autoři velmi rychle odhalili jeho potenciál a založili ve Švédsku firmu MySQL AB. Již od počátku kladli vývojáři důraz na výkon a škálovatelnost. Z tohoto důvodu vznikl velmi optimalizovaný produkt. Z počátku postrádal funkcionalitu jako jsou uložené procedury, triggery a transakce. Funkcionalita postupem času narostla. MySQL je šířen jednak pod GPL licencí, tak pod komerční licenci.

Nejpoužívanějším typem tabulek je MyISAM, který je i nastaven jako výchozí. Tento typ sice nepodporuje transakce, ale je velmi vhodný pro tabulky s intenzivními dotazy pro výběr dat (selecty) a to i v rámci objemných dat. Dále je tento typ tabulek velmi vhodný i pro intenzivní vkládání dat. V případě potřeby zpracovávaní transakcí je k dispozici typ tabulek pod názvem InnoDB. Tento typ tabulek je vhodný pro tabulky s intenzivními aktualizacemi dat, transakční zpracování a automatickou obnovu po havárii. Tento typ tabulek ale například přímo nepodporuje fulltextové vyhledávání. V případě potřeby rychlosti se nabízí typ tabulek HEAP. Tento typ má úložiště dat v paměti a díky tomuto dosahuje i obrovské rychlosti. Nevýhodou naopak je, že při havárii systému přijdeme o data v těchto tabulkách.

4.4 JavaScript

Dle zdroje [6] se vznik JavaScriptu datuje pro rok 1995. JavaScript byl vytvořen firmou Netscape Communications Corporation. JavaScript nemá s programovacím jazykem Java nic společného, takže se předpokládá, že se při pojmenování pouze inspirovali názvem a to z důvodu popularity jazyku Java. JavaScript přinesl do webových stránek možnost doprogramovat nejrůznější efekty, reakce na události myši a klávesnice. Microsoft implementoval obdobu tohoto jazyka pod názvem JScript do Internet Exploreru 3.0. Ačkoliv JScript vypadal stejně jako JavaScript, tak některé skripty byly navzájem nekompatibilní. Proto se obě firmy obrátili na organizaci ECMA (European Computer Manufacturers Association), která vytvořila standardizovaný jazyk pod názvem ECMAScript, pro který se však stále používá název JavaScript.

4.5 Další rozšíření

Kromě výše uvedených technologii jsem pro implementaci využil ještě následující frameworky, třídy a rozšíření funkcionality PHP a JavaScript.

4.5.1 Smarty (PHP)

Jedná se o velmi užitečný opensource šablonovací nástroj, jehož cílem je oddělit logiku aplikačního kódu stránky od samotné prezentace uživateli. Jedná se doplnění k PHP, ne jeho nahrazení. Smarty je šířen jako opensource pod LGPL licencí a je tedy možné jej využít pro komerční účely. Do samotného PHP se toto rozšíření načítá jako třída, pomocí které lze do šablony předat veškeré potřebné proměnné, které budou použity při roz-generování šablony.

4.5.2 XML_Unserializer (PHP)

Jedná se o PHP balíček, který převádí XML dokumenty na pole. Tato třída je šířena pod BSD licencí. Tuto třídu používám pro načtení informací z XML reprezentace procesů a záznamů o jednotlivých instancích procesů.

4.5.3 JavaScript vector-draw library (JavaScript)

Jedná se o velmi užitečnou JavaScriptovou knihovnu, která umožňuje kreslení obrazců do webové stránky. Tato knihovna podporuje obrazce, jako jsou čáry, kruhy, elipsy, obdélníky, mnohoúhelníky. Tato knihovna řeší podporu rozdílností canvasů jednotlivých typů prohlížečů a je šířená pod LGPL licencí. Tato knihovna je využita pro vykreslení šipky, při kliknutí na úkol, který bude následovat při splnění podmínky.

4.5.4 YUI 2 (JavaScript)

Jedná se o JavaScriptový framework s velmi bohatou funkcionalitou. Tento framework je šířen pod BSD licencí. Pro potřeby tohoto rozhraní je využita funkcionalita pro otevření dialogového okénka obsahující další funkcionalitu aplikace v rámci definice procesů.

4.5.5 PHPMailer

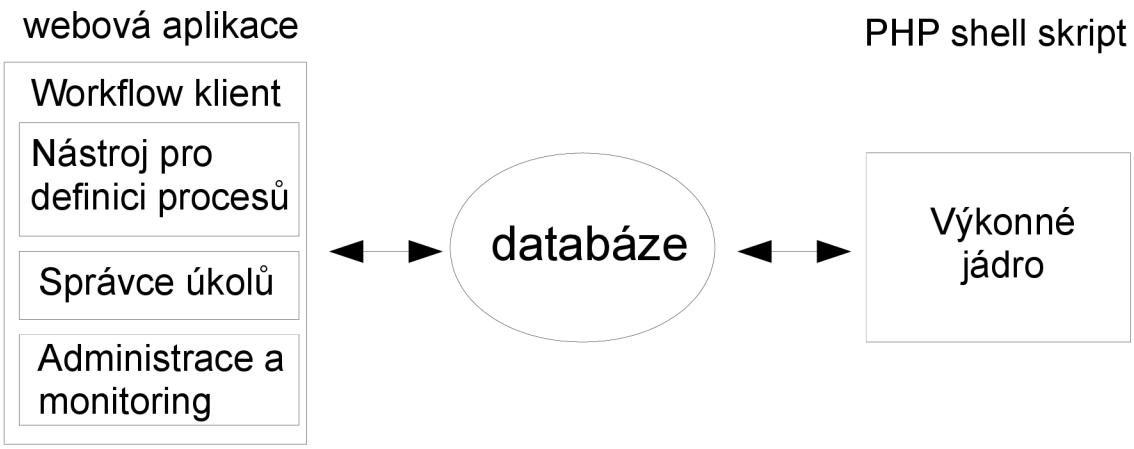
Užitečná třída na zasílání mailu z PHP, která je šířena pod licencí LGPL. Tato třída umožňuje rozšíření případné funkcionality oproti běžné PHP funkci mail(). Tato třída umožňuje zasílání přes redundantní SMTP servery, automatické zalamování textů, snadné zasílání HTML mailů s automaticky vytvořenými alternativními texty pro klienty, které HTML mail přímo nepodporují.

5 Návrh a struktura systému

Fungování celého systému je rozdělené na 3 logické celky, *viz obr. 7*. Jako uživatelské rozhraní slouží webová aplikace, která zpřístupňuje nástroj pro definování procesů, administrativní funkce pro definice uživatelů, zdrojů a monitoring systému. Dále tato aplikace obsahuje rozhraní správce úkolů pro samotné účastníky systému.

Tato webová aplikace se připojuje k databázi MySQL, která uchovává veškerá data v tabulkách. Definice jednotlivých procesů jsou ukládané v XML struktuře rovněž v databázi. Log průběhu jednotlivých instancí procesu je uchováván také v databázi jako XML.

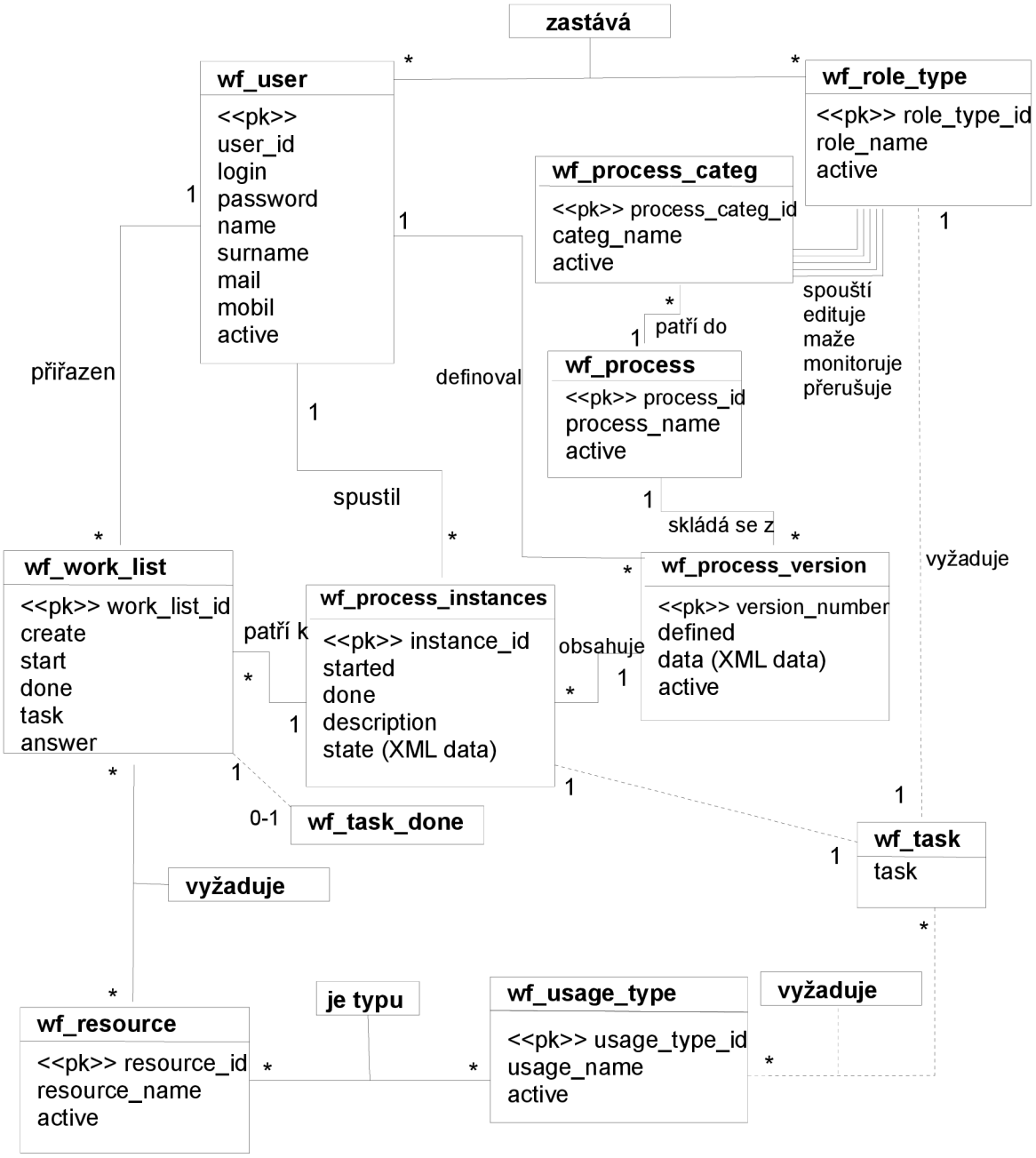
K MySQL databázi se připojuje také PHP shellový skript s implementací výkonného jádra, který zajišťuje chod celého systému. Výkonné jádro v sobě udržuje a řídí všechny aktivní instance procesů. Stará se o přidělování jednotlivých úkolů pro účastníky systému a k těmto úkolům také alokuje potřebné zdroje.



Obr. 7 : Schéma konceptu systému

5.1 Ukládání dat

Data pro celou aplikaci jsou umístěna v celkem 15 MySQL tabulkách. Tabulky mají nastavené uložiště dat typu InnoDB, které podporuje cizí klíče a kontroluje integritní omezení. Toto uložiště také dovoluje výkonnému jádru použít technologii transakčního zpracování. V rámci transakce proběhne načtení a zpracování instrukce nad XML uloženými daty. Jako poslední krok transakce po uloženích všech upravených dat je instrukce vymazána a transakce ukončena. Model databáze se nachází v ER diagramu na *obr. 8*.



Obr. 8 : ER diagram systému

5.2 Funkce jednotlivých tabulek

- **wf_user** – tabulka obsahující jednotlivé uživatele systému a informace o těchto uživatelích. Uživatelé v rámci spojovací tabulky **wf_user_role** získávají možnost získat až několik rolí, které mohou v rámci systému zastávat.
- **wf_role_type** – tato tabulka obsahuje jednotlivé role, které mohou uživatelé získat. Systém obsahuje jednu defaultní roli a to administrátorskou. Další role jsou již definované administrátorem a tyto role mohou umožňovat další funkcionalitu, jako je například spuštění

instancí procesů, definice procesů atd. Druhým účelem role je evidence profesních označení uživatelů pro účely přidělování jednotlivých úkolů v rámci procesu.

- **wf_usage_type** – Tato tabulka uchovává jednotlivé definované typy užití zdrojů. Podle těchto typů jsou pak vybírané vhodné zdroje v rámci jednotlivých úkolů
- **wf_resource** – Tato tabulka uchovává jednotlivé konkrétní zdroje, které jsou v rámci systému k dispozici. Každý jednotlivý předmět má zde svůj záznam a může spadat pod více typů užití v rámci pomocné spojovací tabulky **wf_resource_usage**
- **wf_process_categ** – tato tabulka zajišťuje definování kategorií pro procesy. Ke každé této kategorie lze stanovit: uživatelské role, které mohou spouštět instance procesů z dané kategorie, uživatelské role, které mohou definovat a editovat jednotlivé procesní definice. Dále je možné definovat roli pro mazání hotových verzí procesu, možnost monitoringu průběhu jednotlivých instancí a možnost přerušení aktivní instance a uvolnění všech alokovaných zdrojů.
- **wf_process** – tabulka uchovávající názvy jednotlivých procesů. Samotné verze jsou uchovávané v další tabulce a to **wf_process_version**. Každá tato verze procesu zaznamenává kdo a kdy danou verzi definoval a samotná definice je uložena v databázi v XML struktuře.
- **wf_process_instances** – tato tabulka v sobě uchovává informace pro jednotlivé instance procesů. Eviduje informace jako je čas spuštění a dokončení instance, její popis a kdo instanci spustil. Samotný průběh instance pro účely monitoringu si výkonné jádro ukládá do sloupce state v XML struktuře.
- **wf_task** – jedná se o tabulku, která uchovává dočasné informace. Může se jednat například o příkaz pro výkonné jádro, aby si načetlo mezi své udržované instance nově vzniklou instanci. Či si zde výkonné jádro uchovává úkoly, kterým je zapotřebí přidělit zdroje. Tato tabulka tedy potřebuje odkazovat na instanci, uživatelskou roli systému a kategorie zdrojů, které jsou zapotřebí. Tyto vyžadované zdroje jsou připojené pomocí spojovací tabulky **wf_task_required_resource_type**.
- **wf_work_list** – tato tabulka obsahuje úkoly přidělené konkrétním uživatelům systému. Každý úkol obsahuje také informaci o jakou úlohu a instanci se jedná, kdy byl úkol vytvořen a dokončen. Dále zde má každá úloha přidělené konkrétní zdroje za pomocí spojovací tabulky **wf_work_list_resource**. Reakce na každou úlohu se ukládá do položky answer v XML struktuře a pozornost výkonného jádra na změnu v této tabulce se získá pomocí pomocné ukazující tabulky **wf_task_done**. Díky této tabulce se jádro dotazuje tabulky, kde buďto není momentálně žádný záznam nebo je zde informace o minimálním počtu dokončených úkolů. Výkonné jádro se nemusí takto dotazovat neustále dokola do veliké tabulky s historií všech úkolů.

5.3 XML definice procesů

Definice procesu vytvořená v uživatelském rozhraní se ukládá do databáze jako XML struktura. S touto strukturou následně pracuje výkonné jádro. V PHP využívám třídy XML_Unserializer, která převede XML do stromové struktury uložené do pole. Ukázka takové XML definice je v *kódu 1*.

Celá definice je v kořenovém elementu *process*. V rámci této definice procesu mohou být definované proměnné v elementu *declare_variable*, který obsahuje pro každou proměnou element *variable*. Element *variable* má povinné atributy *name* pro název proměnné a *type* pro definici datového typu proměnné. Definice proměnné má dále nepovinné atributy *default* pro defaulní hodnotu proměnné po její deklaraci. V případě datového typu *int* může proměnná zaobalovat elementy *option, který* obsahuje číselníkové hodnoty. Tento element má dva povinné atributy a to *value* s číselnou hodnotou a *name* s významovou reprezentaci pro danou hodnotu.

Validní definice procesu musí dále obsahovat alespoň jeden element *task*, který zastupuje definici jednoho konkrétního úkolu v rámci definice procesu. Povinnými atributy úlohy je *index*, který označuje pořadí úkolu a označení místa pro případný cíl skoku po vyhodnocení podmínky. Dalším atributem je *profession*, který určuje požadovanou organizační roli pro najití vhodného účastníka systému na vykonání úkolu. Posledním atributem je *name*, který obsahuje popisné pojmenování daného úkolu. Definice úkolu může zapouzdřit další zanořené elementy nastavení dalších specifikací pro úkol. Po každý požadovaný typ užití zdroje je zde připraven element *resource_type*, který v atributu *value* označuje id požadovaného typu užití zdroje. Pro zobrazení aktuálního stavu hodnoty proměnné v uživatelském rozhraní slouží element *variable_notify* s parametrem *name,* který nabývá názvu proměnné k zobrazení. Definování položek odpovědního formuláře se nachází v elementu *answer_form*. Jednotlivé proměnné, které uživatel bude nastavovat v rámci tohoto formuláře, se zapisují do zanořených elementů *variable*, kde každý tento element má jediný atribut *name* pro uvedení názvu proměnné. Posledním možným elementem je element *condition*, který je pro zapsání podmínky přechodu na jinou než následující úlohu. Atributem takovéhoto elementu je *go_task*, které slouží k určení úlohy, která bude následovat při splnění podmínky. V hodnotě samotného elementu je zapsaná podmínka obsahující podmínkový výraz. Na levé straně výrazu je vždy proměnná, za kterou následuje logický operátor. Na pravé straně výrazu je vždy hodnota, která musí odpovídat datového typu proměnné. V případě podmínky obsahující více porovnání, tak jsou jednotlivé části podmínky spojené operátorem *&&*. Takováto složená podmínka je platná při splnění všech jednotlivých částí.

```
<process>
	<declare_variable>
		<variable name="poznamka" type="text"></variable>
		<variable name="barevne" type="bool" default="true" />
		<variable name="mnozstvi" type="int"></variable>
		<variable name="vyhotoveno" type="bool" default="false" />
		<variable name="stav" type="int">
			<option value="1" name="OK" />
			<option value="2" name="Chyba v analýze" />
			<option value="3" name="Chyba v grafickém návrhu" />
		</variable>
		<variable name="vytisknuto" type="bool" />
		// zde může pokračovat deklarace
	 </declare_variable>

	 <task index="1" profession="6" name="Přijmutí požadavků od zákazníka">
		<resource_type value="2" />
		<resource_type value="1" />
		<resource_type value="4" />
		<answer_form>
			<variable name="poznamka" />
			<variable name="barevne" />
			<variable name="mnozstvi" />p
		</answer_form>
	</task>
	// vynechaná jedna úloha
	<task index="3" profession="7" name="Představení návrhu zákazníkovi">
		<resource_type value="3" />
		<variable_notify name="poznamka" />
		<variable_notify name="barevne" />
		<answer_form>
			<variable name="poznamka" />
			<variable name="stav" />
		</answer_form>
		<condition go_task="1">
			stav = 1 &amp;amp;&amp;amp; mnozstvi &amp;lt; &amp;quot;100&amp;quot;
		</condition>
		<condition go_task="1">stav = 2</condition>
		<condition go_task="2">stav = 3</condition>
	 </task>
	// zde mohou být definované další úlohy
</process>
```

Kód 1 : Ukázka XML definice procesu

5.4 XML struktura logu instance procesu

Stejně jak definice procesu, tak log průběhu celé instance procesu je uložen v databázi jako XML struktura. Tato XML struktura postupem průběhu celého procesu narůstá o veškeré potřebné informace k monitoringu. Ukázka XML logu je v *kódu 2.*

Kořenovým elementem je *process_log*, který v případě existence proměnných zapouzdřuje v elementu *variables* aktuální (poslední) stav veškerých proměnných. Každá proměnná má svůj stav uložen v elementu *variable*, kde v atributu *name* se nachází název samotné proměnné. V případě textové proměnné je hodnota proměnné vložena do hodnoty daného elementu. V opačném případě se hodnota nachází v atributu *value*. V případě, že proměnná nemá defaultní hodnotu a nebyla doposud z uživatelského rozhraní nastavena, tak nabývá hodnoty *null*.

```
<process_log>
	<variables>
		<variable name="poznamka">textovy zapis od uzivatele</variable>
		<variable name="barevne" value="true" />
		<variable name="mnozstvi" value="100" />
		<variable name="vyhotoveno" value="false" />
		<variable name="stav" value="null" />
		<variable name="vytisknuto" value="false" />
	</variables>

	<task index="1" task_index="1" employee="1" create="2011-04-24 19:51:25"
		start="2011-04-24 19:51:47" done="2011-04-24 20:00:02">
		<used_resource resource_id="6" />
		<used_resource resource_id="5" />
		<used_resource resource_id="8" />
		<filled_form>
			<variable name="poznamka">
				textový zápis od uživatele
			</variable>
			<variable name="barevne" value="true" />
			<variable name="mnozstvi" value="100" />
		</filled_form>
	</task>

	<task index="2" task_index="2" employee="1" create="2011-04-24 20:00:03"
		start="2011-04-24 20:00:04">
		<used_resource resource_id="6" />
		<used_resource resource_id="2" />
		<used_resource resource_id="8" />
	</task>
</process_log>
```

Kód 2 : Ukázka logu instance procesu

Průběh jednotlivých úloh je zaznamenán do elementů *task*. Tyto elementy jsou indexované, tak jak probíhali v rámci atributu *index*. Samotné pořadové číslo úlohy z definice, které jednoznačně

označuje úkol v definici procesu, je zaznamenáno v atributu *task_index*. Id uživatele, kterému byl úkol přidělen, se nachází v atributu *employee*. Atributy *create, start, done* uchovávají datum a čas vytvoření, přidělení a dokončení daného úkolu. Přidělené konkrétní zdroje jsou zaznamenány ve vnořených atributech *used_resource*, které mají atribut *resource_id* zaznamenávající katalogové číslo zdroje. Po odeslání odpovědního formuláře v rámci splnění úkolu je vložen element *filed_form*, který obsahuje jednotlivé hodnoty pro proměnné odpovědního formulář. Každá proměnná je v zanořeném elementu *variable*, který má atribut *name* pro název proměnné a atribut *value* pro vyplněnou odpovědní hodnotu.

5.5 Výkonné jádro

Výkonné jádro běží asynchronně od uživatelského rozhraní jako shellový PHP skript. Výkonné jádro v sobě udržuje a řídí jednotlivé aktivní instance. Procesy, které mají tedy běžet v rámci workflow systému paralelně, tak jsou ve skutečnosti prováděny sekvenčně. Každá instance je v jádru jako inicializovaný objekt třídy *process*. Tento objekt v sobě zapouzdřuje data definic procesu a data stavu celého procesu spolu s metodami pro operace nad danou instancí. Na základě této definice procesu objekt mění svůj stav a udržuje i svůj průběh v podobě logu v databázi pro případné zotavení systému po jeho restartování. Výkonné jádro neustále v intervalu dvou vteřin nahlíží v databázi do dvou tabulek s instrukcemi. Podle těchto instrukcí volá příslušné metody pro jednotlivé objekty (instance procesů). Za pomocí tabulky *wf_task* získává jádro informaci buďto o nové instanci procesu, kterou je třeba vytvořit nebo informace o zdrojích na které se čeká. V momentě, kdy jsou zdroje volné, tak jádro zavolá patřičnou metodou objektu instance a informuje ji takto o tom. Výkonné jádro využívá technologie transakčního zpracování, kde se převádí databáze z jednoho konzistentního stavu workflow sytému do druhého, a to včetně uložených XML struktur. Toto transakční zpracování tedy zajistí, že v případě výpadku nebudou v systému částečně zpracovaná data. S aktualizací logu procesu v databázi je zároveň garantované odstranění patřičné instrukce pro provedení operace nad procesem a zavedení dalších pomocných dat.

Tento objekt procesu lze zkonstruovat i v rámci uživatelského rozhraní a pro danou instanci poskytuje metody pro zjištění průběhu a statistik za účelem monitoringu instance.

5.5.1 Generování úkolů

V rámci interpretace procesu jsou generovány úkoly, které jsou ukládány do databáze pro následné zpracování úkolů uživateli z uživatelského rozhraní. Plnění úkolů ze strany uživatelů a ukládání jejich reakce zpět do databáze zajišťuje postup instance ve výkonném jádru. Během tohoto postupu jsou generovány další nové úkoly tak dlouho, dokud se nedosáhne doběhnutí instance do jejího konce.

V uživatelském rozhraní je zadána reakce uživatelem zpět pro výkonné jádro. Reakci se rozumí potvrzení o splnění daného úkolu a případné doplnění hodnot pro vyžadované proměnné odpovědního formuláře z definice daného úkolu.

O každém nově vytvořeném úkolu je uživatel, kterému byl přidělen, obeznámen také e-mailem, který obsahuje základní informace o procesu, instanci samotné a patřičném úkolu.

5.5.2 Výběr uživatelů a přidělování zdrojů

Výkonné jádro musí při interpretaci jednotlivých instancí procesů přidělovat jednotlivým uživatelům zdroje. Volbu uživatele jsem zvolil jako náhodný výběr uživatele, který je zařazen do patřičné organizační role. Tento náhodný výběr jsem zvolil z důvodu rovnoměrného rozkládání úkolů na jednotlivé uživatele ze statistického hlediska, kdy je stejná pravděpodobnost, že daný uživatel obdrží úkol.

Přidělování samotných zdrojů je složitější proces, protože jedna úloha může požadovat zdroje s více účely užití. Každý samostatný zdroj může dále zastávat více užití. Z těchto důvodů je třeba vybírat zdroje efektivněji vzhledem k jejich užití. Celý algoritmus funguje ve výsledku velmi jednoduše. Skládá se z cyklu, který probíhá tak dlouho, dokud nepřidělí všechny požadované typy užití zdroje a dokud se mu podařilo v posledním kroku nějaký zdroj přidělit. Algoritmus se postupně snaží odebírat požadované typy užití zdroje a tím i přidělovat konkrétní zdroje. V rámci jednoho průběhu cyklu je vždy nalezen jeden aktuálně nepřidělený zdroj, který může odebrat co nejvíce požadovaných užití. Tento výběr zdroje probíhá na velmi jednoduchém principu načtení dat z databáze. V rámci dotazu proběhne kartézský součin zdrojů s jejich užitím a následné omezení dat na pouze požadované užití. Dále v rámci tohoto dotazu proběhne agregace na jednotlivé zdroje a jejich sestupné setřídění dle počtu agregovaných řádků. Takto je získán na první pozici výsledku zdroj, který obsahuje nejvyšší počet vyhovujících užití.

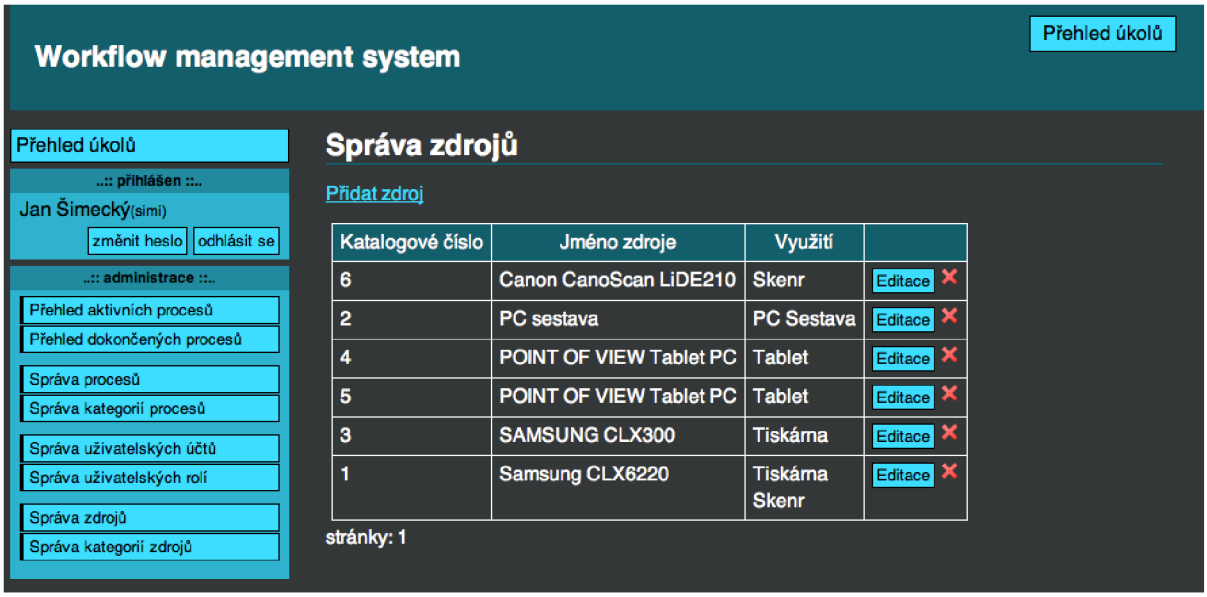
6 Části uživatelského rozhraní

6.1 Zdroje workflow systému

Během definic jednotlivých úkolů v procesech je nutné určit potřebné zdroje pro danou činnost. Tyto zdroje je zapotřebí přidělit, aby mohl být úkol s jejich pomocí splněn. Jako zdroj se dá představit nějaké nářadí, zařízení či speciální softwarový produkt pro provedení konkrétní činnosti.

Definici zdrojů jsem rozdělil do dvou podsekcí. V první podsekci probíhá definice kategorie zdrojů. Tato kategorie představuje jednotlivé možné účely užití zdroje (např. tiskárna, scanner).

V druhé části je možno definovat již konkrétní zdroje, které jsou k dispozici pro alokaci v rámci aktivních instancí procesu. Každý tento zdroj obdrží své unikátní identifikační katalogové číslo a také může být zařazen do více kategorií, tedy může plnit více účelů (např. Samsung CLX6220 bude spadat do kategorie tiskárna, scanner a kopírka). Ukázka ze systému, jak vypadá přehled nadefinovaných zdrojů, je v *obr. 9.*



Obr. 9 : Přehled zdrojů evidovaných v systému

6.2 Uživatelské účty a role

Uživatelské účty umožňují osobám vstup do systému workflow za účelem přečtení a zpracování zadaného úkolu za pomocí přidělených zdrojů. Každý uživatelský účet má v databázi uložené kontaktní informace, zahashované přístupové heslo a role patřící danému uživateli. Každý uživatel může mít přiřazeno více uživatelských rolí.

Uživatelské role mají v systému účel rozdělení uživatelů podle jejich zaměření dle organizační struktury organizace. Na tyto role se následně odkazuje u jednotlivých úkolů v rámci definice

procesu. Workflow výkonné jádro následně přiřazuje úkol některému z uživatelů, který je zařazen do patřičné skupiny uživatelů dle role.

Speciální administrátorská role dovoluje administrátorovi systému kompletní správu systému. Pouze administrátor může definovat jednotlivé uživatelské role, uživatelské účty či zdroje. Administrátor může delegovat privilegia definice procesů, spouštění, monitoringu či případné předčasné přerušení procesu na další uživatelské role systému.

6.3 Správa procesů

Procesy lze třídit do pojmenovaných kategorií. Kategorie je něco jako systémová složka pro procesy, které mají spolu něco společného. Administrátor může také u každé kategorie určit uživatelské role, které mohou procesy v dané kategorii editovat a mazat, spouštět a monitorovat jejich instance či zastavit nedokončenou instanci procesu. Administrátor vidí veškeré vytvořené kategorie. Ostatní uživatelé mají k dispozici pouze kategorie, do kterých mají alespoň nějaké oprávnění.

Každou editací procesu vznikne jeho nová verze. Každá nová verze je automaticky deaktivovaná pro spouštění instancí. Danou verzi je nutné následně aktivovat, aby bylo možné z ní spouštět instance. Nevyhovující staré verze procesu lze deaktivovat pro spouštění. Deaktivované verze procesu lze vymazat. Samotná editace procesu probíhá interaktivně přes JavaScriptové rozhraní.

Definice celého procesu je rozdělena na dvě části tak, jak je tomu podobně v některých programovacích jazycích. Jedná se o část deklarační a část samotné definice posloupnosti instrukcí. V první části mohou být deklarované proměnné, které budou následně využity. V druhé části jsou definované jednotlivé úkoly, které obsahují definici průběhu celého procesu. Ukázka takové definice se nachází na *obr. 10.*

6.3.1 Deklarace proměnných

Proměnné v rámci definice pomáhají udržovat unikátní stav celé instance procesu na základě interakce s uživateli v rámci jednotlivých úkolů. Při interpretaci procesů je dodržován princip silně typového jazyka, takže každá proměnná je určitého datového typu a nelze do ní přiřadit nic jiného. Ke každé proměnné lze definovat její výchozí hodnotu.

Podporované datové typy jsou:

- boolean pro logické hodnoty true a false
- integer pro celočíselné hodnoty
- float pro reálné hodnoty
- text pro textové řetězce.

Pro datový typ integer je dále možné definovat číselník hodnot, kde se ke každé integer hodnotě přiřazuje významová reprezentace. Kdekoliv, kde je hodnota proměnné s číselníkem vypsána, je tak výpis hodnoty přeložen na význam hodnoty. Jako vstup pro tuto proměnnou pak slouží s selectbox s číselníkovou nabídkou.

Obr. 10 : Definice procesu

6.3.2 Definice úkolů procesu

Každý proces se skládá z alespoň jednoho úkolu. Nový blok se automaticky vkládá na konec seznamu úkolů. Jednotlivým blokům lze určit správné pořadí díky tlačítkům na přesun úkolu pod následující úkol, či přesunu úkolu nad předcházející úkol. V případě, že není splněna žádná podmínka při interpretaci instance procesu, tak vždy následuje sekvenčně následující úkol. Pokud při potvrzení

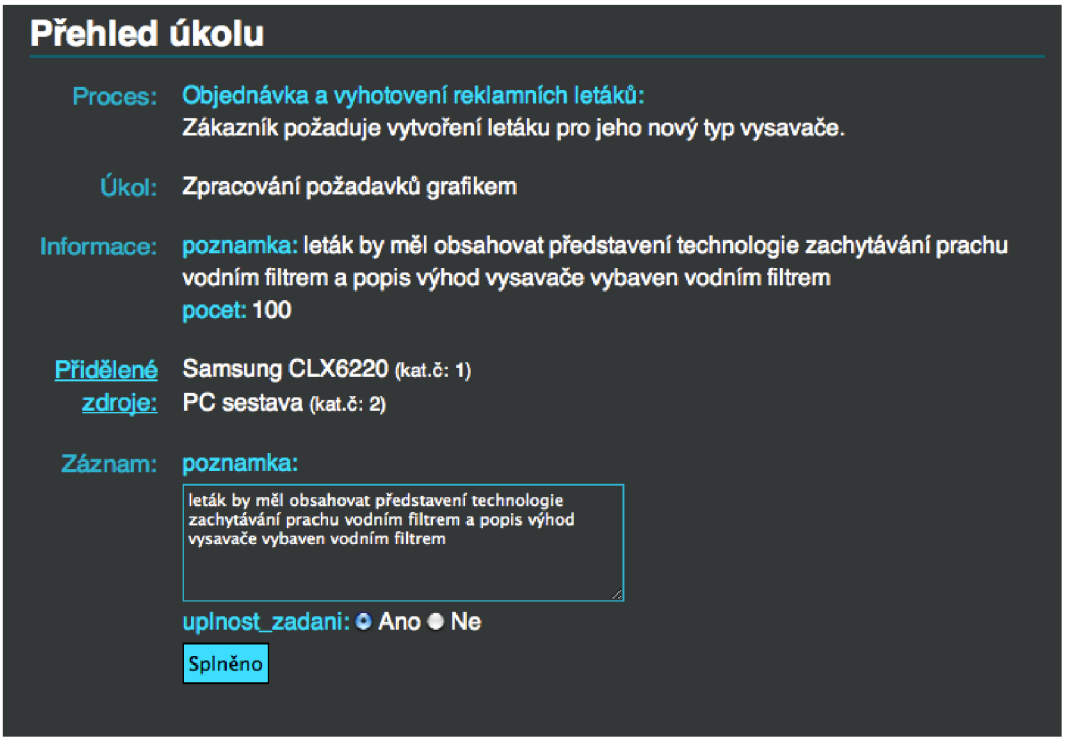
posledního definovaného úkolu není splněna žádná podmínka v rámci tohoto úkolu, tak je instance procesu ukončena.

Definice každého úkolu obsahuje následující položky:

- název úkolu – pojmenování popisující zadání úkolu
- profese – uživatelská role, která je potřebná k vykonání úkolu
- Proměnné notifikace hodnot – jaké proměnné se mají uživateli vypsat v rámci zadání úkolu
- Proměnné odpovědního formuláře – proměnné, kterým uživatel přiřadí hodnotu po vykonání úkolu
- Podmíněné skoky – definice podmínek pro hodnoty proměnných a úlohy, která bude při splnění podmínky následovat

6.4 Seznam úkolů

Každý zanesený úkol výkonným jádrem je v této sekci prezentován uživateli. Zpracovávání úkolu uživatelem zajišťuje pokrok jednotlivých instancí tak, aby dosáhli svého stanoveného obchodního cíle. Ukázka prezentace zadaného úkolu v rozhraní tohoto systému je na *obr. 11*.



Obr. 11 : Ukázka zadaného úkolu

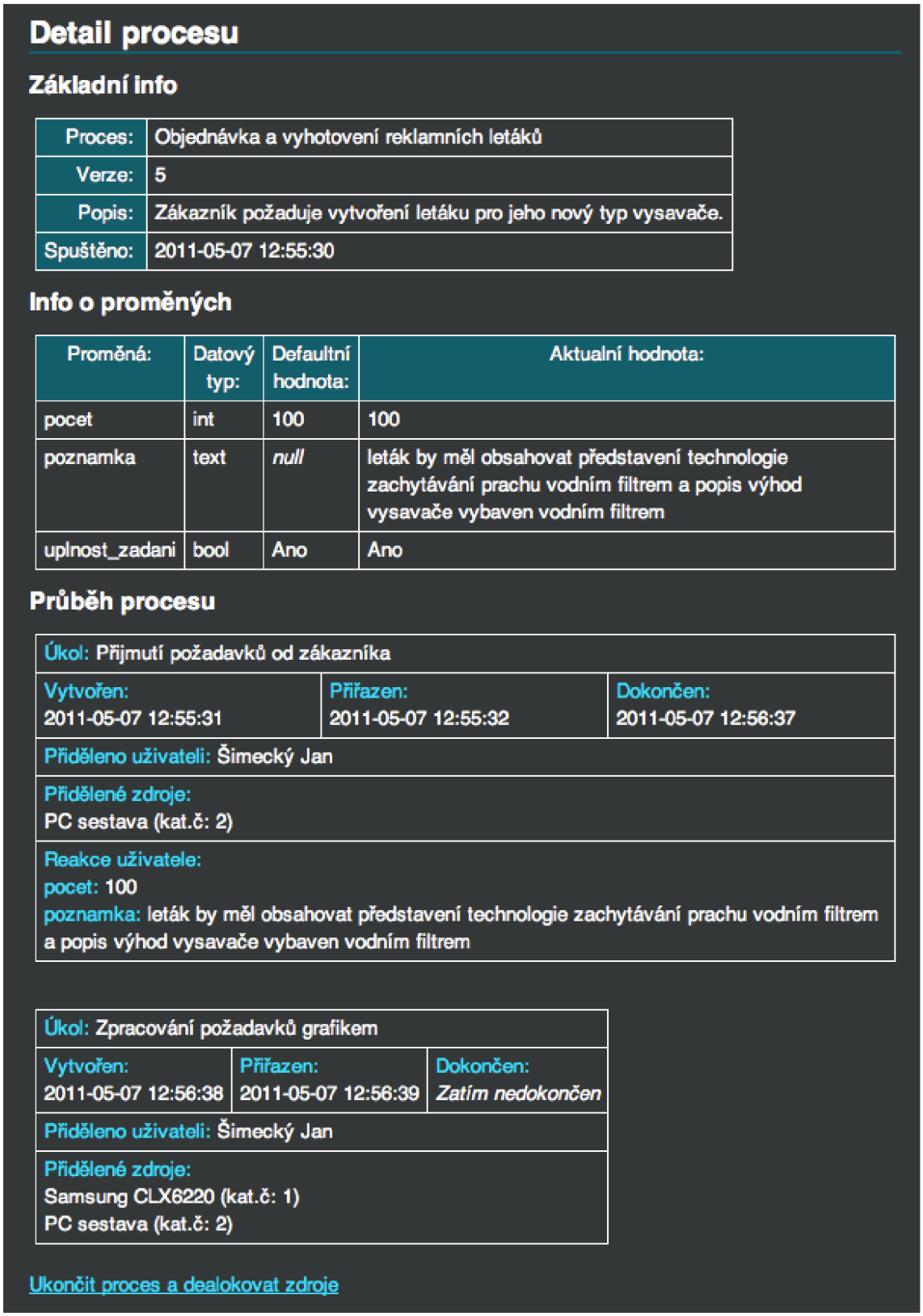
Uživatel systému je zde kompletně seznámen o dané instanci procesu a zadaném úkolu. V zobrazeném úkolu je obsažen název procesu, popis dané instance a popisný název tohoto úkolu. V sekci informace se zobrazují proměnné a jejich hodnota, která se má v rámci notifikace uživateli zobrazit. Pro splnění daného úkolu jsou zapotřebí zdroje, které jsou zde vypsané včetně jejich katalogového čísla. Jakmile uživatel úkol splní, vyplní ve formuláři proměnné, které jsou mu nabídnuté k editaci a potvrdí splnění celého úkolu tlačítkem Splněno. Na mail dotyčného uživatele jsou zasílány stejné informace, jaké jsou zde viditelné. Mail slouží pouze pro zadání úkolu, pro jeho potvrzení je již ale nutné se přihlásit do systému.

6.5 Monitoring

Systém umožňuje sledovat průběh všech aktivních a dokončených instancí procesů. V základním přehledu jednotlivých instancí je vždy název procesu a číslo jeho spuštěné verze. Pro každou instanci je zde dále její popis, který byl zadaný při inicializaci instance. Je zde také uvedeno datum a čas inicializace a v případě ukončeného procesu i datum a čas ukončení instance procesu.

V detailu každé instance procesu lze zjistit její kompletní průběh. Ukázka detailu se nachází na *obr. 12*. V detailu jsou mimo jiné zopakovány informace z přehledu jednotlivých procesů o dané instanci. Dále je zde vyobrazen stav veškerých proměnných instance procesu a samozřejmě následují informace o veškerých proběhlých úkolech. Každý úkol nese informaci o tom, kdy byl inicializován, kdy byl přidělen a kdy dokončen. U každé úlohy je zobrazeno, komu byla přidělena, včetně přidělených zdrojů s jejich katalogovým číslem. Každá úloha zde má zobrazenou kompletní reakci uživatele v rámci odpovědního formuláře.

Běžící proces zde může být zastaven osobou, která má oprávnění pro zastavení procesu. Při zastavení procesu jsou okamžitě uvolněny veškeré alokované zdroje pro další použití.



Detail procesu

Základní info

Proces:	Objednávka a vyhotovení reklamních letáků
Verze:	5
Popis:	Zákazník požaduje vytvoření letáku pro jeho nový typ vysavače.
Spuštěno:	2011-05-07 12:55:30

Info o proměných

Proměná:	Datový typ:	Defaultní hodnota:	Aktualní hodnota:
pocet	int	100	100
poznamka	text	*null*	leták by měl obsahovat představení technologie zachytávání prachu vodním filtrem a popis výhod vysavače vybaven vodním filtrem
uplnost_zadani	bool	Ano	Ano

Průběh procesu

Úkol: Přijmutí požadavků od zákazníka		
Vytvořen: 2011-05-07 12:55:31	Přiřazen: 2011-05-07 12:55:32	Dokončen: 2011-05-07 12:56:37
Přiděleno uživateli: Šimecký Jan		
Přidělené zdroje: PC sestava (kat.č: 2)		
Reakce uživatele: pocet: 100 poznamka: leták by měl obsahovat představení technologie zachytávání prachu vodním filtrem a popis výhod vysavače vybaven vodním filtrem		

Úkol: Zpracování požadavků grafikem		
Vytvořen: 2011-05-07 12:56:38	Přiřazen: 2011-05-07 12:56:39	Dokončen: *Zatím nedokončen*
Přiděleno uživateli: Šimecký Jan		
Přidělené zdroje: Samsung CLX6220 (kat.č: 1) PC sestava (kat.č: 2)		

Ukončit proces a dealokovat zdroje

Obr. 12 : Zobrazení průběhu instance

7 Závěr

V rámci této bakalářské práce se podařilo vytvořit funkční workflow systém, zahrnující funkcionalitu pro uživatelské modelování procesů v rámci JavaScriptového editoru a uložení takto vytvořených procesů. Systém generuje záznamy o pracovních úkolech, které musejí být vykonané a uživatelé jsou o těchto úkolech notifikovaní pomocí e-mailové zprávy. Dle zadání aplikace splňuje požadavek na možnost monitorovat jednotlivé instance procesů a navíc dovoluje i předčasné ukončení a uvolnění alokovaných zdrojů běžící instance.

Funkcionalita byla ověřena testováním v průběhu celé implementace. Vytvořený workflow systém navíc od původního sekvenčního průchodu v zadání tak také podporuje i logiku větvení na základě historie průběhu celé instance.

Jako další možná rozšíření může být podpora priorit procesů, která by mohla být snadno implementovatelná tak, že podle priorit by se hledali volné zdroje pro čekající požadavky. Dalším rozšířením by mohlo být delegování zavádění uživatelů do systému na další uživatele, kteří nemají plné administrátorské oprávnění. Mezi implementačně náročnějšími rozšířeními by mohlo být modelování procesů v rámci grafu, který by obsahoval v uzlech jednotlivé úkoly, a přechodové podmínky by vyjadřovali síť možného průchodu mezi těmito podmínkami. Velmi zajímavým rozšířením by mohla být podpora pro rozvětvení úkolu do paralelně zpracovávaných úloh, které se následně spojí opět do pouze jedné přidělené úlohy.

Literatura

[1] DUMAS, Marlon; AALST, Will van der; TER HOFSTEDE, Arthur. Process-Aware Information System : Bridging People and Software Through Process Technology. New Jersey : John Wiley & Sons, 2005. xvi, 409 s. ISBN 978-0-471-66306-5.

[2] CARDA, Antonín; KUNSTOVÁ, Renáta. Workflow : Řízení firemních procesů. Praha : Grada Publishing, 2001. 136 s. ISBN 80-247-0200-2.

[3] Hollingsworth, David; Workflow Management Coalition : The Workflow Reference Model. Winchester, 1995. 55 s. Document Number TC00-1003

[4] GILMORE, W. Jason. Velká kniha PHP 5 a MySQL : kompendium znalostí pro začátečníky i profesionály. Překlad Jan Pokorný. Vyd. 1. Brno : Zoner Press, 2005. 711 s. ISBN 80-86815-20-X.

[5] POŠMURA, Vlastimil. Apache : Příručka správce WWW serveru. Vydání první. Praha : Computer Press, 2002. x, 311 s s. ISBN 80-7226-696-9.

[6] HOLZNER, Steven. Mistrovství v AJAXu : Naučte se programovat moderní aplikace. Překlad Jakub Zemánek. Vydání první. Brno : Computer Press, 2007. 591 s. ISBN 978-80-251-1850-4.

[7] AALST, Wil van der; HEE, Kees van. Workflow Management : Models, Methods, and Systems. Massachusetts : The MIT Press, 2004. 368 s. ISBN 0-262-72046-9.

[8] SHARP, Alec; MCDERMOTT, Patrick. Workflow Modeling : Tools for Process Improvement and Application Development. 2nd Edition. MA 02062 USA : Artech House Publishers, 2009. 449 s. ISBN 978-1-59693-192-3.

[9] CICHOCKI, Andrzej, et al. Workflow and Process Automation : Concepts and Technology. United States of America : Kluwer Academic Publishers, 1998. x, 125 s. ISBN 0-7923-8099-1.

Seznam příloh

Příloha 1. Manuál

Instalace

1) import databázové struktury MySQL z *databaze.sql*
2) nastavení přístupu do databáze MySQL v class/sql.php (řádek 64)
3) nastavení práv zápisu do *templates_c* (kompilované šablony Smarty)
4) Kontrola zda, lze v interpretu z konzole správně pustit *php engine.php*, případně zda neschází závislost na PEAR (http://pear.php.net/) pro parsování XML z *XML_Unserializer*
5) nastavení spouštěče výkonného jádra – nastavení minutového cronu pro *run.php* (spouštěč kontroluje, zda běží *engine.php* a pokud ne, tak jej zapne).

Práce se systémem krok za krokem

1) Přihlášení přes výchozí administrátorský účet, který má login admin s heslem admin
2) V menu *Správa kategorie zdrojů* nadefinovat jednotlivé typy užití zdroje
3) V menu *Správa zdrojů* nadefinovat jednotlivé zdroje a přiřadit jim užití
4) V menu *Správa uživatelských rolí* nadefinovat další uživatelské role
5) V menu *Správa uživatelských účtů* vytvořit uživatelské účty jednotlivým uživatelům a přiřadit jim jejich role
6) V menu *Správa kategorií procesů* vytvořit kategorie (složky) na procesní definice a nastavit přístupová práva pro jednotlivé akce dle uživatelských rolí
7) V menu *Správa procesů* vložit nové procesy, kterým následně definovat průběh v rámci vytvoření nové verze procesu.
8) Aktivovat patřičné definované verze ke spouštění a spustit instance
9) Plnit úkoly na jednotlivých uživatelských účtech
10) Sledovat průběh běžících a dokončených instancí.