Masarykova univerzita
Fakulta informatiky

Integrace DMS a workflow v Liferay Portal

Diplomová práce

Marek Tlačbaba

Brno, 2014

Prohlášení

Prohlášení

Vedoucí práce: RNDr. Jaroslav Ráček, Ph.D.

Poděkování

Poděkování

Shrnutí

Shrnutí

Klíčová slova

Klíčová slova

Obsah

Úvod

Kapitola 1

Podnikové portály

Pojem portál bývá definován různými způsoby v závislosti na situaci, ve které je používán. Dříve se portály zaměřovaly na jednotný přístup k informacím nezávisle na jejich původu a umístění. S dalším vývojem informačních technologií se postupně tento přístup rozšiřoval a do definice portálu přibyly pojmy jako integrace a agregace služeb a aplikací. V následujícím textu je uvedeno několik definic portálu a podnikového portálu.

Například Gála definoval portál jako jednotné rozhraní, ve kterém lze pracovat s běžnými službami a nástroji jako jsou, například zpravodajství a komunikace, mimo to je v něm zaručen přístup k všeobecným aplikacím jako jsou vlastní stránky a blogy, a k aplikacím specializovaným, například slovníkům. [1]

Další definice portálu je převzata ze standardu JSR-286 a zní následovně: Portál je webová aplikace, která (obvykle) poskytuje možnost personalizace, autentifikace, agregace obsahu z různých zdrojů a slouží jako prezentační vrstva informačních systémů. Agregací se rozumí integrace obsahu z různých zdrojů v rámci jedné webové stránky. Portál může mít vysoce propracované nástroje pro personalizaci, které poskytují obsah upravitelný podle přání uživatele. Stránky portálu mohou mít různé skupiny portletů [1], které vytvářejí obsah pro různé uživatele. [2]

Předchozí definice se týkají portálu obecně, proto zde uvedu také definici podnikového portálu jako takového. Například podle Čecha je podnikový portál definován jako internetové nebo intranetové stránky, které slouží jako vstupní bod, respektive brána k různým datovým, informačním a znalostním zdrojům v organizaci. Jejich cílem je zpřístupnit tyto zdroje specifické skupině lidí. Může to být jak zákazníkům, tak také vlastním zaměstnancům nebo partnerům. [3]

V předchozích definicích portálu jsou naznačeny některé základní rysy podnikových portálů. Mezi důležité vlastnosti můžeme považovat jediný přístupový bod, integrace, federace, přizpůsobivost, personalizace, kontrola pří-

1. Bližší informace v dalších částech této práce.

stupu a vyhledávání v podnikovém obsahu, tak jak jsou uvedeny v [4].

1.1 Konkrétní portálová řešení

Existuje a používá se celá řada konkrétních portálových řešení. Nejčastěji bývají implementovány v jazyce Java a podporují standardy JSR-168 a JSR-286 [2] pro tvorbu portletů v jazyce Java. Existují také portály i pro jiné technologie, například pro platformu .NET je možné využít Microsoft SharePoint, ale nejvíce řešení existuje pro jazyk Java, kde mezi nejznámější patří IBM Websphere Portal, JBoss GateIn, Oracle WebCenter či Liferay Portal. Tyto portály již většinou obsahují velké množství připravených portletů, např. wiki stránky, správu souborů, portlety pro komunikaci a další.

1.2 Liferay Portal

Liferay Portal je open source podnikový portál založený na jazyce Java vyvíjený společností Liferay, Inc.. Podporuje specifikace JSR-168 a JSR-286 pro tvorbu portletů v jazyce Java. Liferay Portal je k dispozici ve dvou edicích, Community Edition (CE) a Enterprise Edition (EE).

Liferay Portal CE je bezplatná verze portálu, která je volně dostupná ke stažení. Tato verze neobsahuje oficiální podpory, je zde pouze podpora poskytovaná prostřednictvím Liferay komunity. Druhá verze portálu, Liferay Portal EE, je placená edice. U této verze je kladen velký důraz na stabilitu, bezpečnost a výkon. K této edici patří dlouhodobá podpora od společnosti Liferay, Inc. nebo jejích oficiálních partnerů.

Ihned po instalaci portálu je k dispozici množství již předinstalovaných portletů. Je možné například okamžitě využívat wiki stránky, fórum, kalendář či galerii obrázků. Liferay Portal podporuje různé způsoby úprav či rozšíření. Pro změny vzhledu můžeme využít témata či šablony, další funkcionalitu portálu je možno přidávat pomocí portletů. Pro změnu portálu či portletů se používají tzv. hooks. Pokud se jedná o větší a zásadnější změny přímo v jádru portálu, ty se provádí pomocí pluginu Ext. [5]

Portál Liferay nabízí svým uživatelům zabezpečené jednotné přihlášení, nástroje pro správu workflow, snadnou instalaci samotného portálu i dalších nových aplikací a rozšíření. Od verze 6.1 CE GA2 umožňuje instalaci rozšíření a aplikací nově také z Liferay Marketplace. Takto je možné stahovat aplikace přímo z rozhraní portálu a jednoduše je přidat do portálu.

Liferay také obsahuje pokročilý systém pro správu obsahu (Liferay CMS),

2. Bližší informace v dalších částech této práce

který umožňuje oprávněným uživatelům vytvářet a spravovat obsah webu přímo z prohlížeče a to i bez znalosti programování. S tím také souvisí možnost dělit uživatele do organizací, skupin a uživatelům přidělovat role, podle kterých jim následně zobrazovat různý obsah, aplikace a umožňovat provádět různé akce.

Mezi další vlastnosti portálu patří vícejazyčné uživatelské rozhraní, personalizace portálových stránek, jednoduchá úprava stránek způsobem táhni a pusť, automatické nahrávání souborů.

Liferay také podporuje různé platformy a proto je možné ho provozovat na různých aplikačních serverech, operačních systémech a databázích. Také podporuje různé metody integrace včetně SOAP, REST, RSS a také další proprietární rozhraní.[6]

To jsou některé z důvodů, proč se podnikový portál Liferay stal rozšířenou a oblíbenou platformou pro vývoj firemních webů a systému v jazyce Java.

Kapitola 2

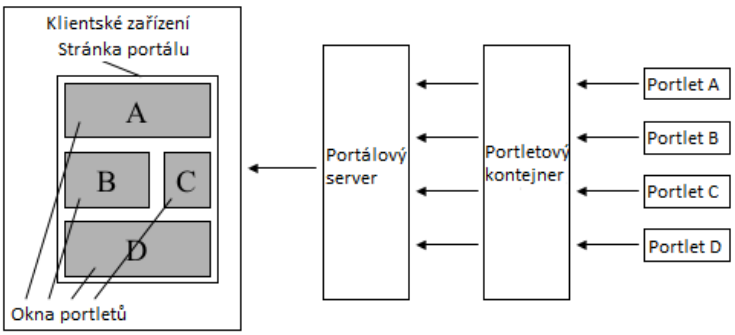
Vývoj portletů

V této kapitole bude popsán vývoj portletů, které tvoří obsah portálů.

2.1 Portlet

Portlet je webová aplikace, která poskytuje uživatelům specifický obsah, typicky informaci nebo službu. Portlet je spravován portletovým kontejnerem, který zpracovává požadavky portletu a následně generuje dynamický obsah v podobě fragmentů v některém ze značkovacích jazyků – HTML, XHTML, WML. Tyto fragmenty jsou pak spojovány a dohromady vytváří portálovou stránku.

Portletový kontejner má tedy odpovědnost za řízení životního cyklu jednotlivých portletů a je v něm také uloženo nastavení uživatele pro daný portlet. Ale části generované jednotlivými portlety slučuje dohromady na stránku portál. Portál a portletový kontejner mohou být postaveny jako jedna komponenta ale mohou to být také komponenty dvě. Na následujícím obrázku 3.1 je zobrazen průběh vytváření stránky v portálu tak, jak bylo výše naznačeno.



Obrázek 2.1: Vytváření stránky v portálu (převzato z [2])

2.2 Specifikace

Důvodem pro vytvoření portletové specifikace byla situace, kdy různí dodavatelé používali svá vlastní API, která neumožňovala přenositelnost mezi portálovými servery.

První verze Java Portlet Specification 1.0 (JSR-168) byla vydána v roce 2003. V této specifikaci ještě bylo stále různé nedostatky a chyby, které pak tvůrci portletů stejně řešili každý po svém. To způsobovalo i nadále, že portlety byly mezi portálovými servery stále problematicky přenositelné či nepřenositelné úplně. Proto vznikla druhá verze Java Portlet Specification 2.0 (JSR-286) a ta byla vydána v červnu 2008. Mezi nejdůležitější přínosy patří následující:

- sdílení parametrů mezi portlety skrz veřejné render parametry,
- meziportletová komunikace pomocí zasílaní událostí,
- transformace informací nově definovanými portletovými filtry,
- možnost portletů poskytovat zdroje jako jsou například PDF soubory.

Většina portletových kontejnerů sice poskytuje k základním požadavkům i svá rozšíření. Tyto rozšíření však nemusíme využívat a tak můžeme zachovat myšlenku snadné přenositelnosti.

2.2.1 Životní cyklus portletu

Každý portlet musí implementovat rozhraní `Portlet`. Toto rozhraní obsahuje čtyři základní metody, které musí být implementovány a které řídí životní cyklus portletu - `init()`, `processAction()`, `render()`, `destroy()`.

Metoda `init()` je volána ihned při vytváření portletu a je používána například pro přípravu zdrojů potřebných portletem.

Metoda `processAction()` je volána vždy po provedení nějaké akce uživatelem, když se touto akcí nějak mění stav portletu. Může to být například akce potvrzení formuláře a následné zpracování a uložení dat v databázi.

Metoda `render()` má za úkol generovat fragment stránky, který je pak dostupný uživateli v portletu. V této metodě by se neměl měnit stav portletu.

Poslední metodou je `destroy()`. Ta je volána před odstraněním portletu z paměti. Poskytuje poslední možnost pro uvolnění zdrojů, které byly portletem používány. [2]

Zpracování požadavků v portletech může procházet několika fázemi zpracování.

- Render – vykreslovací fáze, ve které se typicky nemění stav portletu, odpovídá jí metoda `render()`.
- Action – portlet zpracovává nějakou akci a typicky mění svůj stav, odpovídá ji metoda `processAction()`.
- Resource – portletem je poskytován nějaký zdroj, např. PDF soubory. Aby portlet mohl podporovat tuto funkcionalitu, musí implementovat rozhraní `ResourceServingPortlet`.
- Event – portlet přijímá nějakou událost a nějak na ni reaguje. Portlet v tomto případě musí implementovat rozhraní `EventPortlet`.

Všechny tyto rozhraní implementuje abstraktní třída `GenericPortlet`. Tato třída obsahuje implementaci všech povinných metoda a několika dalších, které je možné využít.

Po každé action fázi musí následovat fáze render. Pokud se na portálové stránce vyskytuje více portletů, je metoda render zavolána u všech zobrazených portletů.

2.2.2 Režimy portletu

Portletová specifikace obsahuje tři režimy portletu, které umožňují zobrazit jiný obsah v závislosti na požadovaném úkolu. Tyto režimy využívá metoda `render()`, která podle nich vygeneruje odpovídající obsah.

- View – základní režim používaný při klasickém zobrazení portletu na portálové stránce.
- Edit – nepovinný, uživateli umožňuje nastavovat portlet.
- Help – nepovinný, zobrazuje se nápověda k danému portletu.

Ve třídě `GenericPortlet` jsou k těmto účelům definovány metody `doView()`, `doEdit()` a `doHelp()`. Jednotlivé portletové režimy by měli být implementovány přetížením těchto metod. Každý portál si také může definovat vlastní portletové režimy. Každý portlet musí mít zadefinováno, které režimy podporuje.

2.2.3 Stavy okna

Portletová specifikace definuje tři základní stavy.

- Minimized – okno portletu je minimalizované.

- Normal – původní velikost portletu, které umožňuje zobrazení všech portletů na na portálové stránce.
- Maximized – okno portletu skryje ostatní portlety na stránce a zobrazuje se pouze tento maximalizovaný portlet, nebo zakrývá většinu dostupné plochy.

2.2.4 Struktura portálové aplikace

Portálové aplikace jsou uloženy ve standardní archivním souboru jazyka Java, Web ARchive (WAR). Mohou obsahovat více portletů, servlety, JSP soubory a další. Musí obsahovat webový deskriptor a také navíc portletový deskriptor implementace `portlet.xml`.

Kapitola 3

Workflow

Jednoznačná definice workflow neexistuje, jelikož každá oblast, ve které je workflow používáno, si ho definuje po svém. Obecně je ale možné definovat workflow jako proces automatizace podnikových procesů. O sjednocení této terminologie se již v roce 1996 pokusila institut Workflow Management Coalition (WfMC), kdy vydal terminologický slovník, ve kterém je workflow definováno takto:

Workflow znamená automatizaci celého nebo části podnikového procesu, během kterého jsou dokumenty, informace nebo úkoly předávány od jednoho účastníka procesu k druhému podle sady procedurálních pravidel tak, aby se dosáhlo nebo přispělo k plnění celkových/globálních podnikových cílů. [8]

V počítačových systémech je workflow definováno jako systém řízení workflow. V tomto systému je možné workflow definovat, vytvořit a celý průběh procesu řídit. Mezi jeho úkoly patří provádět definici procesu, komunikovat s účastníky workflow, případně spouštět další aplikace nutné k vykonání workflow.

3.1 Typy workflow systému

Workflow systémy se mohou dělit do několika skupin podle různých hledisek či typů. [9]

3.1.1 Hledisko charakteru procesů

- **Administrativní** workflow systémy jsou určeny k vyřizování běžných administrativních úkonů, které jsou většinou jednoduché, opakující se a dobře strukturované.
- **Ad-hoc** workflow systémy jsou založeny na náhodnosti vzniku. Procesy bývají jedinečné a je nutné je definovat až při jejich vzniku.
- **Kolaborativní** workflow systémy podporují především týmovou spolupráci. Zde je typickým znakem existence nějakého dokumentu, skrze

který si uživatelé vyměňují poznatky a většinou se tento dokument stává i výsledkem společné práce. Často obsahuje opakovaní několika iterací stejného kroku dokud nedojde ke shodě.

- **Produkční** workflow systémy podporují hlavní podnikové procesy, které vytvářejí přidanou hodnotu k finálnímu produktu.

3.1.2 Hledisko orientace procesů

- **Procesy orientované na lidi.** U těchto systémů spoléhají účastníci sami na sebe. Předávané informace jsou proměnlivé a procesy nejednotné a špatně předpověditelné. Z těchto důvodů jsou průběhy závislé na jednotlivcích.

- **Procesy orientované na sebe.** Tyto systémy jsou zaměřené na klíčové procesy, které obvykle bývají hlavními aktivitami podniku. Jejich pravidla řešení a jejich zpracování jsou pevně daná.

3.1.3 Hledisko technologické infrastruktury

Podle technologické infrastruktury, nad kterou je systém workflow postaven, lze produkty rozdělit do několika skupin.

- Založené na **elektronické poště** – využívají dostupné emailové servery. Uživatelé nepotřebují speciální software.

- Založené na **procesech** – mohou implementovat vlastní komunikační mechanismus a jsou postaveny na určitém databázovém systému. Bývají tvořeny jako komplexní řešení uceleného konceptu workflow.

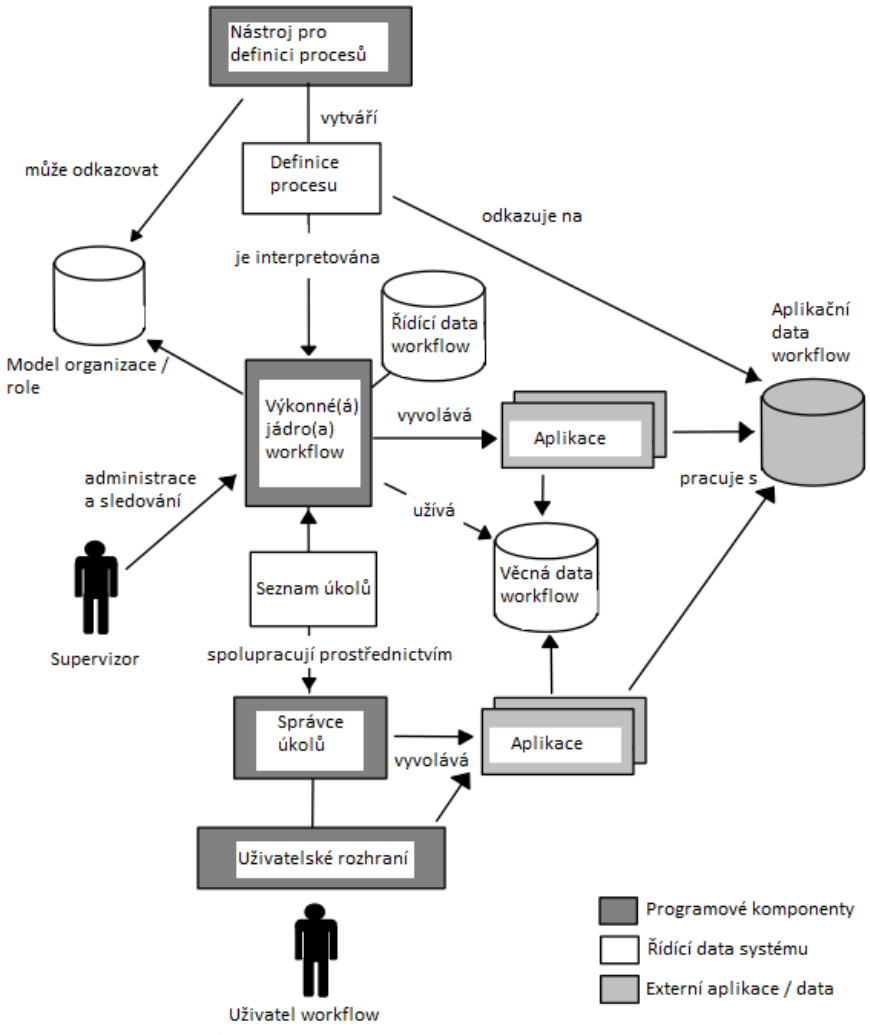
- Založené na **webu** – využívají jednotného rozhraní intranetové aplikace. Je to univerzální platforma pro sdílení informací.

- Založené na **dokumentech** – tyto systémy byly motivovány představou o směrování dokumentů. Využívají systémy pro správu dokumentů.

3.2 Obecný model workflow systému

Institut WfMC vytvořil obecný model, kde mají produkty podobné komponenty viz obr. 4.1. Jednotlivé komponenty systému se dělí na programové a datové.

Programové komponenty.



Obrázek 3.1: Obecný model workflow (převzato z [8])

- Nástroj pro definici procesů (definition tool) umožňuje definovat procesy, přiřazovat role a stanovat pravidla.
- Výkonné jádro workflow (workflow engine) řídí průběh workflow procesů, spouští nutné externí aplikace a udržuje statistiky o průběhu workflow.
- Správce úkolů (worklist handler) má jako hlavní úkol zprostředkovávat komunikaci mezi jádrem workflow a jednotlivými uživateli.
- Uživatelské rozhraní (user interface) zajišťuje komunikaci mezi správcem úkolů a uživatelem. Často tvoří jeden celek spolu se správcem úkolů.

Datové komponenty.

- Definice procesu (process definition) popisuje strukturu procesu.
- Řídící data workflow (workflow control data) jsou interní data, které zpracovává jádro systému.
- Aplikační data workflow (workflow application data) jsou specifická data aplikací.
- Věcná data workflow (workflow relevant data) jsou data používaná jádrem k vyhodnocování dalších kroků.
- Seznam úkolů (work list) představuje datovou strukturu, ve které jsou uloženy úkoly pro uživatele. Mohou být viditelné všechny najednou, nebo mu úkoly mohou být poskytovány postupně.
- Model organizační struktury (organization/role model) popisuje organizační strukturu podniku. Pokud není definován, musí být úkoly přiřazovány pouze konkrétním uživatelům.

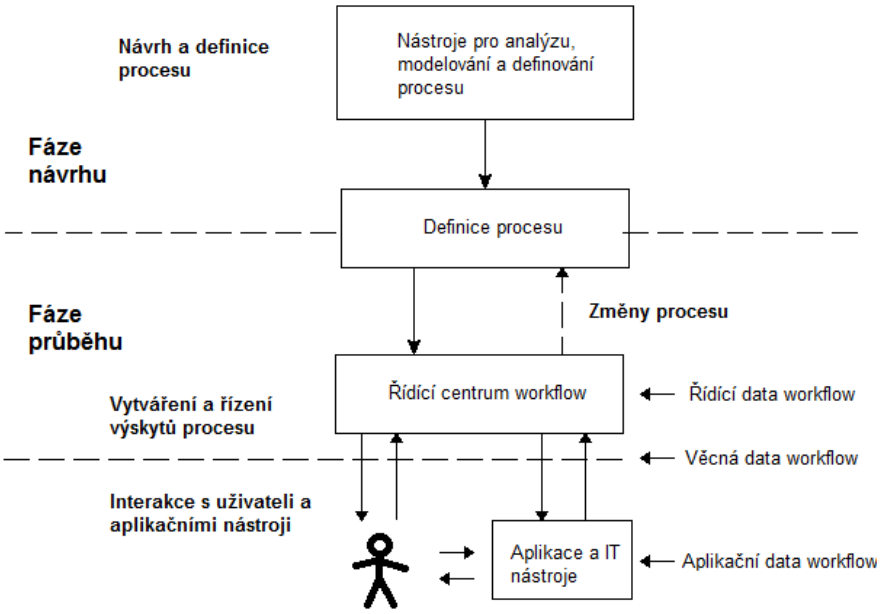
3.2.1 Fáze workflow

Obecně jsou rozlišovány dvě fáze workflow: fáze návrhu workflow a fáze průběhu workflow. [9]

Fáze návrhu workflow v sobě zahrnuje funkce pro návrh a definici procesu, které jsou poskytovány analytickými a modelovacími nástroji. Ty umožňují převod procesu z reálného světa do normalizované podoby. Výsledkem je počítačově zpracovatelný popis procesu ve smyslu: kdo, kdy, co, s čím, za jakým účelem a jak má udělat.

Fáze průběhu workflow se dále dělí na funkce pro řízení běhu procesu, které zabezpečují interpretaci procesu, spouštění, provádění a kontrolu průběhu jednotlivých činností. Další část této fáze představuje interakce s uživateli a aplikačními nástroji. To například znamená předávání úkolů ke zpracování, vyžádání manuální činnosti, automatické spouštění jiných aplikací či předávání dat mezi aplikacemi.

Vše je znázorněno na následujícím obrázku.



Obrázek 3.2: Fáze workflow (převzato z [8])

3.3 Workflow pro procesy pracující s vnitropodnikovými dokumenty

Z pohledu charakteru procesů řadíme workflow pro procesy pracující s dokumenty do administrativních workflow. Většinou jsou to již procesy ustálené, dobře strukturované a tyto procesy jsou využívány většinou uživatelů systému. Z hlediska technologické infrastruktury jsou to systémy založené na dokumentech. Z hlediska orientace procesů patří do procesů orientovaných na sebe, jsou pro něj totiž typické vlastnosti jako předpověditelnost, strukturovanost, stálý postupu a je aktivován dokumenty.

3.4 BPMN

Buisiness Process Model And Notation (BPMN) je standard pro modelování podnikových procesů, jehož výstupem bývá grafické znázornění podnikových procesů v podobě procesních diagramů. [10] Jazyk BPMN také pomáhá sjednocovat významy základních pojmů používaných v této oblasti procesního řízení.

Hlavním účelem BPMN je podporovat procesní řízení. Pro tuto funkci poskytuje notaci, která je jednoduchá, srozumitelná a intuitivní i pro netechnické pracovníky a zároveň dostatečná i pro vyjádření komplexních procesů. Přináší tedy standardizovaný zápis ve srozumitelné podobě pro všechny zainteresované osoby v organizacích.

Poslední verze BPMN 2.0 byla vydána v lednu 2011 a klade si za cíl být jedinou notací pro tvorbu modelů podnikových procesů. Proto mezi základními rysy jsou:

- snaha vytvořit jednotný konzistentní jazyk sjednocením definice podnikových procesů BPMN,
- možnost vytvořit nezávislý nebo integrovaný model,
- podpora a možnost výměny odlišných pohledů na procesní model tak, že je možno se v procesu zaměřit na slabá místa,
- poskytnout xml schémata sloužící pro transformaci modelů.

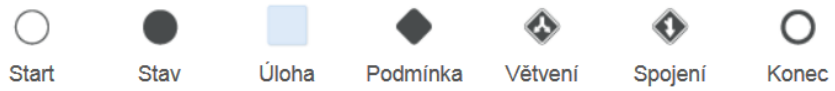
Objekty jazyka BPMN jsou děleny do těchto základních kategorií. Tokové objekty, spojovací objekty, plavecké dráhy a artefakty. [10] Workflow systém Kaleo umožnuje využít pouze tokové objekty a základní spojovací objekty.

3.4.1 Tokové objekty

Bývají označovány jako hlavní grafické prvky definující firemní procesy a dále se dělí, viz Obrázek 3.3.

- Události představují děje či události, ke kterým dochází v průběhu procesu. Na zmíněném obrázky to jsou objekty: Start, Stav a Konec.
- Aktivity vyjadřují činnosti, které se odehrávají uvnitř procesu. Ty se dále dělí úlohy a podprocesy. Je to objekt: Úloha.
- Brány jsou využívány pro zobrazení větvení a slučování toků a procesů, kdy je potřeba zohlednit různé podmínky. Dále jsou děleny na

exkluzivní, inkluzivní a paralelní. Jsou to objekty: Podmínka, Větvení a Spojení.



Obrázek 3.3: Tokové objekty používané v editoru

3.4.2 Spojovací objekty

Spojové objekty slouží ke spojování tokových objektů. Dělí se na sekvenční tok, tok zpráv a asociaci. Workflow systém Kaleo umožňuje využít pouze sekvenční tok, který znázorňuje posloupnoust procesních toků. Zdrojem a cílem je vždy nějaký tokový objekt.

Obrázek 3.4: Spojovací objekt používaný v editoru

3.4.3 Plavecké dráhy

Plavecké dráhy, někdy nazývané kontexty, slouží k organizování a kategorizaci činností. Rozlišují se typy: bazén a dráha.

- Bazén odděluje různé části popisované organizace.
- Dráhy jsou součástí bazénu a používají se ke kategorizaci činností v rámci bazénu.

3.4.4 Artefakty

Artefakty umožňují přidávat další informace do modelu a tím rozšiřují dostupné elementy v BPMN. Tím mohou zvyšovat informační hodnotu modelu. Existují tyto artefakty: datové objekty, skupiny a anotace.

- Datové objekty reprezentují nezbytná data pro vykonání dané činnosti.
- Skupiny jsou využívány pro seskupení různých aktivit.
- Anotace dodávají modelu srozumitelnost a přehlednost.

Kapitola 4

Analýza a návrh aplikace

Výstupem této diplomové práce je grafický editor pro návrh workflow, který bude pracovat v základní notaci BPMN a bude kompatibilní se standardním workflow systémem Kaleo, který je součástí Liferay Portal CE. V této kapitole nejprve popíši webový designer, který je využívaný pro jiný workflow systém Activiti, pak bude popsán Kaleo Designer, který je dodávaný v komerční verzi portálu Liferay Portal EE. Na závěr kapitoly bude provedena analýza požadavků na vytvářený systém.

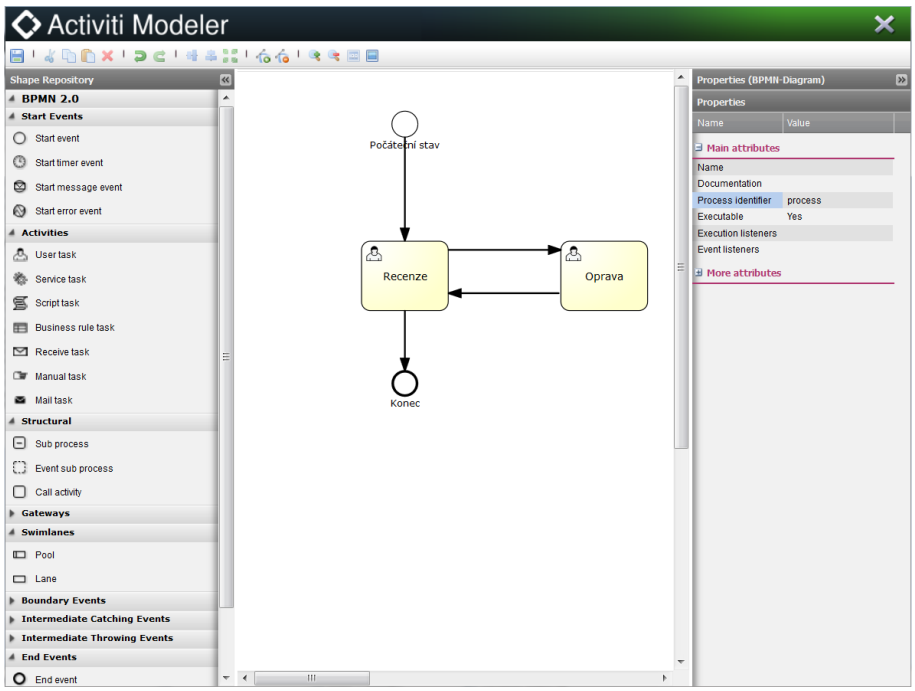
4.1 Activiti workflow

Prvním používaným workflow systémem, který je možno použít v Liferay portálu je Activiti. Activiti se prezentuje jako jednoduché workflow, podporující standard BPMN 2. Tento projekt je šířen jako open-source pod licencí Apache license. Tento systém obsahuje několik nástrojů, které tvoří celou platformu Activiti a jsou to Activiti Engine, Activiti Modeler, Activiiti Designer, Activiti Explorer, Activiti Cycle a Activiti REST.

Pro tvorbu workflow je možné využít nástroje Activiti Modeler a Activiti Designer. Activiti Designer je plugin do vývojového prostředí Eclipse. Výsledkem této diplomové práce má být webová aplikace pro tvorbu workflow a proto tento nástroj není vhodný pro srovnání.

Dalším nástrojem, který umožňuje vytvářet workflow je Activiti Modeler Obrázek 4.1. Je to open source webový nástroj pro modelování procesů. V počáteční fázi byl vyvíjen společností KIS BPM, nyní je již vyvíjen jako součást projektu Activiti. Hlavní cílem této aplikace je podporovat všechny BPMN elementy a rozšíření podporované Activiti Engine.

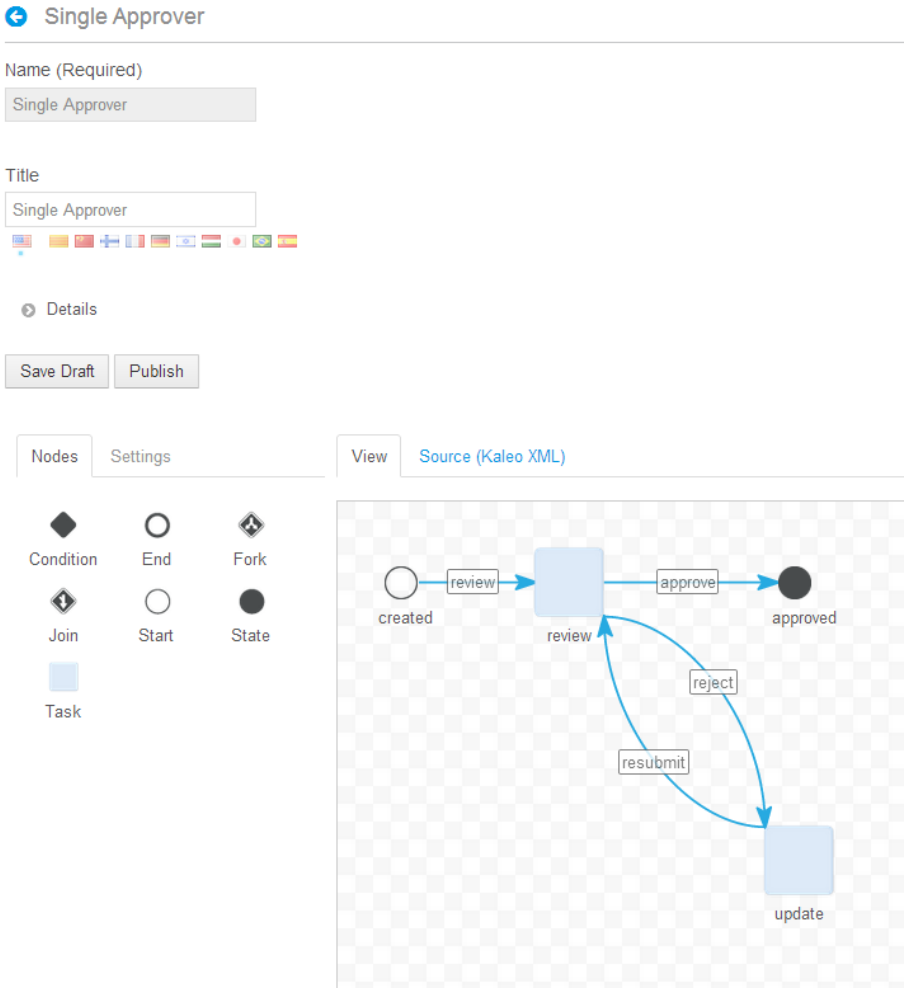
Celá aplikace se skládá ze čtyř základních panelů. V prvním horní vodorovném panelu jsou uloženy všechny tlačítka, které reprezentují všechny akce, které je možné při modelování využít. Panel vlevo slouží jako paleta dostupných BPMN elementů. Na plátno, což je prostřední panel, se vkládají pomocí metody táhni a pusť. Poslední panel, umístěný vpravo, slouží pro úpravy vlastností jednotlivých entit či vlastností celého workflow.



Obrázek 4.1: Activiti modeler

4.2 Kaleo designer v Liferay EE

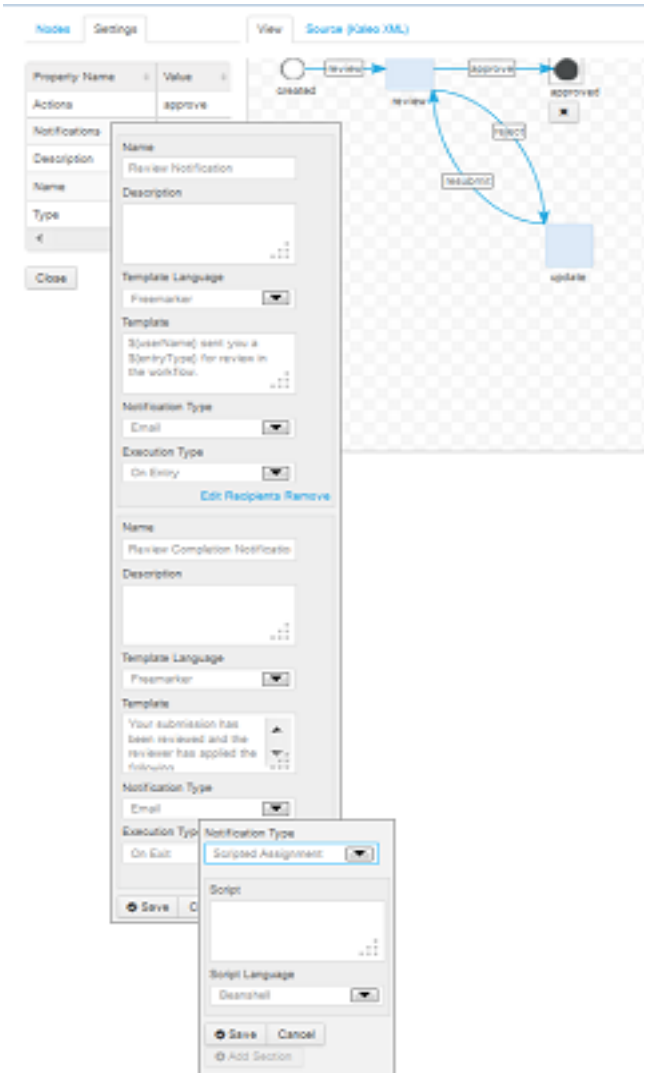
Kaleo designer je součástí portálu Liferay EE a je integrován přímo v portálu a je dostupný z control panelu. Je možné ho rozdělit na dvě hlavní části: seznam dostupných workflow v portálu a samotný grafický designer . V dalším popisu se zaměřím pouze na grafický designer, jehož detail můžeme vidět zde: Obrázek 4.2.



Obrázek 4.2: Kaleo Designer detail

Tento designer se také skládá ze tří základních panelů. V horním panelu jsou zobrazeny základní informace o workflow a také tlačítka pro dostupné akce s workflow. Pak se dělí na dva další panely. Levý panel slouží buď jako

paleta dostupných elementů, nebo v případě vybrání některého elementu se v tomto panelu zobrazují podrobnosti a je možné je měnit pomocí vyskakovacích oken, ve kterých jsou dané formuláře. Těch vyskakovacích oken může být i více mohou se dokonce řetězit na sebe, což z uživatelského hlediska nepřijde úplně šťastné. Celkem jednoduše se můžeme dostat do situace, jenž je znázorněna zde: Obrázek 4.3. Pravý panel obsahuje vlastní plátno nebo zdrojový kód v xml podle volby.



Obrázek 4.3: Řetězení vyskakovacích oken

4.3 Požadavky na vytvářenou aplikaci

Grafický editor pro tvorbu je velice speciální aplikace a při analýze požadavků jsem zjistil, že tento produkt má pouze jeden diagram užití. Hlavně z tohoto důvodu jsem zvolil analýzu v podobě seznamu požadavků.Tento způsob zápisu je také bližší zákazníkům, kterým bude následně tento produkt nabízen.

Tabulka 4.1: Seznam požadavků

Požadavek	**Návrh řešení**
Kompatibilita s workflow systémem používaným v Liferay CE.	Výsledné XML schéma bude odpovídat požadovanému schématu workflow systémem Kaleo.
Aplikace umožní vytvářet a vkládát nové workflow definice. Také umožní načítat již vložené definice v portálu.	Aplikace bude napojena skrz portálové API Liferay k portálu.
Aplikace umožní exportovat výsledné definice ve formátu XML.	Bude možné exportovat vytvářené workflow definice do souboru ve formátu XML.
Aplikace bude v portálu přístupný pouze pro administrátory.	Editor bude umístěn v ovládacím panelu portálu v sekci konfigurace.
Aplikace by být přehledná a jednoduše použitelná.	Bude omezeno použití vyskakovacích oken, které aplikace tohoto typu znepřehledňují.
V editoru bude vytvářet i rozsáhlejší workflow definice.	Editor bude umožňovat nastavit libovolnou velikost plátna.
Uživatel bude mít možnost zkontrolovat správnost vytvářené workflow definice.	Editor bude umožňovat validaci vytvářené workflow definice.
Editor umožní uživateli vracet zpět provedené úpravy.	Editor bude podporovat funkcionalitu zpět / vpřed.
Editor bude pracovat v základní notaci BPMN.	Tento požadavek je omezen vlastnostmi workflow systému Kaleo, jenž umožňuje využít pouze některé prvky.

Pokračování na další straně

Tabulka 4.1 – *Pokračování z předchozí strany*

Požadavek	**Návrh řešení**
Editor bude umožňovat tvořit a upravovat skripty k akcím u jednotlivých elementů.	K těmto účelům bude k dispozici webový editor.
K tvorbě skriptů bude mít uživatel k dispozici předpřipravené některé nejpoužívanější funkce.	Bude k dispozici seznam seznam vložitelných funkcí v jednom zvoleném skriptovacím jazyku.
Výsledná aplikace by měla grafickým rozhraním zapadat do prostředí Liferay Portálu.	Pro tvorbu grafického rozhraní budou využity technologie, které jsou použity pro tvorbu grafického rozhraní portálu.
Kde to bude možné, bude ve formulářích dostupné automatické doplňování uživatelů z portálu.	Bude vytvořeno napojení na portálovou databázi uživatelů a rolí.

Kapitola 5

Realizace

Aplikace není tvořena klasickou třívrstvou architekturou, protože není potřeba řešit perzistenci dat přímo v aplikaci. Výsledná workflow definice se buď vkládá přímo do portálu skrze portálové api nebo je možné výsledný XML soubor exportovat z aplikace. Proto se v následující části zaměřím hlavně na popis prezentační vrstvy a využití portálového api.

5.1 Portletové MVC rámce

Ve druhé kapitole byl představen životní cyklus portletu a hlavní koncepty vývoje portletů v Javě. Při vývoji podle portletové specifikace se portlet vytváří rozšířením třídy `GenericPortlet` a pomocí action a render metod, které obsahují navigační a validační logiku a zároveň se starají o zpracování akcí a generování obsahu. S přibývajícím množstvím funkcí se pak tato třída stává nepřehlednou a špatně udržovatelnou. Tento problém je možné řešit tím, že se rozdělí zodpovědnosti za zpracování požadavků do více specificky zaměřených komponent pomocí některého portletového rámce.

Portletové či webové rámce v sobě implementují osvědčené postupy a vzory pro zjednodušení vývoje. Pro vývoj portletů lze využít i ne přímo portletových rámců, například Struts nebo JSF, ale v takovémto případě je nutné použít takzvané portletové mosty, které řeší rozdíly mezi servletovým a portálovým životním cyklem. Toto řešení ale v sobě obsahuje vícefázový životní cyklus portletů a někdy to znesnadňuje využití plných možností portletové technologie. Portletový rámec navržený přímo pro vývoj portletů je rámec Spring MVC Portlet, který byl zvolen také pro vývoj této aplikace. Tento portletový rámec byl zvolen zejména proto, že je standardně používaný ve firmě IBA CZ, pro kterou je tato práce řešena v rámci Sdružení průmyslových partnerů Fakulty informatiky.

5.1.1 Spring Portlet MVC

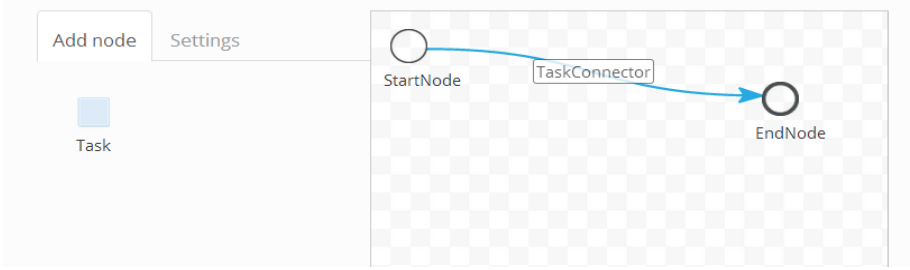
Rámec Spring Portlet MVC je založen na stejných konceptech, třídách a rozhraní jako Spring Web MVC framework [11]. Spring Portlet MVC je navržen pouze pro vývoj portletů a je možné pomocí něj vyvíjet portlety, které splňují portletové standardy JSR-168 i JSR-286. Také implementuje klasickou architekturu MVC. Ta se skládá z následujících částí. [12]

- Model (model) - je to doménově specifická reprezentace informací s nimiž aplikace pracuje.
- View (pohled) - převádí data reprezentovaná modelem do podoby vhodné k prezentaci uživateli.
- Controller (řadič) - reaguje na události a zajišťuje změny v modelu a pohledu.

Nejdůležitější roli v portletu má třída `DispatcherPortlet`, která obstarává všechny příchozí požadavky, pro které volí vhodný řadič na zpracování požadavku. Každá instance dispečeru si udržuje vlastní aplikační kontext.

5.2 Alloy UI framework

Alloy je UI meta framework, který poskytuje konzistentní a jednoduché API pro vytváření webových aplikací na třech úrovní prohlížeče: struktura, styl a chování. Je založených na moderních technologiích CSS3, HTML5 a javascriptovém frameworku YUI3. [13] V Liferay Portal od verze 6 nahradil tento framework dříve používané jQuery a staví se na něm grafické rozhraní Liferay portálu.



Obrázek 5.1: Alloy UI Diagram Builder komponenta

Pro tento framework jsem rozhodl z několika důvodů. Liferay Portal tento framework využívá pro tvorbu grafického rozhraní celého portálu a je proto

již standardní součástí portálu, což jeho využití zjendnodušuje. Jeho použitím tedy dosáhnu stejného vzhledu jako je vzhled portálů, což je také jedním z požadavků kladených na tuto aplikaci. Dalším důvodem bylo, že Alloy UI poskytuje mnoho připravených komponent, které je možno využít.

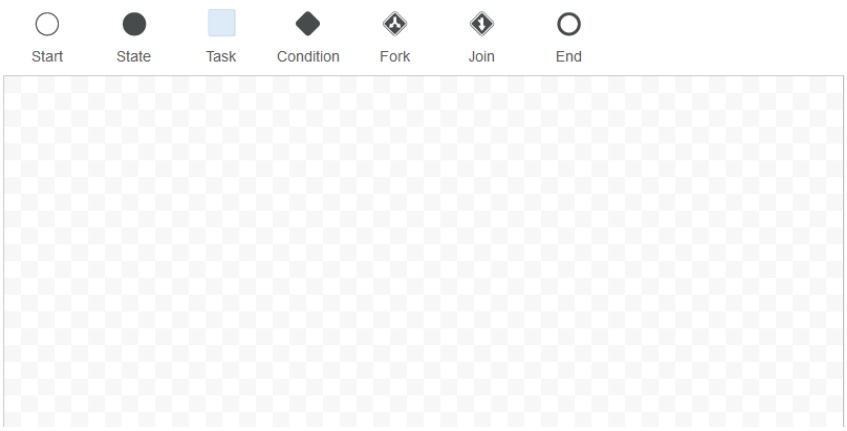
Z pohledu této práce je nejdůležitější komponentou **Diagram Builder** , viz. Obrázek 5.1. Pro potřeby aplikace byla tato komponenta rozšířena tak, aby ji bylo možné využít. Ukázka kódu 5.1 obsahuje základní postup, jak se rozšiřují komponenty z Alloy UI frameworku.

```
AUI.add('my-diagram-builder', function(A, NAME) {
    var MyDiagramBuilder = A.Component
        .create({
          NAME : DIAGRAM_NAME,
          EXTENDS : A.DiagramBuilder,

            prototype : {
              select : function(diagramNode) {
              }
            }
    });
    A.MyDiagramBuilder = MyDiagramBuilder;

}, '2.0.0', {
  requires : [ ],
});
```

Ukázka kódu 5.1: Rozšíření komponenty Diagram Builder

V části `EXTENDS : A.DiagramBuilder` je definováno, jaká komponenta je rozšiřována. V bloku `prototype` jsou umístěny přepisované či nově definované funkce. V části `A.MyDiagramBuilder = MyDiagramBuilder` je nová komponenta přidána do frameworku Allyou UI a je možné ji možné používat. Vzhled komponenty je možné ovlivnit přepsáním daných třídy v css pravidlech. Komponenta po provedených úprávách viz. Obrázek 5.2.

Další zajímavou komponentou která je využita v aplikaci je **Ace editor**. Ta se využívá pro psaní vlastních skriptů pro jednotlivé akce přiřazované jednotlivým uzlům, či definování skriptu pro podmínky při tvorbě definice workflow. Pro psaní těchto skriptů je možné využít různé skriptovací jazyky. Tuto komponentu k těmto účelům nebylo nutné nijak rozšiřovat. Je však využito možnosti nastavit tzv. mód editoru v závislosti na zvoleném skriptovacím jazyku, což pak zajišťuje barevné zvýraznění syntaxe daného jazyka. Dále byla ještě přidána při tvorbě skriptů možnost vkládat některé definové často používané funkce. Pro účely aplikace bylo vytvořeno několik funkcí v jazyku Groovy. Uživatel má také možnost přidávat své vlastní funkce. Na souborovém disku vytvoří adresář se jménem "scripts"a v něm další adresář

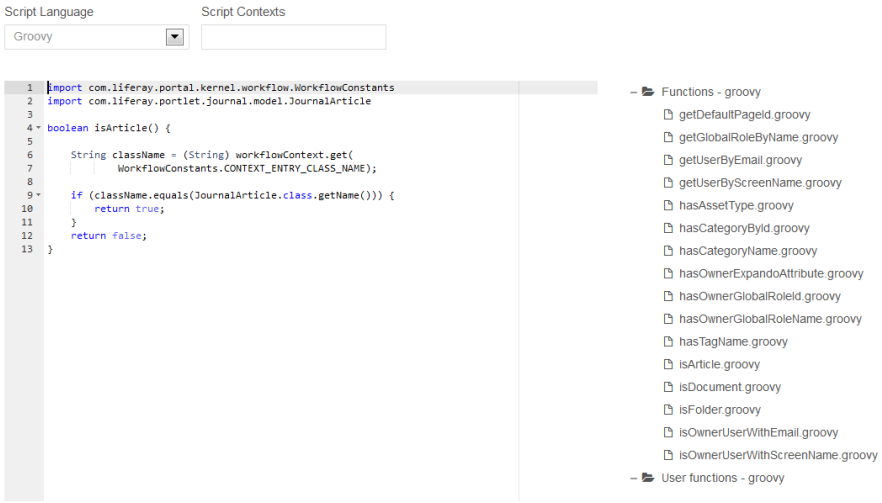


Obrázek 5.2: Alloy UI Diagram Builder komponenta po úpravách

se názvem podle scriptovacího jazyka, u kterého mají být funkce zobrazovány. Do tohoto adresáře jsou pak vkládány scripty s funkcemi. Cestu k tomuto adresáři pak musí zadefinovat v konfiguračním souboru portálu Liferay `portal-ext.properties` takto:

`cz.muni.fi.mtlacbaba.designer.scripts="cesta k adresáři"`

Pak jsou také tyto funkce načteny a k dispozici při tvorbě workflow definic. Výsledný vzhled editoru na úpravu skriptů je vidět viz. Obrázek 5.3.



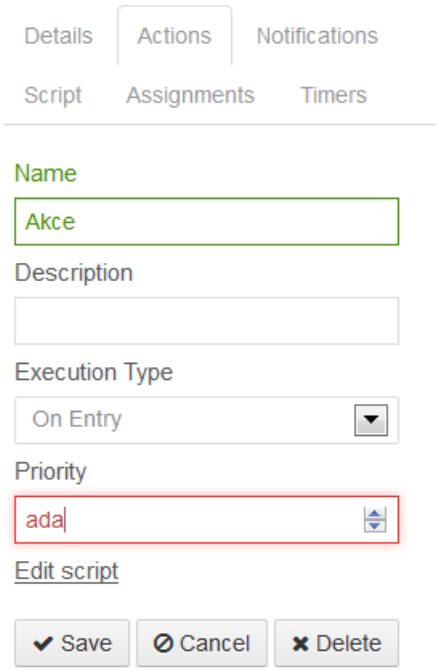
Obrázek 5.3: Ace editor použitý v aplikaci

Pro validaci formulářů byla využita další dostupná komponenta, Form Validator. Je použita pro veškeré validace vstupů od uživatele ve všech formulářích.

```
var actionValidator = new A.FormValidator({
    boundingBox: '#action-form',
    rules: {
        name: {required: true},
        priority: {digits:true}
    }
});
```

Ukázka kódu 5.2: Použití komponenty Form Validator

Ukázka kódu 5.2 ilustruje použití komponenty. V ní `rules` představuje pole omenzení pro jednotlivé pole formuláře pro zadání nové akce. Pole `name` je povinné a pole `priority` může obsahovat pouze číslice. Validator je uložen do proměnné `actionValidator` a navázán na formulář s id `action-form`. Použití v aplikaci viz. Obrázek 5.4.



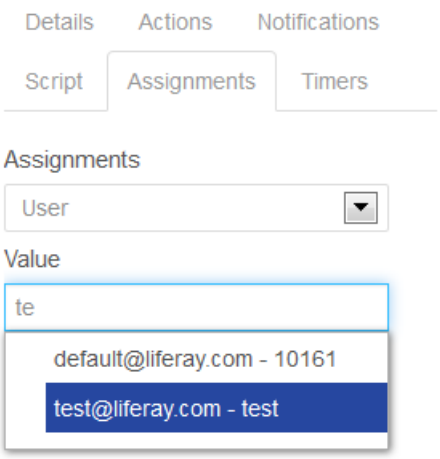
Obrázek 5.4: Form Validator

Další použitou komponentu je Autocomplete. Ukázka kódu 5.3 ukazuje

použití této komponenty. V proměnné `data` jsou data použitá pro našeptávání, která jsou posílány ze serverové části aplikace a tam jsou získáváná pomocí portálového API. Blíže bude vysvětleno zde: Kapitola 5.4. V části `resultHighlighter: 'phraseMatch'` se nastavují pravidla pro zvýrazňování fráze při výběru, v tomto případě se zvýrazňuje celá fráze. V části `resultFilters: 'charMatch'` se nastavují pravidla pro vyhledávání v nápovědě. Zde je nastaveno vyhledávání podle zadaných znaků, které slova obsahují i v různém pořadí. Použití v aplikace viz. Obrázek 5.5.

```
var data = ["default@liferay.com-10161",
                    "test@liferay.com-test"]

A.one('#role').plug(A.Plugin.AutoComplete, {
  resultHighlighter: 'phraseMatch',
     resultFilters: 'charMatch',
  source: data
});
```

Ukázka kódu 5.3: Použití komponenty autocomplete



Obrázek 5.5: Autocomplete

5.3 JAXB

K ukládání worfklow definice je využíván dokument ve formátu XML a struktura XML dokumentu používaného workflow systémem Kaleo je defino-

vaná XML schématem, označovaném jako XML Schema Definition (XSD). Je to XML dokument s pevnou strukturou obsahující omezující podmínky na strukturu a obsah XML dokumentu. Proto je možné využít klasické nástroje pro práci s XML a jedno z možných využití je generování anotovaných Java tříd popisující XML dokumenty vyhovující danému XML schématu.

K tomuto účelu jsem zvolil nástroj JAXB, jenž umožňuje tvořit návaznosti mezi Java objekty a XML schématem, díky nimž je pak možno zpracovávat XML dokumenty. Tento nástroj je založený na anotacích a poskytuje rozhraní pro serializaci a deserializaci Java objektů do/z XML dokumentů podle daných anotací. Navíc také poskytuje vyšší úroveň abstrakce než SAX [1] či DOM [2] a je součástí JDK.

Pro vygenerování sady tříd jazyka Java na základě daného schématu jsem využil kompilátor vazeb (xjc), jenž je také součástí JDK. Takto byly anotované třídy sloužící pro konvertování XML dokumentů do stromů objektů a zpět. Ukázka takto anotované třídy Ukázka kódu 5.4.

```
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "abstract-workflow-node-complex-type",
propOrder = {
    "name",
    "description",
    "metadata"
})
@XmlSeeAlso({
    Task.class,
    ActionTimerWorkflowNodeComplexType.class
})
public abstract class AbstractWorkflowNodeComplexType{

    @XmlElement(required = true)
    protected String name;
    protected String description;
    protected String metadata;
//upraveno
}
```

Ukázka kódu 5.4: Java třída s JAXB anotacemi

Z taktov vytvořených tříd je vytvořen strom objektů, který reprezentuje obsah a strukturu XML dokumentu. Pomocí těchto tříd je možné upravavot reprezentaci dokumentu a nakonec je tento strom opět převeden na XML dokument. S tímto je spojena první fáze validace workflow definice, jelikož

1. SAX je zkratka pro Simple API for XML
2. DOM je zkratka pro Document Object Model

se tímto kontroluje, zda odpovídá zadanému schématu. Tímto je pouze zaručené, že workflow definice obsahuje validní objekty, ale ještě není zaručeno, že workflow je validní jako celek. Úplná validace se provádí skrze portálové API.

5.4 Liferay Portal API

Liferay Portal API[3] poskytuje vývojářům přístup ke zdrojům v portálu. Mezi tyto zdroje například patří uživatelé, role, články, dokumenty, workflow a další. S těmito zdroji je možné pracovat skrze servisní třídy poskytující metody pro manipulaci s nimi.

V některých formulářích bylo implementováno našeptání uživatelů, rolí a workflow definicí existujících v portálu. K tomu účelu byly využity následující statické servisní třídy:

- pro uživatele (`UserLocalServiceUtil`),
- pro role (`RoleLocalServiceUtil`),
- a pro workflow definice (`KaleoDefinitionLocalServiceUtil`).

Ukázka kódu 5.5 ilustruje načtení všech existujících workflow definic v portálu.

```
 List<KaleoDefinition> definitions =
  KaleoDefinitionLocalServiceUtil
  .getKaleoDefinitions(-1, -1);
}
```

Ukázka kódu 5.5: Načtení workflow definic z portálu

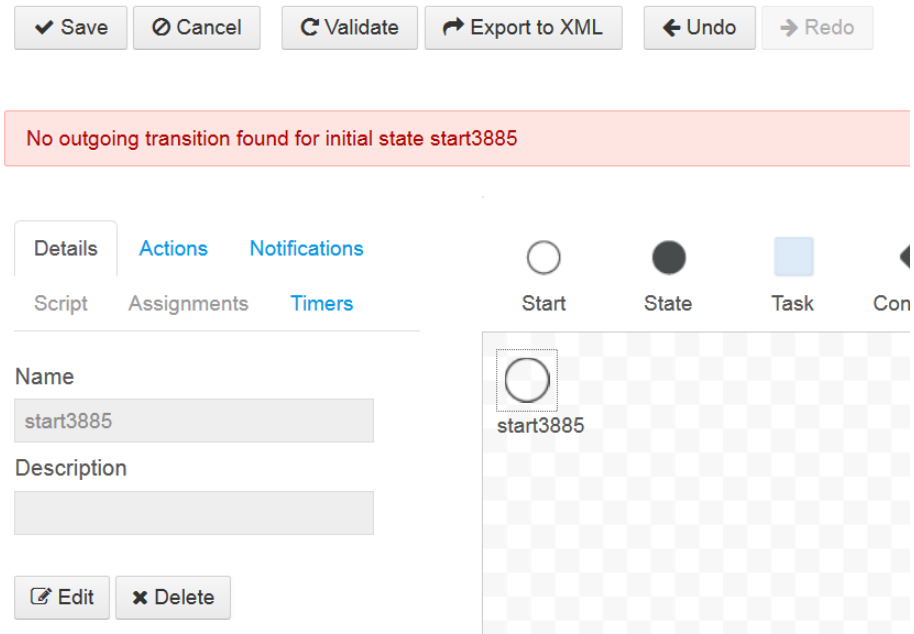
Pro vložení a validaci definice byla využita třída `WorkflowDefinitionManagerUtil`.

Ukázka kódu 5.6 ukazuje proces validace workflow. Proces validace v tomto případě probíhá následovně. Vytvořené workflow se validuje pomocí metody `validateWorkflowDefinition`. Pokud je workflow validní, tak metoda úspěšně skončí, pokud ovšem workflow validní není, tak je vyhozena vyjímka, která obsahuje důvod selhání validaci. Tato vyjímka je aplikací zachycena a důvod selhání je vrácen uživateli v podobě chybové hlášky, Obrázek 5.6

3. Zkratka pro Application Programing Interaface označuje rozhraní pro programování aplikací.

```
try {
WorkflowDefinitionManagerUtil.
validateWorkflowDefinition(xml.getBytes("utf-8"));
        } catch (WorkflowException e) {
        //zpracovani vyjimky
        }
}
```

Ukázka kódu 5.6: Validace workflow pomocí portálového API



Obrázek 5.6: Chybová hláška při validaci worklow

Kapitola 6

Testování aplikace

Testováním je možné nazvat jakoukoliv aktivitu, která odhalí, že chování programu porušuje specifikaci. Nejčastěji bývá na testování nahlíženo jako na disciplínu, jejímž úkolem je ověřování kvality. Příkladem může být definice od Kanera [14]: Softwarové testování je empirický technický výzkum kvality testovaného produktu nebo služby, prováděný za účelem poskytnutí těchto informací lidem, kteří mají zájem na dané věci. Na pojem kvalita produktu v tomto případě může být nahlíženo tak, že produkt má splnit požadavky určité potřeby, která stojí za jeho vznikem. Tedy kvalitní software je takový, který tyto potřeby uspokojuje.

Množina využitelných testů je velmi široká a proto je vždy nutné určit, jaké druhy testů budou použity, kdy a jak budou aplikovány. Webové aplikace patří do speciální skupiny softwaru a proto se zde používají určité druhy testování a ty je možné rozdělit do kategorií [15]: funkční testování, výkonnostní testování, testování použitelnosti, testování serverového rozhraní, testování kompatibility a testování bezpečnosti.

6.1 Zvolené druhy testů

Pro účely této aplikace jsem jako nejvhodnější zvolil funkční testování v kombinaci s testováním kompatability.

6.1.1 Funkční testování

Je to postup, kdy je ověřováno správné fungování aplikace nebo její části podle předem definovaného nebo očekávaného chování. Aplikace se při tomto testu prochází z pohledu uživatele a je tedy kontrolováno, zda reakce na uživatelovy akce a vstupy jsou správné a jestli aplikace podává správné výstupní informace a výsledky.

Zde je možné zahrnout kontrolu správnosti odkazování, ověřování funkčnosti formulářových prvků včetně validace vstupů a reakcí na chybný vstup.

Mezi další patří také správná funkčnost JavaScript nebo AJAX částí aplikace.

6.1.2 Testování kompatability

Protože existují různé platformy spolu s rozmanitými prohlížeči, kde mohou být webové aplikace spouštěny, je testování kompatability velmi důležité. Je tedy vždy vhodné otestovat aplikaci na nejběžnějších kombinacích systémů, aby se tak zajistila správná funkčnost aplikace pro co nejširší okruh uživatelů.

6.2 Příprava a průběh testů

Aplikace workflow editor je určena pro tvorbu workflow definic v portálu Liferay. Jejím jediným uživatelem je administrátor portálu.Testování probíhalo podle předem určených určených testových scénářů na různých prostředí a v různých webových prohlížečích. Testové scénáře pokrývají všechny důležité funkce editoru a jsou to: vytvoření nového workflow, změna vloženého workflow, exportování workflow do souboru xml, přidání nového uzlu, přidání akce k uzlu, úprava akce u uzlu, přidání oznámení k uzlu, úprava oznámení u akce, u podmínky úprava scriptu, u úlohy úprava přiřazení, přidání časovače u uzlu, úprava časovače u uzlu. Níže jsou podrobně rozepsány scénaře na změnu workflow a úpravu akce u uzlu.

6.2.1 Ukázkový testový scénář č.1 - Změna vloženého workflow

Vstupní podmínky: Uživatel je v roli Administrátora (má tedy přístup k do kontrolního panelu), je přihlášený v portálu a v sekci kontrolní panel. Uživatel již vytvořil workflow a vložil ho do portálu na základě Vytvoření nového workflow.

Scénář:

1. Přechod na stránku s workflow editorem
2. Kliknutí na odkaz "Edit"na řádku workflow vytvořeného na základě Vytvoření nového workflow
3. Přidání nového stavu kliknutím na ikonu "Nový stav"(vizuální podoba - kruh)
4. Kliknutím do kreslící plochy (tímto se umístí nový stav)
5. Kliknutím na nově vytvořený stav

6. Pak táhnutím nad cílový stav a kliknutím na něj se vytvoří nová vazba

7. Kliknutí na tlačítko "Uložit"pro uložení pozměněného workflow

6.3 Vyhodnocení testů

Testy byly provedeny na různých prostředích a v různých prohlížečích. Bylo během nich nalezeno několik drobných chyb, které byly opraveny a odstranění chyb bylo znovu následně ověřeno.

Kapitola 7

Dokumentace

Závěr

Literatura

[1] GÁLA, L. et al. *Podniková informatika*. 1. vyd. Praha: Grada Publishing, 2006.

[2] *JavaTM Portlet Specification*, The Java Community Process, dostupné na `http://www.jcp.org/en/jsr/detail?id=286` [cit. 2014-03-27].

[3] ČECH, P. *Přínos podnikových portálů pro management znalostí*, in Internet a konkurenceschopnost, Zlín, 2004.

[4] *Enterprise portal*, Wikipedia, dostupné na `http://en.wikipedia.org/wiki/Enterprise_portal` [cit. 2014-03-27].

[5] *Liferay Portal 6.2 Developer's Guide*, Liferay Inc., dostupné na `http://www.liferay.com/documentation/liferay-portal/6.2/development` [cit. 2014-04-02].

[6] *Portal Features*, Liferay Inc., dostupné na `http://www.liferay.com/products/liferay-portal/features/portal` [cit. 2014-04-02].

[7] SARIN, Ashish. *Portlets in action*, Shelter Island, NY: Manning, 2011.

[8] *Terminology & Glossary*, Workflow Management Coalition, dostupné na `http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf` [cit. 2014-04-14].

[9] CARDA, A., KUNSTOVÁ, R. *Workflow : Nástroj manažera pro řízení podnikových procesů*. 2. vyd. Praha: Grada Publishing, 2003.

[10] *Business Process Model and Notation (BPMN)*, Object Management Group, Inc., dostupné na `http://www.omg.org/spec/BPMN/2.0/` [cit. 2014-04-23]

[11] Web MVC framework. *Spring Framework Reference Documentation*, dostupné na `http://static.springsource.org/spring/docs/current/spring-framework-reference/html/mvc.html` [cit. 2014-05-04]

[12] Model-view-controller. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikipedia Foundation, 2011

[13] *http://alloyui.com/*

[14] ABRAN, A., MOORE, J.W. *Guide to the Software Engineering Body of Knowledge* [online]. IEEE Computer Society, 2004 Version, dostupné na `http://www.inf.ed.ac.uk/teaching/courses/seoc/2006_2007/resources/SWEBOK_Guide_2004.pdf` [cit. 2014-05-08]

[15] VIJAY, S. *Web Testing, Example Test cases* [online]., dostupné na `http://www.softwaretestinghelp.com/web-testing-example-test-cases` [cit. 2014-05-08]

Příloha A

Položky seznamu