Masarykova univerzita
Fakulta informatiky

Aplikácia pre správu vzdelávacích plánov na platforme podnikového portálu

Bakalárska práca

Miroslav Cupák

Brno, 2011

Prehlásenie

Prehlasujem, že táto bakalárska práca je mojím pôvodným autorským dielom, ktoré som vypracoval samostatne. Všetky zdroje, pramene a literatúru, ktoré som pri vypracovaní používal alebo z nich čerpal, v práci riadne citujem s uvedením úplného odkazu na príslušný zdroj.

Miroslav Cupák

Vedúci práce: Ing. Petr Adámek

Poďakovanie

Chcel by som poďakovať Ing. Petrovi Adámkovi, vedúcemu tejto bakalárskej práce, za trpezlivosť, ústretovosť a cenné rady pri písaní práce.

Zhrnutie

Hlavným cieľom tejto bakalárskej práce je predstaviť problematiku vývoja aplikácií na platforme podnikových portálov a popísať vytvorenú aplikáciu určenú na správu individuálnych vzdelávacích plánov v prostredí IT firmy.

Práca vysvetľuje význam a tvorbu vzdelávacích plánov zamestnancov. Popisuje základné princípy, metódy a štandardy v oblasti vývoja portletov a portálových aplikácií a využitie týchto postupov i ďalších technológií pri návrhu a implementácii kolekcie portletov umožňujúcich manipuláciu s týmito plánmi.

Práca je vypracovaná v rámci Sdružení průmyslových partnerů Fakulty informatiky pre spoločnosť IBA CZ.

Kľúčové slová

JSR-168, JSR-286, vzdelávací plán, portál, portálová aplikácia, portlet, Liferay, Spring.

Obsah

1 Úvod

Svet je riadený informáciami a rýchly prístup k nim môže byť rozdiel medzi úspechom a neúspechom nielen v obchode. Hlavne v súčasnosti však nemusí byť jednoduché včas získať relevantné informácie z množstva dát, ktoré má vďaka pokroku v oblasti informačných technológií k dispozícii takmer každý. Efektívne zvládanie tohto procesu vyžaduje nástroje schopné dáta z rôznych zdrojov získať, filtrovať a vyhodnocovať. Touto problematikou sa zaoberú aj stále populárnejšie podnikové portály.

Portály si kladú za cieľ agregovať dáta z množstva systémov a sprístupniť relevantné informácie pohodlne, z jedného miesta. Dôraz je pritom kladený na jednoduchosť integrácie s ďalšími systémami a možnosť konfigurácie funkcionality, obsahu i vzhľadu. Vďaka týmto charakteristikám má nasadenie portálu v organizácii obvykle pozitívny vplyv na efektivitu pracovníkov, zníženie nákladov a úspech organizácie ako celku, a tak čoraz viac firiem sprístupňuje zamestnancom aplikácie vo svojom intranete práve týmto spôsobom. Patrí medzi ne aj spoločnosť IBA CZ.

Niektoré záležitosti a procesy v spoločnosti sú svojim charakterom mimoriadne vhodné na to, aby bol prístup k nim riešený formou portálovej aplikácie. Zamestnancami žiadaný je hlavne prístup k artefaktom, s ktorými prichádzajú do styku denne a ktoré sa priamo týkajú ich kariéry či náplne práce. Jednou z takýchto vecí je pre zamestnanca aj jeho vzdelávací plán predstavujúci zoznam aktivít súvisiacich s jeho osobnostným a kariérnym rastom, ktorých splnenie sa očakáva do istého termínu. Jednoduchý prístup k týmto údajom je teda vhodný nielen z hľadiska motivácie, ale aj pripomenutia daných termínov.

Súčasťou tejto bakalárskej práce je okrem analýzy problematiky vývoja portálových aplikácií tiež návrh a implementácia konkrétnej aplikácie umožňujúcej správu individuálnych vzdelávacích plánov. Cieľom je vytvoriť kolekciu portletov, ktorých funkcionalita je prispôsobená procesom v spoločnosti IBA CZ. Predpokladá sa následné nasadenie aplikácie v prostredí tejto spoločnosti.

Práca začína popisom a klasifikáciou portálov v kapitole 2. Špeciálna pozornosť je venovaná podnikovým portálom a analýze výhod i nevýhod tejto technológie. V kapitole 3 predstavujeme špecifiká vývoja aplikácii v prostredí portálu s dôrazom na popis základných štandardov týkajúcich sa tejto oblasti. Nasleduje analýza významu vzdelávacích plánov a problematiky ich správy v kapitole 4. V kapitole 5 vyhodnocujeme požiadavky kladené na aplikáciu a popisujeme osobitosti tvorby plánov v spoločnosti IBA CZ. Dôraz

je kladený na špecifikáciu aplikácie a prípady použitia. Kapitola 6 popisuje celkovú architektúru aplikácie a jej dátový model. Nasleduje popis implementácie, použitých technológií a analýza možností budúceho rozšírenia aplikácie v kapitole 7. V kapitole 8 následne zhŕňame výsledky a prínos tejto práce.

2 Portálové technológie

Kapitola je venovaná portálovým technológiám a kladie si za cieľ zoznámiť čitateľa s portálmi a ich súčasným stavom. V sekcii 2.1 predstavujeme portály v najširšom význame slova a zavádzame ich klasifikáciu. V časti 2.2 sa podrobnejšie venujeme podnikovým portálom, na ktoré je táto práca zameraná. Význam portálu si pritom ukážeme na príklade. V sekcii 2.3 popisujeme hlavné výhody a nevýhody portálových technológií a v časti 2.4 si predstavíme najpopulárnejšie produkty, ktoré sa dnes v praxi používajú.

2.1 Portály

Portál je v súčasnosti pomerne všeobecný pojem používaný vo viacerých významoch. Krátky slovník slovenského jazyka ho definuje ako *architektonicky upravený vchod* [14], pričom v kontexte informačných technológií ho chápeme ako webovú aplikáciu zobrazujúcu informácie z rôznych zdrojov jednotným spôsobom [21], čiže ako akúsi bránu k informáciám a službám [19].

Aj kvôli šírke významu pojmu webový portál a prudkému vývoju tejto oblasti neexistuje v súčasnosti jednotná, všeobecne prijatá klasifikácia portálov. Najvhodnejšou sa zdá byť tá, ktorú navrhli Burgess, Davison a Tatnall v [3]. Na základe nej rozlišujeme tieto kategórie portálov (príklady zástupcov danej kategórie sú uvedené v zátvorkách za jej menom):

- všeobecné (*Google*[1] a *Yahoo*[2]) – pôvodne pokročilé vyhľadávače typicky poskytujúce aj iné služby (email, prehľad článkov zo spravodajských serverov a pod.), ktorých hlavnou úlohou je poskytovať odkazy na iné webové stránky a zdrojom príjmu predovšetkým šírenie reklám závislé na návštevnosti [18];

- komunitné (*MySpace*[3]) – portály zamerané na skupinu ľudí so spoločnými záujmami (komunitu), ktorej členovia sa na základe záujmov združujú a komunikujú;

- vertikálne (*CouchSurfing*[4]) – systémy zamerané na výmenu produktov istej kategórie;

---

1. Google, `http://www.google.com/`.
2. Yahoo, `http://www.yahoo.com/`.
3. MySpace, `http://www.myspace.com/`.
4. CouchSurfing, `http://www.couchsurfing.org/`.

- horizontálne (*BIZEWEST*[5]) – pomerne špecifické portály viazané na konkrétne druhy priemyslu či služby poskytované v danej geografickej oblasti;
- obchodné (*Amazon*[6]) – systémy, ktorých účelom je niečo predávať, pričom často hlavne sprostredkúvajú komunikáciu medzi spoločnosťami nejakej obchodnej skupiny a ich dodávateľmi;
- mobilné (*12wap*[7]) – malá a špecifická skupina portálov zameraných na mobilné zariadenia, ktorú spomíname skôr pre úplnosť;
- informačné (*ESPN*[8]) – portály poskytujúce špecifický druh informácií, ku ktorým by sme mohli zaradiť aj systémy z už spomínaných skupín;
- podnikové (*Liferay*[9]) – brány do intranetov spoločností, ktoré slúžia primárne na správu znalostí v rámci organizácie (ďalej v tejto práci budeme slovom portál označovať práve podnikové portály).

Uvedené kategórie sú skôr orientačné a nie sú disjunktné – jeden portál môže patriť do viacerých kategórií.

2.2 Podnikové portály

Podnikové portály sú prezentačnou vrstvou informačných systémov. Sú to webové aplikácie umožňujúce jednoduchý prístup k množstvu ďalších aplikácií pohodlne, z jedného miesta. Podporujú integráciu s ďalšími systémami používanými v organizácii, umožňujú získavanie dát z rôznych zdrojov, ich agregáciu a následné zobrazenie jednotným, nastaviteľným spôsobom. Hlavným cieľom organizácie pri zavádzaní portálu je spravidla zlepšenie jej fungovania či zvýšenie efektivity zamestnancov ako prezentovaním spracovaných aktuálnych informácií na jednom mieste, tak aj znížením časovej náročnosti niektorých činností.

Popísanú situáciu si môžme ilustrovať na fiktívnom príklade portálu IT spoločnosti. Na základe údajov načítaných z korporátnej adresárovej služby *LDAP*[10] je programátor po prihlásení do portálu automaticky prihlásený

5. Informácie o dnes už neexistujúcom portáli BIZEWEST, `http://what-when-how.com/portal-technologies-and-applications/the-bizewest-portal/`.
6. Amazon.com, `http://www.amazon.com/`.
7. 12wap, `http://www.12wap.net/`.
8. Športový portál ESPN, `http://espn.go.com/`.
9. Liferay, `http://www.liferay.com/products/liferay-portal`.
10. Lightweight Directory Access Protocol, `http://ldap.zdenda.com/`.

do ďalších aplikácií, s ktorými počas dňa pracuje. Na jednej stránke následne vidí prehľad naplánovaných stretnutí načítaných zo systému *Zimbra*[11], najnovšie súkromné emaily získané zo schránky služby *Gmail*[12] a obedové menu načítané z webovej stránky jeho obľúbenej reštaurácie. Vedľa je zobrazený stav dôležitého projektu zo systému *JIRA*[13] a zoznam nájdených chýb v programátorom vyvíjanej aplikácii z nástroja *Bugzilla*[14]. Manažér má po prihlásení k dispozícii údaje o svojich podriadených získané z databázy zamestnancov, stav firemných akcií získaný z internetu či rôzne analýzy dát dôležité pre obchodné rozhodnutia zo systému *Cognos BI*[15]. Asistentka má k dispozícii aktuality načítané pomocou *RSS*[16], kalendár a pod. Aj takto by mohol vyzerať dobre fungujúci portál IT firmy.

Z pohľadu organizácie môžu jej portál využívať zamestnanci (*portál B2E, Business-to-Employee*), zákazníci (*portál B2C, Business-to-Customer*), dodávatelia a obchodní partneri firmy (*portál B2B, Business-to-Business*). Niekedy ako samostatný typ vyčleňujeme portál vo verejnom sektore (*G2C, Government-to-Constituent*).

2.3 Výhody a nevýhody

Portály ako koncept majú zaujímavé znaky, ktoré činia ich zavedenie do infraštruktúry firmy výhodným. V nasledujúcom zozname analyzujeme kľúčové vlastnosti a ponúkané služby:

- Prispôsobiteľnosť – užívateľ môže portál konfigurovať z hľadiska vzhľadu, obsahu i funkcionality. To znamená, že si v rámci svojich prístupových práv môže zvoliť, aké aplikácie chce vidieť, ako majú byť rozmiestnené a ako majú vyzerať. Väčšina portálových aplikácií pritom umožňuje nejakým spôsobom nastaviť aj obsah, ktorý má byť zobrazený – napríklad aplikácia zobrazujúca najnovšie správy by typicky umožnila nastaviť adresu spravodajského serveru, z ktorého sú získavané.
- Integrovateľnosť – portál je navrhnutý tak, aby ho bolo možné integrovať s ďalšími systémami, čo je kľúčové pri jeho zavádzaní do organizácie.

11. Zimbra Collaboration Suite, `http://www.elostech.cz/zimbra/`.
12. Gmail, `https://mail.google.com/`.
13. Atlassian JIRA, `http://www.atlassian.com/software/jira/`.
14. Bugzilla.org, `http://www.bugzilla.org/`.
15. Cognos Business Intelligence and Financial Performance Management Software, `http://www-01.ibm.com/software/data/cognos/`.
16. Rich Site Summary, `http://www.whatisrss.com/`.

- Centrálne prihlasovanie (*single sign-on*) – pri prihlásení užívateľa do portálu prebehne jeho prihlásenie do všetkých systémov integrovaných s portálom automaticky. Okrem ušetrenia času pri pravidelne opakovanej činnosti je hneď zrejmá ďalšia výhoda – užívateľ si potrebuje pamätať len jednu skupinu prihlasovacích údajov.

- Jednotné rozhranie k aplikáciám – portál sa istým spôsobom stará o aplikácie na ňom bežiace a umožňuje k nim pristupovať jednotným spôsobom.

- Jednoduchá správa práv a užívateľov – portály umožňujú užívateľom priraďovať roly či členstvá v skupinách, na základe čoho je možné kontrolovať prístup.

- Množstvo vstavaných služieb – portály spravidla prinášajú balík služieb, ktoré môžu vývojári využiť. Typickým príkladom by mohlo byť prihlasovanie, navigácia, vyhľadávanie.

- Rozšíriteľnosť – portály majú zväčša veľmi modulárnu architektúru a sú jednoducho rozšíriteľné pomocou nových aplikácií, rozšírení a zásuvných modulov.

Samozrejme portály nepredstavujú dokonalú technológiu a je potrebné spomenúť ich hlavné nedostatky a problémy, ktoré s nimi bývajú často spojené:

- Komplikovanejší vývoj aplikácií – napriek tomu, že portály už prichádzajú s funkciami, ktoré môžu vývojári vo svojich aplikáciách využiť, vytvorenie aplikácie nemusí byť vždy úplne jednoduché. Dôvodom je komplikovanejší model komunikácie a ukladania dát v portálových aplikáciách (viď kapitolu 3), ako aj problémy s podporou rámcov pre vývoj webových aplikácií. Ďalším problémom je neznalosť okolia aplikácie z pohľadu jej vývojára – je ťažké predvídať, kde si užívateľ aplikáciu v portáli umiestni a čo pri nej bude.

- Náročnosť na výkon – častým argumentom v neprospech portálov je ich náročnosť na výkon serverov. O tomto argumente by sa dalo diskutovať, pretože často môže byť slabý výkon spôsobený chybou vývojára konkrétnej aplikácie. Je ale očakávateľné, že keďže si môže každý užívateľ nastaviť portál a aplikácie v ňom bežiace podľa svojho vkusu, je obsluha takýchto individuálnych požiadavkov z pohľadu serveru náročnejšia ako obsluha staticky nastavenej webovej aplikácie.

- Zložitosť zavedenia do infraštruktúry spoločnosti – napriek tomu, že portály sú navrhované tak, aby ich bolo možné integrovať medzi ostatné systémy používané vo firmách, takáto integrácia nemusí byť vždy jednoduchá. Ako príklady prekážok spomeňme možnú nutnosť vývoja istých vlastných komponentov, potrebu vynaložiť isté financie na zoznámenie zamestnancov s portálom a motivovať ich k jeho používaniu či potrebu mierne prispôsobiť procesy fungujúce v spoločnosti. Aj z tohto dôvodu býva portál vhodným riešením skôr pre podniky strednej a väčšej veľkosti.

2.4 Konkrétne riešenia

Na trhu existuje v súčasnosti vyše 40 portálových riešení [21]. Aj keď portály sú si do značnej miery podobné, pretože implementujú tie isté štandardy (základné z nich si predstavíme v kapitole 3), prinášajú obvykle aj ďalšie funkcie, ktoré štandardmi pokryté nie sú. V takomto prípade poskytujú vlastné aplikačné rozhranie na prístup k mnohým z týchto funkcií a je len na vývojárovi, či sa rozhodne ich využiť. Dôsledkom je samozrejme následná závislosť aplikácie na výrobcovi a znížená prenosnosť. Najpopulárnejšou platformou pri portáloch je *Java EE*[17], ale existujú aj implementácie postavené nad skriptovacím jazykom *PHP*[18] či platformou *.NET*[19].

Veľmi žiadaným riešením na trhu je *IBM WebSphere Portal*[20], ktorý sa vyznačuje množstvom veľmi užitočných podporných nástrojov pre vývoj a správu portletov (napr. *WebSphere Portlet Factory*[21]). Známy je tiež *JBoss EPP*[22], ktorý sa stáva v poslednej dobe populárnym aj vďaka veľmi šikovnému rozšíreniu na pokročilú správu webového obsahu – *Site Publisher*[23]. Z produktov s otvoreným zdrojovým kódom sú populárne portály *GateIn*[24] a Liferay, pričom práve pre posledný menovaný bola primárne vytvorená aplikácia predstavená v tejto práci a budeme sa mu podrobnejšie venovať v kapitole 7, sekcii 7.2.3.

---

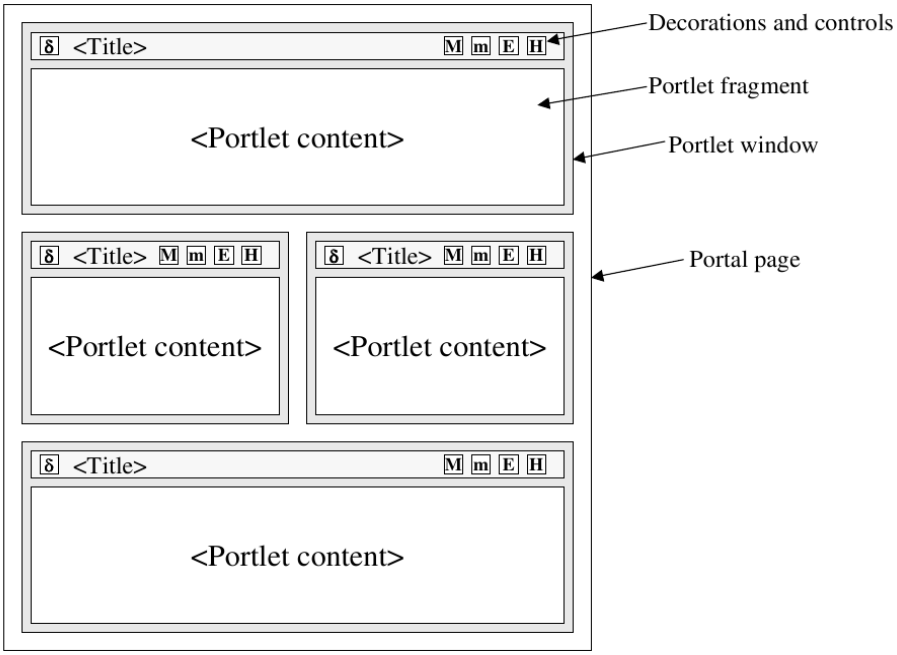
17. Java Platform, Enterprise Edition, `http://www.oracle.com/technetwork/java/javaee/`.
18. PHP: Hypertext Preprocessor, `http://www.php.net/`.
19. Microsoft .NET Framework, `http://www.microsoft.com/net/`.
20. IBM WebSphere Portal, `http://www-01.ibm.com/software/websphere/portal/`.
21. WebSphere Portlet Factory, `http://www-01.ibm.com/software/genservers/portletfactory/`.
22. JBoss Enterprise Portal Platform, `http://www.jboss.com/products/platforms/portals/`.
23. Site Publisher, `http://www.redhat.com/f/pdf/RH_JBoss_SitePublisher_web.pdf`.
24. GateIn Portal, `http://www.jboss.org/gatein/`.

3 Vývoj portletov

Táto kapitola popisuje špecifické vlastnosti vývoja portálových aplikácií. V sekcii 3.1 vysvetľujeme, čo sú portlety a v sekciách 3.2, 3.3 a 3.4 popisujeme ich základné vlastnosti dané najdôležitejšími štandardmi. V časti 3.5 stručne vysvetlíme štruktúru portálovej aplikácie.

3.1 Portlet

Portlet je malá webová aplikácia určená pre nasadenie na portálový server. Je to webový komponent, ktorý spracúva požiadavky a na základe nich generuje dynamický obsah [1]. Samotná portálová aplikácia môže byť zložená z jedného i viacerých samostatných či navzájom komunikujúcich portletov, pričom jeden portlet zvyčajne rieši len jednu konkrétnu úlohu či prípad použitia. Portlety sú v portáli umiestnené na stránkach. Každá stránka môže obsahovať niekoľko rôznych portletov, často dokonca aj niekoľko inštancií toho istého portletu.



Obrázok 3.1: Schéma portálovej stránky. Zdroj: [1].

3.2 JSR-168

Špecifikácia *JSR-168 – Portlet Specification 1.0* [1] vznikla v roku 2003 ako prvý oficiálny pokus o vytvorenie štandardu pre portály, ktoré sa dovtedy výrazne líšili od k výrobcu k výrobcovi. Cieľom bolo získať možnosť vytvárať portlety, ktoré môžu bežať na ľubovoľnom portáli prakticky bez akýchkoľvek úprav. Špecifikácia popisuje terminológiu a viacero aspektov tvorby portletov a komponentov portálu, jej najdôležitejšie časti sú predstavené v nasledujúcich sekciách.

3.2.1 Portlety a servlety

Portlety sú veľmi podobné iným webovým komponentom spravovaným kontajnerom – *servletom*. Servlety sú kompilované triedy, ktoré môžu byť dynamicky spúšťané webovým serverom. S portletmi majú niekoľko spoločných vlastností [1]:

- Sú to komponenty založené na platforme *Java*[1].
- Generujú dynamický obsah.
- Ich životný cyklus má na starosti kontajner.
- Komunikujú s klientom na základe modelu požiadavok/odpoveď.

Pre portlety platia, na rozdiel od servletov, nasledujúce tvrdenia [1]:

- Negenerujú kompletné stránky, len fragmenty spájané do výslednej stránky portálom.
- Nie sú priamo viazané na *URL*[2].
- Komunikácia s klientmi prebieha cez portál.
- Spôsob spracovania požiadavkov je členitejší – existuje dekompozícia na úrovni spracovania a vykresľovania.
- Majú preddefinované módy a stavy okna, ktoré naznačujú, čo portlet práve robí.
- Môžu v portáli existovať veľakrát.

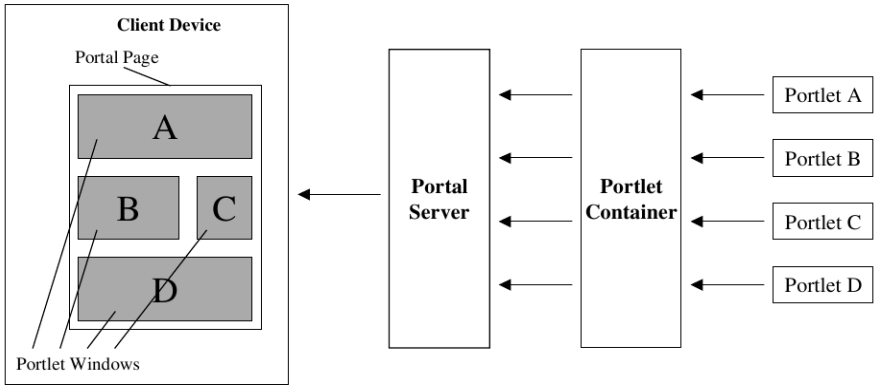
---

1. Java, `http://www.java.com/`.
2. Uniform Resource Locator (jednoznačná adresa zdroja), `http://en.wikipedia.org/wiki/Uniform_Resource_Locator`.

- Majú viac možností uloženia dát v relácii (*session*).
- Nemajú možnosť nastavovať kódovanie a *HTTP hlavičky*[3] na odpovedi.

3.2.2 Portletový kontajner

Portletový kontajner spúšťa portlety a stará sa o ich životný cyklus. Môže byť samostatným komponentom alebo priamou súčasťou portálu. Nestará sa o agregáciu obsahu vytvoreného portletmi, to je úlohou portálu. Kontajner len z portálu prijíma požiadavky a spúšťa ich na portletoch, ktoré v ňom bežia. Tým zároveň poskytuje istú formu trvalého úložiska dát pre ich nastavenia [1].



Obrázok 3.2: Schéma vzťahu portálu a portletového kontajnera. Zdroj: [1].

3.2.3 Životný cyklus portletu

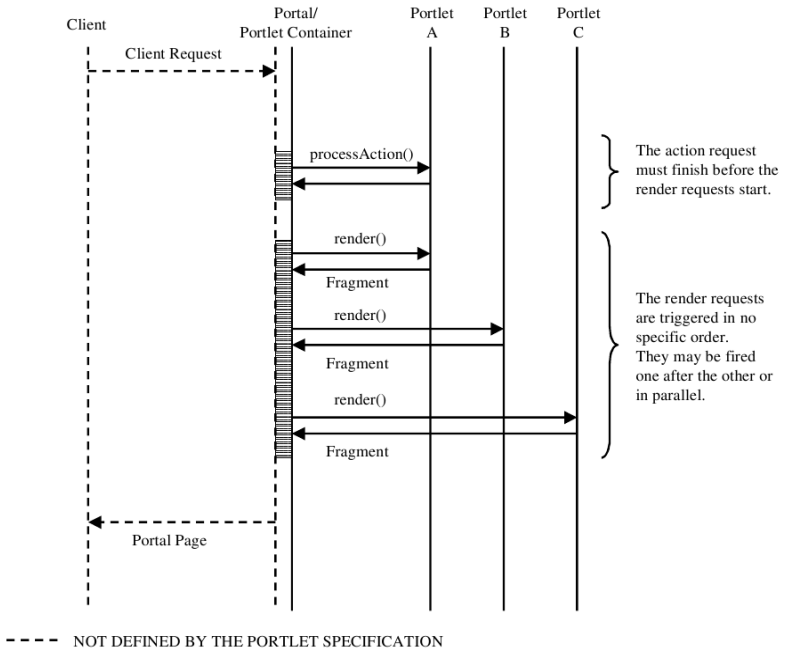
JSR-168 definuje fázy životného cyklu cez štyri metódy rozhrania `Portlet`:

- `init()` – vytvorenie inštancie, ku ktorému môže dôjsť po spustení kontajnera či pri prvom požiadavku;
- `processAction()` – spracovanie požiadavku, metóda je volaná pred volaním metódy `render()` a len na portletoch, ktoré sa na spracovaní akcií z požiadavkov priamo podieľajú;

3. HTTP/1.1 Header Field Definitions, `http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html`.

- `render()` – generovanie obsahu, metóda môže byť na portletoch zavolaná kedykoľvek je potrebné získanie obsahu pre portálovú stránku;
- `destroy()` – ukončenie životného cyklu typicky súvisiace s uvoľnením zdrojov využívaných portletom, po zavolaní tejto metódy už kontajner danú inštanciu nebude využívať na spracovanie požiadavku.

Štandardnou URL adresou je adresovateľná portálová stránka, nie jednotlivé portlety. Často je ale potrebné, aby mal portlet k dispozícii svoju adresu, na základe ktorej môžme formulovať požiadavky klienta – takéto adresy nazývame *portletové URL*. Portletové URL sú dvojakého typu – *URL na akciu* (*action URL*) a *URL na vykreslenie* (*render URL*). Môžu byť bezpečné (*HTTPS*[4]) a obsahovať aj módy portletu a stav okna (viď kapitolu 3, sekcie 3.2.4 a 3.2.5).



Obrázok 3.3: Schéma spracovania požiadavkov podľa JSR-168. Zdroj: [1].

V prípade, že je požiadavok klienta spustený pomocou URL na akciu, portletový kontajner zavolá metódu `processAction()` na príslušnom portlete. Po jej skončení následne zavolá metódu `render()` na každom portlete umiestnenom na aktuálnej portálovej stránke (výnimku môžu predstavovať

4. Hypertext Transfer Protocol Secure, `http://en.wikipedia.org/wiki/HTTP_Secure`.

portlety, ktorých obsah je súčasťou vyrovnávacej pamäte). Ak je požiadavok spustený pomocou URL na vykreslenie, metóda `processAction()` nesmie byť zavolaná na žiadnom z portletov na portálovej stránke a metóda `render()` je volaná na každom z nich, opäť s možným zohľadnením obsahu vyrovnávacej pamäte [1].

3.2.4 Módy

Portlety na rozdiel od servletov disponujú *módmi*, ktoré sú indikáciou toho, čo portlet aktuálne vykonáva – majú teda funkciu rady, aký obsah je potrebné zobraziť metódou `render()`. Určenie módu portlet prijíma od portletového kontajnera, ale môže ho aj sám programovo zmeniť pomocou metódy `processAction()`. Možnosť prepnúť módy pritom môže byť užívateľovi obmedzená vzhľadom na jeho prístupové práva [1].

Špecifikácia definuje tri základné módy:

- `VIEW` – očakávané správanie portletu je generovanie značkovania odzrkadľujúceho aktuálny stav portletu. Tento mód by sme mohli označiť ako základný a každý portlet ho musí podporovať. Implementujeme ho prekrytím metódy `doView()` z triedy `GenericPortlet`.
- `EDIT` – užívateľ si môže prispôsobiť správanie portletu zmenou jeho nastavení. Implementujeme ho prekrytím metódy `doEdit()` z triedy `GenericPortlet`.
- `HELP` – portlet by mal zobraziť informácie o sebe. Mód je implementovaný prekrytím metódy `doHelp()` z triedy `GenericPortlet`.

Portlet pritom nemusí implementovať všetky módy a môže definovať aj módy vlastné.

3.2.5 Stavy okna

Okno portletu vytvára portletový kontajner a predstavuje konkrétny výskyt portletu na portálovej stránke – jedna portálová stránka teda môže často obsahovať aj viac okien zviazaných s tým istým portletom.

Stav okna je indikátor množstva miesta na portálovej stránke, ktoré bude mať obsah generovaný portletom k dispozícii. Informáciu o stave okna dostáva portlet od portletového kontajnera a môže sa na základe nej rozhodnúť, koľko obsahu bude generovať. Podobne ako v prípade módov, aj stav okna môže portlet programovo zmeniť sám.

Špecifikácia definuje tri základné stavy okien:

- `NORMAL` – používa sa ako indikátor toho, že portlet môže zdieľať stránku s inými portletmi, resp. cieľové zariadenie má limitované možnosti zobrazovania údajov na obrazovke. Portlet by nemal zobrazovať obrovské množstvo údajov.
- `MAXIMAZED` – znamená, že portlet môže byť jediným vykresľovaným portletom na stránke, resp. má k dispozícii viac miesta v porovnaní s ostatnými zobrazovanými portletmi. Typicky pri tomto stave okna portlet generuje bohatší obsah.
- `MINIMIZED` – pri tomto stave by mal portlet generovať minimálny, príp. žiadny výstup.

Portály môžu definovať aj iné, vlastné stavy okien, ktoré môžu následne portlety na nich nasadené využívať.

3.2.6 Preferences

Portlety sú typicky konfigurovateľné z hľadiska správania a zobrazovania údajov. Základná konfigurácia býva reprezentovaná ako perzistentná množina usporiadaných dvojíc (kľúč, hodnota) a JSR-168 ju označuje pojmom *portletové nastavenia* (*portlet preferences*). Výhodou je, že tieto nastavenia sú špecifické vzhľadom k užívateľovi, o ich načítanie a ukladanie sa stará portletový kontajner a s portletom sú spojené za behu v podobe objektu daného rozhraním `PortletPreferences`. Východzie hodnoty sú štandardne získané z deskriptora portálovej aplikácie a môžu byť portletom menené v metóde `processAction()`.

3.3 JSR-286

Špecifikácia JSR-168 nepokryla dostatočne všetky potrebné aspekty portletov. Nedostatky boli stále v komunikácii medzi portletmi, obsluhovaní alternatívnych zdrojov (obrázky a pod.), podpore udalostí a reakcií na ne, podpore rámcov pre tvorbu webových aplikácií, používaní technológie AJAX[5], podpore transformácií informácií v požiadavku a odpovedi portletmi za behu (*portletové filtre*) atď. Preto bolo v roku 2007 vytvorené rozšírenie JSR-168 nazvané *JSR-286 – Portlet Specification 2.0* [10]. Najdôležitejšie zmeny zavedené v tomto dokumente sú popísané v nasledujúcich sekciách.

---

5. Asynchronous JavaScript and XML, `https://developer.mozilla.org/en/AJAX/Getting_Started`.

3.3.1 Medziportletová komunikácia

Špecifikácia popisuje tri základné mechanizmy pre podporu komunikácie medzi portletmi:

- Zdieľanie dát medzi rôznymi časťami tej istej webovej aplikácie pomocou relácie.

  Aby bolo možné vytvárať efektívne portálové aplikácie, je potrebné mať možnosť istým spôsobom spájať požiadavky od toho istého klienta (typickým príkladom použitia je autentizácia). Na tento účel slúži rozhranie `PortletSession` definované v JSR-168, ktoré umožňuje sledovať reláciu. Každá portálová aplikácia má svoj vlastný objekt `PortletSession` pre každého z aktuálnych užívateľov, pričom portletový kontajner sa musí postarať o to, aby atribúty tohto objektu neboli zdieľané medzi rôznymi aplikáciami.

  Portlet môže pripojiť k objektu `PortletSession` atribúty s dvojakým rozsahom platnosti – `PORTLET_SCOPE` a `APPLICATION_SCOPE`. Pri prvom menovanom sú objekty dostupné oknu portletu, z ktorého boli vytvorené (nie sú pred ostatnými komponentami nijako bezpečne skryté, sú len viazané ku konkrétnemu mennému priestoru). Pri druhom menovanom sú atribúty dostupné aj všetkým ostatným portletom z danej portálovej aplikácie. Práve to môžme považovať za základnú možnosť komunikácie spomenutú už v JSR-168.

- *Verejné parametre* (*public render parameters*).

  Verejné parametre zavedené v JSR-286 slúžia na zdieľanie stavu vykresľovania medzi portletmi, dokonca aj medzi portletmi z rôznych aplikácií. Pri použití sa parametre vyskytnú v URL, prístup k nim musí portlet definovať vo svojom deskriptore. Verejné parametre sú dostupné vo všetkých fázach životného cyklu portletu a môžu byť čítané a menené inými portletmi.

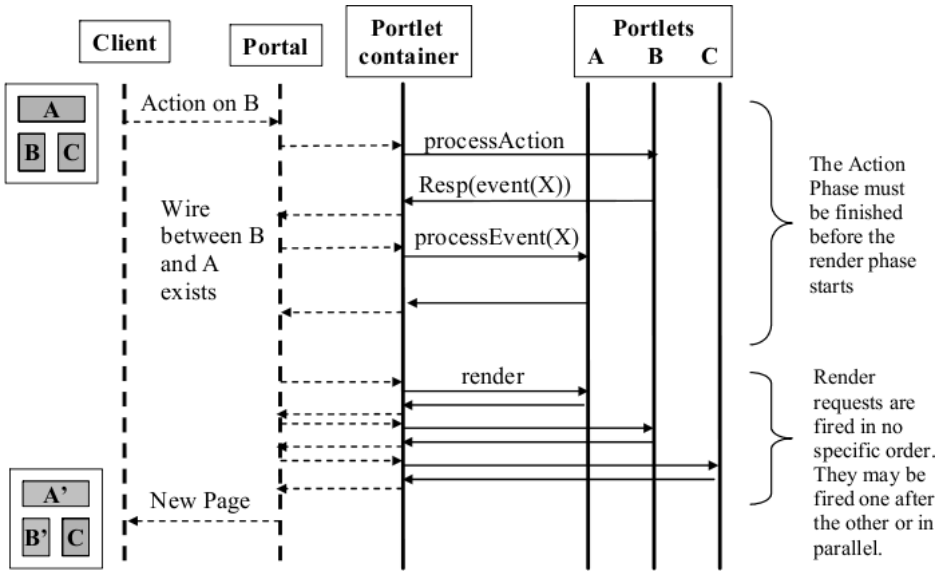
- *Udalosti*, ktoré môže portlet prijímať a odosielať (*events*).

  Udalosti predstavujú najzložitejší a najflexibilnejší spôsob medziportletovej komunikácie zavedený v JSR-286. Umožňujú portletom reagovať aj na zmeny stavu, ktoré priamo nesúvisia s interakciou užívateľa s portletom. Ich správne doručenie nie je zaručené, nepredstavujú teda náhradu zasielania správ v štýle napr. *JMS*[6].

6. Java Message Service, `http://jcp.org/aboutJava/communityprocess/final/jsr914/`.

Udalosti, ktoré chce portlet posielať a prijímať by mali byť deklarované v jeho deskriptore (s výnimkou udalostí definovaných priamo samotným portálom). Pokiaľ portlet nastaví udalosť, ktorá deklarovaná nebola, nemusí ju portál rozoslať iným portletom správne. Udalosti sú viazané na konkrétny požiadavok a v rámci životného cyklu sú spracovávané pred vykreslením v metóde `processEvent()` deklarovanej v rozhraní `EventPortlet`.

Okrem toho je samozrejme možné donútiť portlety komunikovať inou cestou nedefinovanou v portletovej špecifikácii. Jedno z riešení je použiť *JavaScript*[7], čo je vhodné hlavne v prípade, keď môžme predpokladať, že užívateľ má JavaScript v prehliadači povolený a komunikujúce portlety sú aktuálne zobrazené. Dáta môžme tiež zdieľať v nejakom externom úložisku, napríklad relačnej databáze.



Obrázok 3.4: Schéma spracovania požiadavkov podľa JSR-286. Zdroj: [10].

3.3.2 Poskytovanie zdrojov

Potreba dynamicky vytvárať zdroje (napr. obrázky či *PDF*[8] súbory) nie je pri webových aplikáciách zriedkavá. Zatiaľ čo pri bežných webových apli-

7. JavaScript, `https://developer.mozilla.org/en/JavaScript`.
8. Portable Document Format Reference, `http://www.adobe.com/devnet/pdf/pdf_reference.html`.

káciách nie je problém generovať zdroj na základe URL a nastavenia typu odpovedi v HTTP hlavičke, portlety nie sú pomocou URL jednoducho adresovateľné (resp. môžu získať len adresu na seba) a je potrebné použiť iný mechanizmus.

Použitie URL na akciu nie je ideálne, pretože je spojená so spustením akcie. Podobne URL na vykreslenie je určená výhradne na vykreslenie. Preto JSR-286 zavádza *URL zdroja* (*resource URL*). Príslušný zdroj potom môžme generovať v metóde `serveResource()` zo základnej triedy `GenericPortlet` (resp. rozhrania `ResourceServingPortlet`) na základe identifikátora zdroja (`ResourceID`). To je mimoriadne výhodné v spojení s technológiou JavaScript (resp. AJAX) – napríklad našepkávanie pri vypĺňaní políčka vo formulári na stránke môžme jednoducho napojiť na URL zdroja, ktorý nám bude pri doplnení znaku dynamicky vracať relevantné záznamy.

3.4 Ďalšie štandardy

JSR-168 a JSR-286 predstavujú základné, nie však jediné štandardy týkajúce sa portletov. Medzi ďalšie dôležité štandardy priamo súvisiace s vývojom portálových aplikácií patria napr.:

- J*SR-301*[9] a *JSR-329*[10]

  Štandardy predstavujú *Portlet Bridge špecifikáciu 1.0* (JSR-301) a *2.0* (JSR-329) *pre JSF*[11] *1.2*.

  Portlet Bridge je technológia umožňujúca vývojárom tvoriť aplikácie pre portál prakticky bez znalosti aplikačného rozhrania pre portlety. Jedná sa o veľmi komplikovaný mechanizmus založený na princípe prekladača medzi použitým programovým modelom a portálovým prostredím. JSR-301 a JSR-329 sú zamerané na populárny rámec JSF.

  Tieto štandardy sú v súčastnosti podporované väčšinou populárnych portálov. Typicky je v týchto portáloch možné s JSF využívať aj ďalšie súvisiace technológie, napr. *RichFaces*[12] a *Seam*[13]. To tvorí z Portlet Bridge veľmi silný nástroj.

---

9. Portlet 1.0 Bridge Specification for JavaServer Faces 1.2, `http://www.jcp.org/en/jsr/detail?id=301`.
10. Portlet 2.0 Bridge for JavaServer Faces 1.2 Specification, `http://www.jcp.org/en/jsr/detail?id=329`.
11. JavaServer Faces, `http://javaserverfaces.java.net/`.
12. RichFaces Project, `http://www.jboss.org/richfaces`.
13. Seam Framework, `http://seamframework.org/`.

- *WSRP 1.0*[14] a *WSRP 2.0*[15]

  Web Services for Remote Portlets definuje protokol založený na princípe webových služieb a návrhovom vzore producent/konzument, ktorý umožňuje agregáciu obsahu a interaktívnych webových aplikácií zo vzdialených zdrojov. V praxi to znamená, že v portáli môžme vzdialene využívať portlety bežiace na iných serveroch. Môžme tak prinútiť viac portálov používať ten istý portlet či portlety postavené nad inou platformou, ktoré pracujú napr. s cudzími databázami, pričom koncový užívateľ má pocit, že používa bežný, lokálne inštalovaný portlet.

3.5 Štruktúra portálovej aplikácie

Portálová aplikácia je svojou štruktúrou veľmi podobná webovej aplikácii. Okrem portletových tried implementujúcich rozhranie `Portlet` sú dôležitou súčasťou deskriptory umiestnené typicky v adresári *WEB-INF*. Základom je deskriptor webovej aplikácie v súbore *web.xml* a deskriptor portálovej aplikácie v súbore *portlet.xml*. Práve tu sa nachádzajú základné informácie o portletoch – meno, popis, podporované módy, udalosti, verejné parametre, inicializačné parametre, konfigurácia nastavení, podporované jazyky a pod. Existujú aj ďalšie deskriptory špecifické pre portál, príp. aplikačný rámec.

14. Web Services for Remote Portlets Specification 1.0, `http://www.oasis-open.org/committees/download.php/3343/oasis-200304-wsrp-specification-1.0.pdf`.
15. Web Services for Remote Portlets Specification 2.0, `http://www.oasis-open.org/committees/download.php/18617/wsrp-2.0-spec-pr-01.html`.

4 Vzdelávacie plány

V tejto kapitole prinášame analýzu problematiky vzdelávacích plánov. Po vysvetlení významu vzdelávacích plánov v organizácii (sekcia 4.1) pokračujeme popísaním ako individuálneho (sekcia 4.2), tak aj taktického a strategického plánu (sekcia 4.3). Následne prinášame prieskum existujúcich nástrojov zaoberajúcich sa správou vzdelávacích plánov (sekcia 4.4).

4.1 Motivácia

Vzdelávacie plány či plány kariérneho rozvoja zamestnancov hrajú dôležitú rolu vo fungovaní prakticky každej firmy a IT spoločnosti nie sú výnimkou. Predstavujú zoznam aktivít viac či menej súvisiacich s pracovnou náplňou zamestnanca a sú zostavované pre každého člena spoločnosti individuálne na isté pevne stanovené obdobie.

Vzdelávacie plány sú dôležité pre zamestnanca i samotnú firmu a majú niekoľko základných funkcií:

- Zamestnanec získa presnú predstavu o tom, čo sa od neho vo firme očakáva a v akom časovom intervale by mal tieto očakávania naplniť, je motivovaný.
- Záujmy a ciele ľudí sa menia a zamestnanec môže priebežne ovplyvniť, čo bude robiť a ktorým smerom sa bude jeho kariéra uberať.
- Zamestnanec sa dozvie, kam smeruje organizácia ako celok.
- Zamestnávateľ získa predstavu o tom, čomu sa chce zamestnanec v budúcnosti vo firme venovať.
- Zamestnávateľ môže zhodnotiť, do akej miery sa zamestnancovi podarilo splniť očakávania a na základe toho sa rozhodnúť, či nebude vhodné prijať na danú pozíciu ďalších ľudí, nájsť pre zamestnanca vhodnejšiu náplň práce a pod.
- Zamestnávateľ môže pomocou plánov zamestnancov sledovať, ovplyvňovať a odhadnúť celkové náklady na vzdelávanie.
- Na základe plnenia individuálnych plánov môže zamestnávateľ získať zaujímavé štatistiky nielen vo vzťahu k financiám a výkonu zamestnancov, čo je dobrým predpokladom pre správne rozhodovanie a plánovanie.

- Procesy súvisiace so správou plánov zaručujú, že manažér i zamestnanec budú mať pravidelné stretnutia, na ktorých môžu obe strany získať dôležitú spätnú väzbu.

4.2 Individuálny vzdelávací plán

Individuálny vzdelávací plán (*IEP*[1]) môže byť zostavovaný v rôznych firmách na obdobia rôznej dĺžky, jedná sa ale typicky o pravidelné intervaly. Bežná dĺžka intervalu býva šesť mesiacov alebo jeden (fiškálny) rok, ale často je ešte okrem toho formulovaný aj dlhodobejší plán, zväčša na obdobie od troch do piatich rokov. Výnimkou môže byť prvý plán zamestnanca vytvorený krátko po jeho nástupe do firmy. Keďže je vtedy zamestnanec vo firme nový a obvykle ešte nemá presnú predstavu o tom, čo vo firme bude robiť, ako dobre to bude zvládať a aká bude jeho spokojnosť s prácou, je iniciálny plán často zostavený na kratšie obdobie – typicky jeden až tri mesiace.

Plán je formulovaný na základe rozhovoru zamestnanca s manažérom. Obsahuje kľúčové položky, za ktoré je zamestnanec zodpovedný a ktoré by mal v danom období dosiahnuť v súlade s plánom organizácie ako celku. Okrem týchto hlavných vecí zahŕňa aj ďalšie aktivity, ktoré chce zamestnanec vykonať pre seba a svoj kariérny rast (napr. absolvovanie nejakých *soft skills* školení). Je dôležité, aby k tvorbe plánov pristúpili manažér i zamestnanec zodpovedne, vyjadrili svoj názor a sformulovali dosiahnuteľné ciele, s ktorými budú obe strany spokojné. Plán by mal byť zaznamenaný tak, aby bolo zachované súkromie zamestnanca (je vhodné, ak zamestnanci nemajú prístup k plánom svojich kolegov) a aby k nemu následne mali obe strany prístup, prípadne ho mohli ovplyvniť. Po uplynutí stanoveného obdobia dochádza k ďalšiemu stretnutiu, zhodnoteniu uplynulého plánu a formulácii nových cieľov na nasledujúce obdobie.

Vzdelávacie plány sú obvykle súčasťou zložitejších procesov a v závislosti od spoločnosti okolo nich existuje nejaká schéma (*workflow*). Jednotlivé položky vyžadujú schválenie a alokáciu potrebných zdrojov, preto v procese tvorby plánov obvykle figurujú okrem samotného zamestnanca a jeho manažéra aj ďalšie osoby. To si podrobnejšie vysvetlíme v kapitole 5 na príklade spoločnosti IBA CZ.

Aby boli vzdelávacie plány použiteľné, musia spĺňovať *SMART kritériá*[2], čiže musia byť:

- špecifické (specific) – konkrétne, jasne definované;

1. Individual Education Plan.
2. SMART Goals, `http://www.projectsmart.co.uk/smart-goals.html`.

- merateľné (measurable) – musí byť jasné, kedy bol cieľ splnený, prípadne aj ako ďaleko od splnenia cieľu sa aktuálne nachádzame;
- akceptované (agreed upon) – bolo dohodnuté, čo presne je cieľom;
- realistické (realistic) – splniteľné vzhľadom k času, znalostiam a zdrojom;
- termínované (time-based) – je určené, kedy majú byť ciele dosiahnuté a je overené, že je k dispozícii optimálne množstvo času.

4.3 Taktický a strategický plán

Okrem individuálnych vzdelávacích plánov je dôležitý aj koncept *strategického a taktického plánovania* z pohľadu organizácie ako celku. Strategické plánovanie funguje na makroúrovni – definuje dostupné zdroje (čas, peniaze a pod.) a určuje ciele, ktoré chceme dosiahnuť. Taktické plánovanie naopak operuje na mikroúrovni – popisuje, čo konkrétne je potrebné urobiť vzhľadom k zdrojom, aby boli ciele dosiahnuté [20]. Výsledkom je teda plán špecifický pre organizáciu, ktorý je potrebné splniť pomocou plánov jednotlivých zamestnancov.

4.4 Dostupné nástroje

Vzdelávacie plány predstavujú koncept, ktorý má svoje miesto vo fungovaní takmer každej organizácie, resp. všade tam, kde je možný kariérny rast. Napriek tomu však pri vykonaní prieskumu trhu zisťujeme, že neexistuje veľa nástrojov zameraných výhradne na ich správu.

Toto zistenie nie je také prekvapivé, keď si uvedomíme faktory súvisiace so správou plánov. Je potrebné akceptovať, že vzdelávacie plány sú vnútornou záležitosťou firmy. Priamo súvisia s rozpočtom, celkovým smerovaním firmy a ďalšími vnútornými procesmi, je teda pochopiteľné, že mnohé organizácie nezverejňujú žiadne detaily a často nie sú dohľadateľné ani tak základné informácie, ako napríklad názov systému, ktorý spoločnosť používa. Navyše procesy súvisiace s tvorbou vzdelávacích plánov môžu byť v závislosti na firme veľmi rôznorodé a môžu súvisieť s ďalšími konkrétnymi systémami používanými v organizácii, čo vytvorenie nejakého univerzálneho, robustného systému robí ešte ťažšou úlohou.

V prípade, že firma chce nejaký nástroj na spravovanie svojich vzdelávacích plánov, existujú teda prakticky dve možnosti. Prvou je vyriešiť to

v rámci spoločnosti využitím vlastných zdrojov. Tu sa núka varianta vytvoriť nový systém zjednodušený tým, že bude nastavený na mieru potrebám firmy (nie je to ojedinelé obzvlášť v prípade IT firiem, ktoré sú hlavným predmetov nášho záujmu). Výhodné môže byť integrovanie vzdelávacích plánov do existujúcich procesne orientovaných nástrojov používaných pôvodne na iné účely. Často stačí zaznamenať plány v systéme prepojených dokumentov dostupných v intranete, ktorý môže byť založený napríklad na systéme *wiki*[3] pri zavedení nejakých opatrení pre zachovanie súkromia zamestnanca, pričom výskyt takého systému na zdieľanie obsahu vo firmách nie je výnimkou. Príkladmi systémov, ktoré by mohli byť takto na správu plánov použité môžu byť nástroje na tímovú spoluprácu. Často používané sú produkty spoločnosti *Atlassian*[4] – systém na riadenie a sledovanie úloh a požiadavkov v projekte JIRA či systém na správu znalostí a zdieľanie dokumentov *Confluence*[5]. Výhodou sú relatívne nízke náklady (využitie už používaného systému), nevýhodou napr. to, že takéto riešenie spravidla nebude poskytovať také možnosti, aké by špecializované riešenie na správu plánov poskytovať mohlo. Táto možnosť je aplikovaná aj v spoločnosti IBA CZ, čo bude bližšie popísané v kapitole 5.

Druhou možnosťou je prispôsobiť existujúce riešenie vlastným potrebám. Je možné zakúpiť licenciu na nejaký komerčný nástroj a integrovať ho do procesov v organizácii. Je potrebné poznamenať, že správa vzdelávacích plánov je pomerne špecifický proces a na trhu sú predovšetkým pomerne komplexné tzv. *performance management* systémy, ktoré okrem správy vzdelávacích plánov poskytujú aj mnohé ďalšie služby orientované na *HR* (*human resources*) (napr. správu benefitov, ocenení a povýšení). Nevýhodou je, že môžu byť drahé (aj keď potrebuje firma len správu plánov, platí často za celý balík iných služieb, ktoré riešenie ponúka), nemusia presne vyhovovať potrebám spoločnosti a ich integrácia nemusí byť kvôli ich komplexnosti vždy jednoduchá. Konkrétnymi populárnymi riešeniami z tejto kategórie sú napr. *EmpXtrack Performance Management System*[6] – aplikácia prístupná cez webové rozhranie zameraná na hodnotenia zamestnancov, správu školení, kurzov, povýšení a pozícií – a *PROPHIX Performance Management Solution*[7] – aplikácia integrujúca procesy súvisiace s rozpočtom, plánovaním, vytváraním správ pre manažment, hodnotením zamestnancov a následnou pokročilou analýzou dát.

3. Wiki, `http://en.wikipedia.org/wiki/Wiki`.
4. Atlassian, `http://www.atlassian.com/`.
5. Confluence, `http://www.atlassian.com/software/confluence/`.
6. EmpXtrack, `http://www.empxtrack.com/performance-management-system/`.
7. PROPHIX, `http://www.prophix.com/solutions/performance-management-software/`.

Popri týchto systémoch stoja určite za zmienku aj systémy pre správu vzdelávania – *learning management* systémy (*LMS*). Jedná sa zvyčajne o webové aplikácie založené na princípoch elektronicky podporovanej výuky (*e-learning*), ktorých pôvodná myšlienka síce bola sprostredkovať akúsi virtuálnu školu, ale svoje uplatnenie dnes nachádzajú hlavne vo firmách. LMS poskytujú nástroje na plánovanie a hodnotenie procesu vzdelávania, pričom najväčší dôraz kladú interaktivitu – typicky poskytujú tiež zdieľanie študijných materiálov, video konferencie, diskusné fóra, chat medzi študentom a lektorom a pod. Medzi ďalšie poskytované funkcie patrí napríklad možnosť online testovania znalostí, sledovanie priebehu školení či vydávanie certifikátov. Výhodou týchto systémov je, že poskytujú spravidla kompletnú infraštruktúru súvisiacu so vzdelávaním. Existujú riešenia s užívateľskou podporou i riešenia, ktoré môžu byť nasadené v prostredí intranetu spoločnosti. Riešenia zamerané na organizácie navyše často poskytujú rozhrania na prepojenie s typickými systémami používanými v HR oddelení. Nevýhodou týchto systémov zase môže byť, že sú zamerané primárne na interaktívnu online výuku, čo v prípade správy vzdelávacích plánov zväčša nie je hlavná požadovaná funkcia. Omnoho dôležitejšou je integrácia systému s procesmi fungujúcimi v spoločnosti, čo môže byť v prípade týchto systémov problém. Všeobecne sa javia LMS ako pomerne dobrá voľba pre správu vzdelávacích plánov, všetko však samozrejme závisí na potrebách konkrétnej spoločnosti. Pozitívne je, že na trhu existuje aj viacero open-source implementácií, ktoré je možné upraviť a použiť. Konkrétnymi príkladmi takýchto systémov sú najznámejší *Moodle*[8], korporátne zameraný *Docebo*[9] či minimalistický *eFront*[10].

Existujú tiež spoločnosti, ktoré sa zaoberajú vyslovene hodnotením výkonu a plnenia cieľov (*performance management*), kariérnym rastom (*career development*) či riadením vzdelávania (*learning management*). Tieto firmy sú schopné navrhnúť systém vyhovujúci požiadavkom konkrétnej spoločnosti, dodať komplexné riešenie zahrňujúce aj správu vzdelávacích plánov a príp. ho prevádzkovať na vlastných serveroch. Zvyčajne sa jedná o pomerne drahú záležitosť, ale okrem konkrétnych nástrojov a ich podpory tieto firmy poskytujú aj ďalšie služby, ako napríklad rôzne kurzy v oblasti *soft skills*, školenia pre manažérov či vykonanie ďalších úkonov súvisiacich s HR. Príkladom môžu byť spoločnosti *HR Smart*[11], *PerformAWorks*[12] či *Workscape*[13],

8. Moodle, `http://moodle.org/`.
9. Docebo, `http://www.docebo.org/doceboCms/`.
10. eFront, `http://www.efrontlearning.net/`.
11. HR Smart, `http://www.hrsmart.com/`.
12. PerformAWorks, `http://www.aboutus.org/PerformAWorks.com`.
13. Workscape, `http://www.workscape.com/`.

ktoré majú klientov aj v Českej republike.

V súčastnosti existuje aj riešenie použiteľné v portáli – *Career Development Portlet* od spoločnosti *IBM*[14], ktorého úlohou je umožniť zamestnancom na jednom mieste spravovať artefakty spojené s ich kariérou [2]. Na základe profilu osoby dokáže portlet zobraziť informácie o pláne kariérneho rastu zamestnanca, profil získaných zručností a formu životopisu. K dispozícii sú tiež odkazy na aplikácie umožňujúce vyhľadávanie medzi dostupnými produktmi, pracovnými pozíciami, mentormi a pod. Zatiaľ čo zamestnanec si v portlete môže informácie o svojom pláne len zobrazovať, manažér môže vzdelávacie plány aj vytvárať a meniť (vrátane informácií o práci, kurzoch a zručnostiach). Je mu tiež umožnené vyhľadávať zamestnancov na základe viacerých údajov z korporátneho LDAP (napr. mena, emailovej adresy, oddelenia), zručností či pracovnej náplne. Tento portlet však nepredstavuje dostatočne flexibilné, a teda použiteľné riešenie. Dôvodom je jeho viazanosť na spoločnosť IBM a jej partnerov – je určený pre ich zamestnancov, nasadenie na portáli IBM Websphere Portal a je integrovaný so systémami spoločnosti IBM.

14. IBM, `http://www.ibm.com/`.

5 Analýza požiadavkov

Táto práca je vypracovaná v rámci Sdružení průmyslových partnerů Fakulty informatiky pre spoločnosť IBA CZ. Jej súčasťou je vytvorenie aplikácie bežiacej v prostredí portálu, ktorá umožňuje správu individuálnych vzdelávacích plánov zamestnancov. Projekt je nazvaný *IEP Manager* a aplikácia je prispôsobená potrebám spoločnosti IBA CZ, kde bude nasadená na portáli Liferay.

Kapitola začína analýzou situácie v spoločnosti IBA CZ v sekcii 5.1. V sekcii 5.2 definujeme špecifikáciu aplikácie a v sekcii 5.3 popisujeme prípady použitia.

5.1 Správa vzdelávacích plánov v IBA CZ

Správa plánov v spoločnosti IBA CZ má svoje špecifiká, ktoré museli byť pri návrhu aplikácie IEP Manager zohľadnené. Kľúčové aspekty sú popísané v tejto sekcii.

5.1.1 Štruktúra plánu

Každý zamestnanec spoločnosti má svoj individuálny vzdelávací plán. Ten má formu zoznamu položiek, ktoré by mal zamestnanec splniť, pričom každá položka musí byť splnená k nejakému dátumu. Položka je viazaná na konkrétnu aktivitu definovaného typu. Príkladom aktivity môže byť *Administrácia portálu Liferay* typu *Školenie* alebo *Testovanie aplikácií v prostredí portálu* typu *Samoštúdium*. V prípade, že typ aktivity podporuje inštancie viazané na konkrétny dátum, je pri plánovaní do položky doplnená aj konkrétna inštancia. Inštanciou *Administrácie portálu Liferay* teda môže byť napr. školenie konajúce sa *15. 9. 2011 o 15:00*. Položka má navyše definované podmienky splnenia, ktoré má každý zamestnanec špecifikované individuálne. Môže to byť napr. *získanie certifikátu* alebo *absolvovanie záverečnej skúšky s hodnotením A*.

5.1.2 Proces

V procese tvorby vzdelávacích plánov v spoločnosti IBA CZ aktuálne vystupuje niekoľko ľudí. Základom plánu je *hlavný motivačný pohovor*, ktorého sa typicky zúčastní *zamestnanec* a jeho manažér (*líniový manažér*). Na základe pohovoru požiada líniový manažér *manažéra pre vzdelávanie* (*training manager*) o zostavenie konkrétneho plánu do stanoveného termínu. Po splnení

úlohy informuje manažér pre vzdelávanie o vytvorenom pláne príslušného líniového manažéra, ktorý plán preberie so zamestnancom a doplní príslušné termíny. Ak je všetko v poriadku, plán je považovaný za hotový. V opačnom prípade je plán vrátený na doplnenie manažérovi pre vzdelávanie. Každý zamestnanec má aktuálne len jeden individuálny vzdelávací plán – po uplynutí obdobia daného na splnenie cieľov a ďalšom motivačnom pohovore nie je vytváraný nový plán, ale je aktualizovaný plán predošlý. V niektorých prípadoch môže manažéra pre vzdelávanie zastúpiť jeho *asistent*.

5.1.3 Súvisiace systémy

S procesom tvorby plánov v IBA CZ súvisí niekoľko systémov:

- JIRA – populárny systém riadenia a sledovania požiadavkov v projekte založený na princípe položiek reprezentujúcich úlohy, ktoré vyžadujú vyriešenie (*tickets*). V IBA CZ je okrem iného tento systém využívaný na riadenie toku súvisiaceho s procesom tvorby plánov – je vytvorená položka pre každý plán, ktorý prechádza niekoľkými stavmi podľa toho, kto sa ním má zaoberať. Po vytvorení portálovej aplikácie bude evidencia procesu tvorby plánov naďalej ponechaná v systéme JIRA.

- Confluence – systém správy znalostí a zdieľania dokumentov v prostredí organizácie založený na princípe *wiki*. V súčasnosti je okrem iného využívaný na evidenciu plánov, aktivít a ich inštancií. Keďže obsah dokumentov v ňom uložených je jednoducho upraviteľný kýmkoľvek s potrebnými prístupovými právami, môžu sa zamestnanci pohodlne vyjadriť napr. ku kurzu, ktorý absolvovali, čo môže byť cennou informáciou pre budúcich účastníkov. Táto vlastnosť je žiadaná, popis jednotlivých aktivít a ich inštancií bude preto naďalej ponechaný v tomto systéme a IEP Manager bude evidovať len odkazy na príslušné stránky.

- *Skill Manager* – projekt vyvíjaný paralelne s našou aplikáciou spravujúcou vzdelávacie plány. Jedná sa o portálovú aplikáciu, ktorá bude po dokončení umožňovať správu zručností zamestnancov. Získané zručnosti sú dôležité aj z pohľadu plánov, pretože predstavujú výsledok jednotlivých aktivít a sú rozhodujúce z pohľadu taktického plánu organizácie. Z tohto dôvodu bude tvorba taktických plánov prenechaná projektu Skill Manager, IEP Manager sa špecializuje na individuálne plány zamestnancov.

5.1.4 Problémy a ciele

Pri správe vzdelávacích plánov sa v IBA CZ používa niekoľko rôznych systémov, ku ktorým nie je jednotný prístup a manipulácia s plánmi tak nie je pohodlná. Manažér musí tieto systémy sledovať a užívateľ takisto nemá svoj plán jednoducho k dispozícii – musí sa do systému prihlásiť a príslušný dokument vyhľadať. IEP Manager je riešenie, ktoré si kladie za cieľ správu plánov zefektívniť a podstatné informácie pohodlne sprístupniť v portáli, ktorý zamestnanci používajú. Zamestnanec má okamžite k dispozícii svoj plán s prípadnými odkazmi do súvisiacich systémov a zamestnávateľovi je umožnené vykonať dôležité úkony súvisiace so správou plánov jednotlivých zamestnancov prehľadne priamo z prostredia portálu. Aplikácia navyše zjednodušuje sledovanie procesu tvorby plánov, umožňuje na úlohy pohotovo reagovať a na rozdiel od aktuálneho riešenia poskytuje možnosť generovať štatistiky a správy dôležité pre plánovanie.

5.2 Špecifikácia

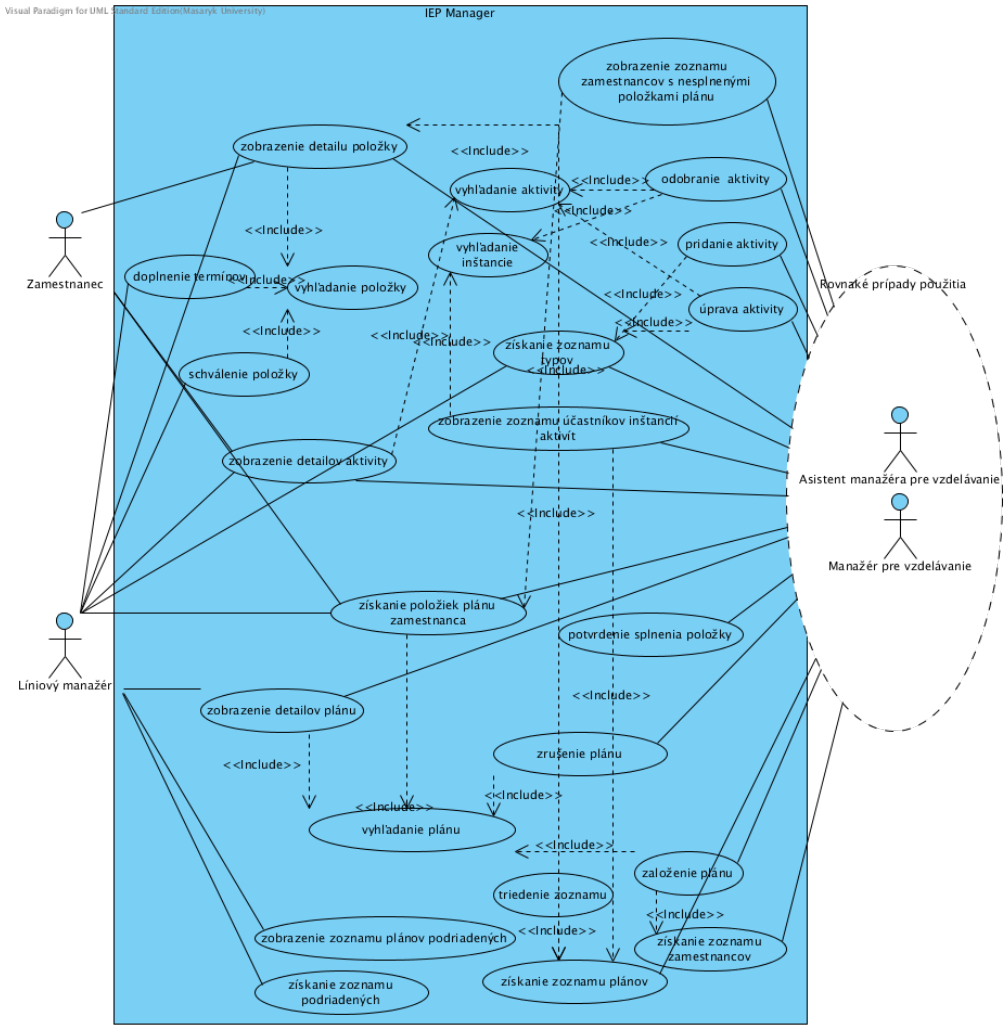
Aplikácia musí umožňovať:

- získanie zoznamu zamestnancov;
- získanie zoznamu podriadených líniového manažéra;
- zobrazenie zoznamu vzdelávacích plánov podriadených líniového manažéra;
- zobrazenie zoznamu vzdelávacích plánov, aktivít, inštancií aktivít, typov aktivít, stupňov voliteľností položiek plánu;
- založenie a zrušenie vzdelávacieho plánu zamestnanca;
- zobrazenie položiek plánu zamestnanca;
- pridanie, odobranie a upravenie položky plánu, aktivity, inštancie aktivity, typu aktivity, stupňa voliteľnosti položiek plánu;
- zobrazenie detailov o pláne, položke plánu, aktivite, inštancii aktivity, type a stupni voliteľnosti položiek plánu;
- pri plánoch zobrazenie odkazu na príslušnú úlohu v systéme JIRA;
- pri aktivitách a inštanciách zobrazenie odkazu na popis v systéme Confluence;

- schválenie položky plánu;
- potvrdenie splnenia položky plánu;
- doplnenie termínov k položkám plánu;
- vyhľadanie zamestnanca, plánu, aktivity, inštancie, typu a stupňa voliteľnosti položiek plánu;
- triedenie zamestnancov, aktivít, inštancií, typov a stupňov voliteľnosti položiek plánu pri zobrazení;
- odlíšenie položiek plánov vyžadujúcich schválenie;
- zvýraznenie zatiaľ nesplnených položiek plánu po dátume splnenia;
- zvýraznenie zatiaľ nesplnených položiek plánu tesne pred dátumom splnenia;
- zobrazenie zoznamu ľudí, ktorí majú pridelené jednotlivé aktivity;
- zobrazenie účastníkov inštancií aktivít;
- generovanie reportu o položkách, ktoré splnené nie sú a splnené v aktuálnom čase už mali byť.

5.3 Prípady použitia

Prípady použitia sú zobrazené v diagrame na obrázku 5.1. Upozorňujeme, že diagram je skôr ilustračný a nie je kompletný – kvôli veľkosti podrobného diagramu boli niektoré prípady použitia zjednodušené či vynechané. Je to prípad operácií ako zobrazenie zoznamu a jeho triedenie, pridanie, odobranie, upravenie a zobrazenie detailu, ktoré prebiehajú na všetkých entitách analogicky, a preto ich v diagrame neznázorňujeme na každom type entity. Rozdiel medzi operáciami môže byť len v overení závislostí, ktoré ale plynú z dátového modelu (napr. pri odstraňovaní typu je potrebné overiť, či sa v databáze nevyskytujú aktivity daného typu). Podobne vyhľadávanie môže prebiehať na základe rôznych kritérií, čo v diagrame v rámci zachovania jednoduchosti nerozlišujeme.



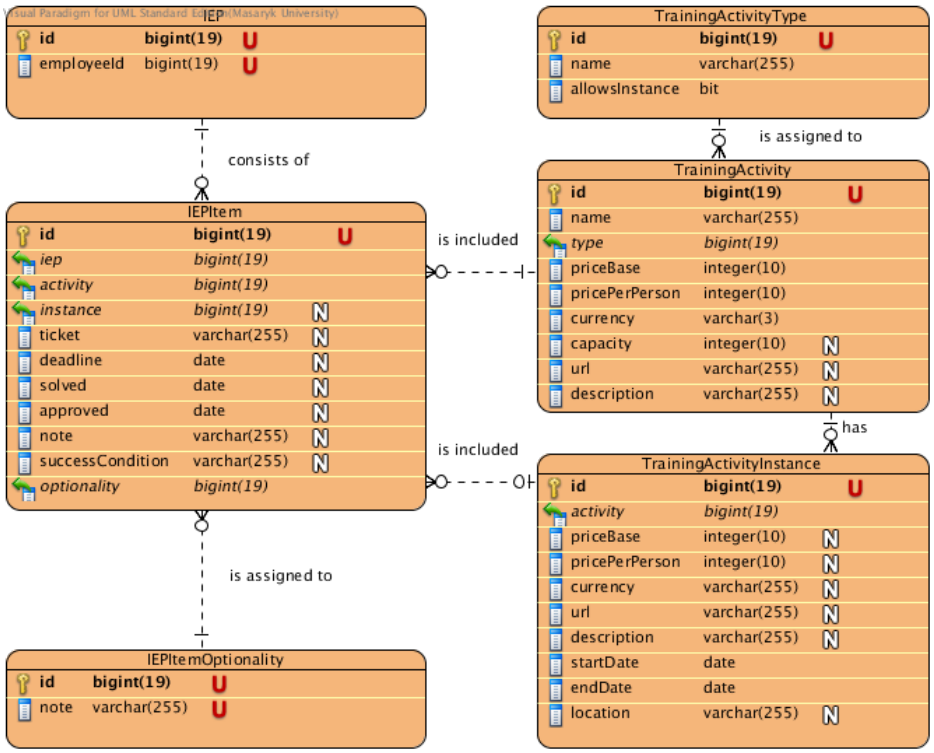
Obrázok 5.1: Stručný diagram prípadov použitia pre aplikáciu IEP Manager.

6 Návrh aplikácie IEP Manager

Táto kapitola popisuje architektúru a dátový model IEP Manager. Aplikácia bola rozdelená do dvoch modulov – *IEP Manager Backend* a *IEP Manager Portlets*.

6.1 Dátový model

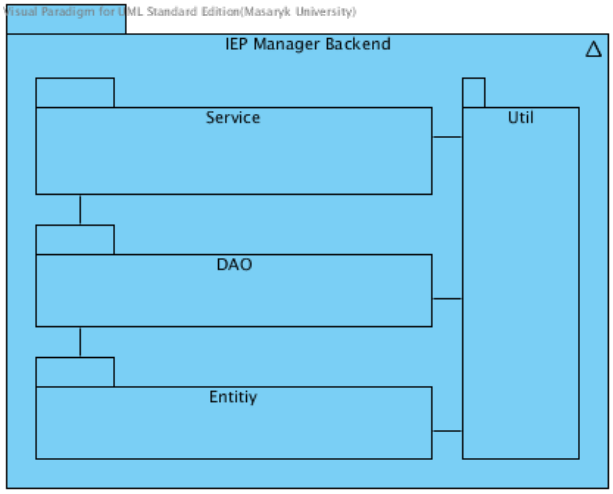
Na obrázku 6.1 je znázornený dátový model aplikácie IEP Manager. Vzdelávací plán je reprezentovaný tabuľkou `IEP`, položka plánu tabuľkou `IEPItem`, tréningová aktivita tabuľkou `TrainingActivity`, inštancia aktivity tabuľkou `TrainingActivityInstance`, voliteľnosť položky vzdelávacieho plánu tabuľkou `IEPItemOptionality` a typ aktivity tabuľkou `TrainingActivityType`. Vzťahy medzi jednotlivými entitami odpovedajú popisu plánu v kapitole 5, sekcii 5.1.



Obrázok 6.1: Entitno-relačný diagram.

6.2 Modul IEP Manager Backend

IEP Manager Backend obsahuje hlavnú aplikačnú logiku a stará sa o prístup k databáze. Skladá sa zo štyroch základných častí nazvaných *Entity*, *DAO*, *Service* a *Util*. Jeho štruktúra je znázornená na obrázku 6.2.



Obrázok 6.2: Štruktúra modulu IEP Manager Backend.

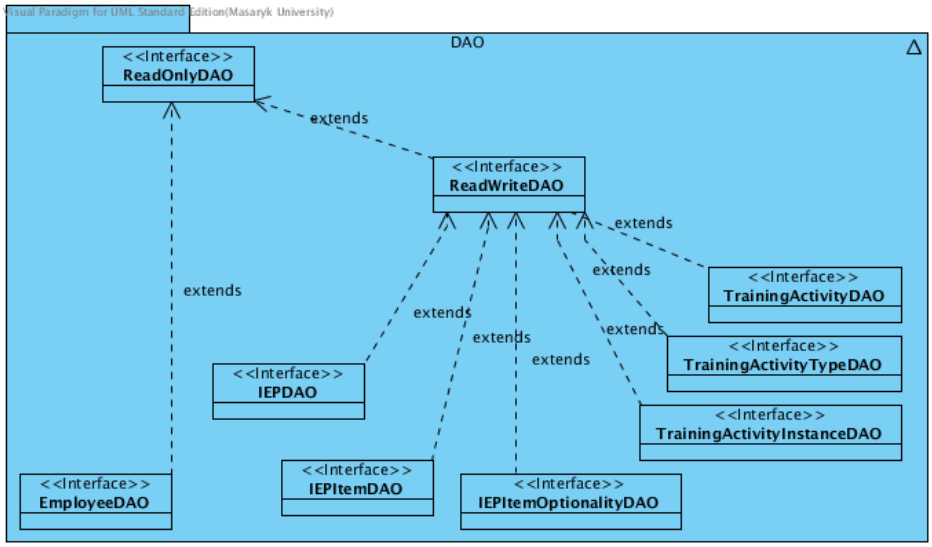
6.2.1 Entity

Balík `eu.ibacz.k3.gadgets.iepmanager.backend.entity` obsahuje základné objekty vystupujúce vo vzdelávacích plánoch (triedy rozširujúce abstraktnú triedu `IEPEntity`). Jedná sa o entity znázornené na obrázku 6.1, ktoré sú ukladané do databázy. Okrem toho tu je entita `Employee` reprezentujúca zamestnanca spoločnosti, ktorej atribúty sú získané priamo z portálu. Všetky triedy tohto balíku spadajú do kategórie JavaBean – obsahujú východzí bezparametrický konštruktor, sú serializovateľné a ich atribúty sú sprístupnené pomocou základných metód (*get*, *set*, *is*).

6.2.2 DAO

Balík `eu.ibacz.k3.gadgets.iepmanager.backend.dao` predstavuje vrstvu prístupu k dátam a obsahuje rozhrania a ich východzie implementácie pracujúce nad triedami z časti entity. Jednotlivé implementácie rozhraní pracujú aktuálne s dvoma typmi úložísk dát – relačnou databázou a portálom.

Hierarchiu usporiadania rozhraní znázorňujeme v diagrame na obrázku 6.3. Pomenované sú podľa toho, ku ktorej entite nám sprostredkúvajú prístup. Každé rozhranie má implementujúcu triedu a tie sú usporiadané analogicky.



Obrázok 6.3: Štruktúra vrstvy DAO.

6.2.3 Service

Balík `eu.ibacz.k3.gadgets.iepmanager.backend.service` predstavuje servisnú vrstvu – najvyššiu vrstvu modulu IEP Manager Backend, s ktorou pracuje modul IEP Manager Portlets. Jedná a o rozhrania a ich východzie implementácie poskytujúce transakčné operácie nad dátami. Sú priamou nadstavbou vrstvy DAO.

Vrstvu tvorí šesť rozhraní a šesť implementácií týchto rozhraní, z ktorých každé sa stará o operácie nad jednou entitou (s výnimkou rozhrania `IEPService`, ktoré na tejto úrovni združuje prácu so zamestnancami a vzdelávacími plánmi, keďže tieto entity spolu výrazne súvisia).

6.2.4 Util

Balík `eu.ibacz.k3.gadgets.iepmanager.backend.util`obsahuje pomocné triedy potrebné pre správne fungovanie modulu – konštanty, výnimky a pod.

6.3 Modul IEP Manager Portlets

Na základe analýzy požiadavkov bola navrhnutá dekompozícia aplikácie IEP Manager na tri portlety – *Administrator*, *Viewer* a *Reporter*. Každý portlet je umiestnený v samostatnom balíku, z ktorých každý okrem troch implementácií radičov (jeden pre každý portletový mód) obsahuje triedu s konštantami. Okrem toho je tu ešte balík s pomocnými triedami slúžiacimi na konverziu, validáciu, mapovanie entít a pod.

Prístupové práva k portletom sú nastavované v portáli manuálne na základe pozície zamestnanca vo firme.

6.3.1 Portlet Administrator

Administrator je portlet umožňujúci správu jednotlivých entít vystupujúcich vo vzdelávacích plánov, ku ktorému má vo firme prístup manažér pre vzdelávanie, jeho asistent a líniovíí manažéri. Umožňuje pridávanie, odoberanie a úpravu vzdelávacích plánov, tréningových aktivít, inštancií, typov aktivít a voliteľnosti jednotlivých položiek plánu, pričom líniovíí manažéri si môžu jednotlivé entity len prezerať.

6.3.2 Portlet Viewer

Viewer je portlet, ktorého hlavnou úlohou je zobraziť vzdelávací plán. Je prístupný každému zamestnancovi firmy pre zobrazenie položiek v jeho vlastnom pláne. Líniový manažér si v ňom môže zobraziť vzdelávací plán niektorého zo svojich podriadených a manažér pre vzdelávanie (resp. jeho asistent) plán akéhokoľvek zamestnanca firmy. Portlet môže teda fungovať v 3 režimoch:

- zobrazenie plánu aktuálne prihláseného užívateľa portálu (východzí režim)
- zobrazenie plánu pevne zvoleného zamestnanca
- zobrazenie plánu zamestnanca dynamicky voleného v portlete Administrator

Posledná možnosť je pritom založená na komunikácii portletu Administrator a Viewer pomocou verejných parametrov a následným uložením potrebných údajov do relácie, aby ostal plán trvalo zobrazený.

6.3.3 Portlet Reporter

Reporter je portlet umožňujúci generovať reporty týkajúce sa vzdelávacích plánov. V súčasnosti portlet umožňuje výber z dvoch štatistík a ich následné zobrazenie priamo v portlete, pričom sa predpokladá, že táto funkcionalita bude v budúcnosti rozšírená (viď kapitolu 7, sekciu 7.3).

Prvou štatistikou je zoznam ľudí, ktorých vzdelávací plán obsahuje jednotlivé tréningové aktivity a zoznam účastníkov jednotlivých inštancií. Druhou štatistikou je plnenie vzdelávacích plánov. Dôležitý je predovšetkým zoznam zamestnancov, ktorí niektoré položky zo svojich plánov nesplnili v stanovenom čase.

Portlet je dostupný manažérovi pre vzdelávanie a jeho asistentovi.

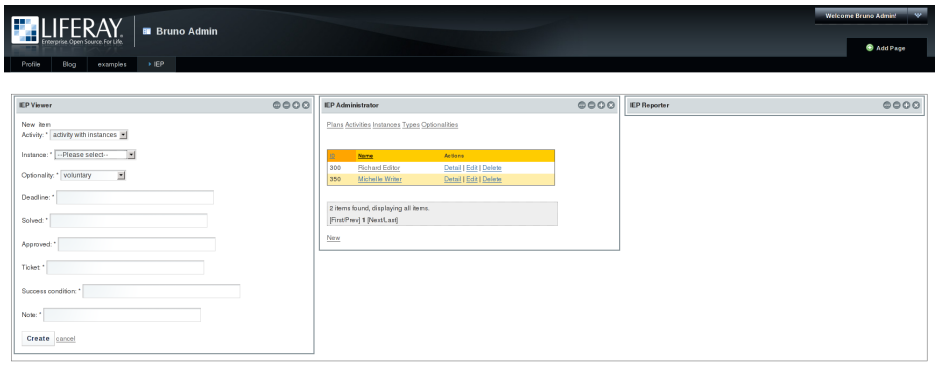
7 Implementácia

V tejto kapitole popisujeme implementáciu aplikácie IEP Manager. Dôraz je kladený predovšetkým na použité technológie, pričom rozhodnutie pre každú z nich zdôvodňujeme a popisujeme, ako bola využitá. V závere kapitoly analyzujeme možné rozšírenia aplikácie v budúcnosti.

7.1 Rozhranie

Funkcionalita jednotlivých portletov je popísaná v predošlej sekcii a vystihuje základné znaky grafického rozhrania. K nemu snáď len dodáme, že rozhranie aplikácie je kompletne lokalizované do dvoch jazykov – češtiny a angličtiny.

Náhľad rozhrania v prostredí portálu Liferay 5 je zobrazený na obrázku 7.1.



Obrázok 7.1: Rozhranie portálovej aplikácie IEP Manager.

7.2 Použité technológie

Pri implementácii portálovej aplikácie bolo využitých niekoľko zaujímavých technológií a rámcov, z nich najdôležitejšie si popíšeme v tejto sekcii.

7.2.1 Spring

Spring[1] je aplikačná platforma zameraná na budovanie Java aplikácií distribuovaná pod licenciou *Apache License 2.0*[2]. Jedná sa o rámec s veľmi

1. SpringSource.org, `http://www.springsource.org/`.
2. Apache License 2.0, `http://www.apache.org/licenses/LICENSE-2.0.html`.

modulárnou architektúrou, ktorá umožňuje vývojárovi použiť len tie časti, ktoré naozaj potrebuje. Jeho hlavným cieľom pritom je poskytovať množstvo funkcií tak, aby aplikácie neboli viazané na konkrétne Java EE služby či rozhrania samotného rámca – triedy nemusia používať jeho rozhrania a operuje s obyčajnými Java objektami (POJO[3]) [12].

Hlavným princípom rámca Spring je vkladanie závislostí a inverzia riadenia (*Dependency Injection and Inversion of Control*). Jedná sa o návrhový vzor zameraný na tvorbu systémov pozostávajúcich z voľne viazaných komponentov. Komponent, ktorý Spring spravuje sa označuje pojmom *bean* a deklarujeme ho v konfiguračnom súbore vo formáte XML[4] alebo anotáciou. Oproti tým pevne viazaným komponentom je základný rozdiel v tom, že voľne viazané komponenty nezískavajú závislosti (iné objekty) tak, že si ich vytvoria sami, ale tak, že ich dostanú pri ich vytváraní alebo počas behu priamo od rámca Spring, resp. jeho časti označovanej ako *IoC kontajner*. Podľa toho, ako kontajner do komponenty objekty dosadzuje, rozlišujeme vkladanie závislostí *set* metódou (*Setter Injection*), konštruktorom (*Constructor Injection*) a rozhraním (*Interface Injection*) [8].

Spring obsahuje okolo 20 modulov, z ktorých IEP Manager používa nasledujúce:

- *Core Container*.

  Jedná prakticky o skupinu štyroch modulov (Core, Beans, Context a Expression Language), ktoré sa starajú o základné funkcie celého rámca [12] – vkladanie závislostí, transparentné vytváranie kontextu, prístup k objektom a zdrojom, propagáciu udalostí, dotazovanie apod.

- *ORM*, *Transaction*, *JDBC* a *AOP*.

  ORM poskytuje integračné vrstvy pre populárne riešenia objektovo-relačného mapovania, ako je napr. *Hibernate*[5]. Vďaka tomuto modulu je prakticky možné používať rámce pre objektovo-relačné mapovanie v kombinácii so všetkým, čo ponúka Spring. Transaction zase poskytuje deklaratívnu správu transakcií pre triedy implementujúce špecifické rozhrania a všetky objekty typu *POJO* [12]. K deklaratívnemu prístupu k transakciám je nutný tiež modul AOP prinášajúci implementáciu aspektovo orientovaného programového modelu[6]. ORM a

3. Plain Old Java Objects, `http://en.wikipedia.org/wiki/Plain_Old_Java_Object`.
4. eXtensible Markup Language, `http://www.w3schools.com/xml/`.
5. Hibernate, `http://www.hibernate.org/`.
6. TODO: spring cite

Transaction naviac využívajú aj modul JDBC poskytujúci jednoduchú abstrakciu nad bežným prístupom k databáze.

Moduly ORM a Transaction využíva IEP Manager vo svojom module IEP Manager Backend. ORM je využitý práve pri integrácii Hibernate, zatiaľ čo Transaction sa používa na servisnej vrstve pri anotovaní metód pre vykonanie v transakcii (anotácia `Transaction`).

- *Web*, *Web-Servlet*, *Web-Portlet*.

  Modul Web poskytuje základné integračné funkcie orientované na web. Web-Servlet obsahuje implementáciu *MVC*[7] rámca pre webové aplikácie – *Spring Web MVC* – a Web-Portlet poskytuje analogickú MVC implementáciu použiteľnú v prostredí portálu – *Spring Portlet MVC*.

  Spring Portlet MVC predstavuje podporu vývoja portletov podľa základných portletových štandardov (viď kapitolu 3). Potrebnú konfiguráciu je možné dodať aplikácii vo forme XML, ale nové verzie rámca už umožňujú väčšinu záležitostí nastaviť cez anotácie.

  Základom portálovej aplikácie pri využití Spring Portlet MVC je trieda `DispatcherPortlet` starajúca sa o distribúciu požiadavkov medzi radiče (triedy implementujúce rozhranie `Controller`, neimplementujú teda rozhranie `Portlet`). `DispatcherPortlet` využíva na vykresľovanie špeciálny servlet `ViewRendererServlet`, ktorý poskytuje potrebnú abstrakciu a konverzie umožňujúce využitie prakticky ľubovoľnej technológie na zobrazovanie (napr. JSP[8] a Tiles[9]) [12].

  Radiče sú POJO komponenty zväčša určené anotáciou `@Controller`. To, ktorý radič má obslúžiť požiadavok od `DispatcherPortlet`, je určené príslušným mapovaním na objekty implementujúce rozhranie `HandlerMapping`. Radiče následne požiadavok spracujú a vrátením objektu `ModelAndView` späť do `DispatcherPortlet` vo fáze vykreslenia rozhodnú, čo má byť zobrazené. Spring poskytuje niekoľko pripravených radičov, ktoré je možné využiť [12].

  Spring sa na rozdiel od niektorých rámcov nesnaží portletové fázy spracovania požiadavku skryť do jedného celku a pripodobniť tak vývoj portletov vývoju bežných webových aplikácii, ale umožňuje vý-

7. Model-View-Controller – návrhový vzor oddeľujúci dátový model (*model*), užívateľské rozhranie (*view*) a riadiacu logiku aplikácie (*controller*). Viac informácií na `http://java.sun.com/blueprints/patterns/MVC-detailed.html`.
8. JavaServer Pages, `http://www.oracle.com/technetwork/java/javaee/jsp/`.
9. Apache Tiles, `http://tiles.apache.org/`.

vojárovi využiť výhody tejto dekompozície. To, čo by sa v klasickom portlete podľa JSR-286 vykonalo v jednotlivých metódach triedy `GenericPortlet` pri obsluhovaní požiadavku (viď kapitolu 3, sekciu 3.3), je navyše jednoduché rozdeliť do samostatných metód a na konkrétne fázy spracovania ich naviazať pomocou príslušných anotácií (základnými sú `@ActionMapping`, `@EventMapping`, `@RenderMapping` a `@ResourceMapping`). O dáta prichádzajúce do radiča sa stará objekt implementujúci rozhranie `Model`, ktorý môžme sprístupniť v metóde označením jej parametra anotáciou `@Model`. Jednotlivé údaje z požiadavku získame anotáciami `@ModelAttribute` a `@RequestParam`, ktorými ich môžme automaticky naviazať na objekt. Podobne môžme cez `@SessionAttributes` na úrovni triedy definovať údaje viazané na reláciu [12].

Spring Portlet MVC poskytuje aj mnoho ďalších funkcií, z ktorých viaceré využíva IEP Manager vo svojom module IEP Manager Portlets. Príkladom môže byť podpora práce s formulármi (spracovanie validácie, automatické naviazanie položiek na atribúty objektu následne dostupného cez `@ModelAttribute`) a pod.

Základom aplikácie postavenej nad Spring Portlet MVC je niekoľko súborov typicky umiestnených v adresári *WEB-INF*. Základný objekt `DispatcherPortlet` je definovaný v súbore *portlet.xml*. Pri jeho inicializácii dochádza k vytvoreniu komponentov nájdených v súboroch *[meno portletu]-portlet.xml*, ktoré definujú kontext pre radiče. Súbor *web.xml* musí definovať minimálne `ViewRendererServlet` a `ContextLoaderListener`, ktorý načíta súbor *applicationContext.xml*. Tu definujeme technológiu pre zobrazovanie, cestu k ďalším konfiguračným súborom a prípadne ďalšie komponenty, napríklad súbor s lokalizovanými správami.

Hlavným dôvodom použitia rámca Spring je množstvo ponúkaných funkcií, ich jednoduché použitie, automatické nájdenie komponentov a vkladanie závislostí. Výhodou je tiež celková univerzálnosť rámca – možnosť uplatnenia na prakticky všetkých vrstvách aplikácie (od dátovej vrstvy cez vrstvu aplikačnej logiky až po prezentačnú vrstvu). Zároveň tento rámec patrí medzi podporované technológie v IBA CZ.

7.2.2 Hibernate a JPA

Tradičný prístup k databáze z Java aplikácií realizovaný pomocou JDBC[10] je síce z pohľadu výkonu efektívny a poskytuje vývojárovi prakticky plnú kontrolu nad tým, čo sa s databázou deje, obvykle je ale pomerne nepraktický. Vývojár musí mať slušné znalosti SQL[11], sám spravovať spojenie s databázou a vymyslieť spôsob, ako do databázy ukladať objekty, s ktorými pracuje.

Hibernate je nástroj distribuovaný pod licenciou *LGPL 2.1*[12], ktorý poskytuje objektovo-relačné mapovanie pre Java aplikácie. Jeho hlavnou úlohou je, tak ako v prípade všetkých produktov starajúcich sa o objektovo-relačné mapovanie, riešiť problém nekompatibility objektovo orientovanej a relačnej paradigmy. Inými slovami, Hibernate rieši mapovanie dát z objektov do dátového modelu relačnej databázy, čím umožňuje vývojárovi pracovať s akousi virtuálnou databázou objektov a nenúti ho zaoberať sa detailami SQL (aj keď istá znalosť je výhodou) [15].

Hibernate ponúka jednotné rozhranie k väčšine dnes používaných databáz, čím uľahčuje prípadnú migráciu na iný druh databázy. Poskytuje tiež okrem samotného mapovania množstvo pokročilých služieb, ako napr. dotazovací jazyk HQL, validáciu či dávkové spracovanie.

Java Persistence API (*JPA*) je špecifikácia rozhrania pre zaistenie perzistencie a objektovo-relačného mapovania. Prvá špecifikácia tohto rozhrania bola súčasťou *JSR-220*[13] a popisovala základné princípy a terminológiu – entity, dotazovací jazyk *JPQL* (Java Persistence Query Language), metadáta [7]... Dnes bežne používaná verzia JPA 2 bola štandardizovaná v roku 2009 v podobe JSR-317[14] a zjednotila mnohé vylepšenia implementované výrobcami najrozšírenejších rámcov pre objektovo-relačné mapovanie – napr. podporu validácie, metamodel a kritériá [11].

Hibernate je dnes technológiou plne kompatibilnou so štandardom JPA, okrem neho však poskytuje aj vlastné rozhranie obsahujúce funkcie, ktoré síce (zatiaľ) nie sú súčasťou štandardu, v mnohých situáciách je však pohodlné a efektívne ich použiť.

Keďže IEP Manager sa stará o správu viacerých entít a vykonáva viacero operácií nad databázou, rozhodli sme sa pre využitie ORM rámca, a to práve Hibernate. Pristupujeme k nemu ale výhradne pomocou rozhrania JPA, pretože sa jedná o univerzálny štandard a aplikácia tak nie je viazaná

10. Java Database Connectivity, `http://www.oracle.com/technetwork/java/overview-141217.html`.
11. SQL Fundamentals, `http://databases.about.com/od/sql/a/sqlfundamentals.htm`.
12. LGPL 2.1, `http://www.gnu.org/licenses/old-licenses/lgpl-2.1.html`
13. Enterprise JavaBeans 3.0, `http://www.jcp.org/en/jsr/detail?id=220`.
14. Java Persistence 2.0, `http://www.jcp.org/en/jsr/detail?id=317`.

na výrobcu konkrétneho rámca.

IEP Manager využíva JPA vo svojom module IEP Manager Backend. Základom je balík `eu.ibacz.k3.gadgets.iepmanager.backend.entity` a v ňom umiestnené *entity* – triedy označené anotáciou `@Entity` (resp. tak nakonfigurované v XML), ktoré reprezentujú objekty perzistované do databázy. Takáto entita musí mať bezparametrický konštruktor deklarovaný ako *public* (resp. *protected*) a žiadne jej metódy, atribúty a ani ona samotná nesmú byť označené ako *final*. Jedná sa vždy o triedu najvyššej úrovne, nie rozhranie či výčtový typ (*Enum*). Entita v zmysle JPA by pritom nemala obsahovať aplikačnú logiku, len atribúty a metódy k nim pristupujúce (*get*, *set*, *is*) [7], čoho sa pridržiavame aj v tejto aplikácii. Okrem samotného konceptu JPA entít pritom využívame na tejto úrovni aj ďalšie funkcie poskytované JPA – prepojenie objektov (`@OneToMany`, `@ManyToOne`), generovanie primárneho kľúča (`@Id`) databázou automaticky (`@GeneratedValue`) na základe sekvencií (`@SequenceGenerator`), *lazy loading* (získavanie entít naviazaných na danú entitu nie spolu s ňou, až pri priamom prístupe k nim samotným – `FetchType.LAZY`) a pod.

V balíku `eu.ibacz.k3.gadgets.iepmanager.backend.dao` nájdeme všetky *DAO* objekty (*Data Access Object*) vykonávajúce základné operácie nad entitami s využitím JPA, konkrétne pomocou metód triedy `EntityManager` a dotazovacieho jazyka JPQL. JPQL je založený na podobnom princípe ako samotné SQL, umožňuje však napr. vytvárať typované dotazy (`TypedQuery`) vracajúce objekty.

7.2.3 Liferay

Liferay je populárny podnikový portál založený na technológii Java EE, ktorý umožňuje jednoduchú integráciu informácií, aplikácií a procesov. Je distribuovaný v dvoch edíciách – Liferay Portal Community Edition je verzia dostupná zadarmo pod licenciou *MIT*[15], zatiaľ čo Liferay Portal Enterprise Edition je platená, stabilná verzia s podporou [16].

Portál je už v základnej verzii vybavený niekoľkými desiatkami portletov a ponúka množstvo služieb: systém pre správu obsahu, integráciu s kancelárskym balíkom Microsoft Office[16], vyhľadávanie v obsahu, wiki, chat, email, blogy, RSS, zdieľaný kalendár a pod.

Liferay poskytuje mnohé služby formou aplikačného programového rozhrania. IEP Manager toto rozhranie využíva na získanie informácií z profilu užívateľa v portáli (mena, emailu, fotografie a pod.) pri zobrazení detai-

15. Licencia MIT, `http://www.opensource.org/licenses/mit-license.php`.
16. MS Office, `http://office.microsoft.com`.

lov vzdelávacieho plánu. Podobne zoznam zamestnancov firmy je získaný zo zoznamu užívateľov portálu (zamestnanci spoločnosti majú vytvorený účet v portáli).

Aplikácia obsahuje v adresári *WEB-INF* dva deskriptory špecifické pre portál Liferay. Súbor *liferay-portlet.xml* obsahuje detaily týkajúce sa Liferay aplikácie – mapovanie rolí užívateľov bezpečných z pohľadu aplikácie, kaskádové štýly a skripty využívané v jednotlivých portletoch (budú zahrnuté do hlavičky stránky, ktorú portlet negeneruje sám) a príznak určujúci možnosť umiestniť v portáli na stránky viac inštancií jedného portletu. Súbor *liferay-display.xml* zase určuje názvy portletov a ich kategórie, pod ktorými sa zobrazia pri pridávaní na stránku.

Hlavným dôvodom cielenia aplikácie pre tento portál je fakt, že Liferay je portálom používaným v IBA CZ.

7.2.4 JSP

JSP (*JavaServer Pages*) predstavujú Java EE technológiu určenú na tvorbu aplikácií dynamicky generujúcich webový obsah (*HTML*[17], XML) a sú súčasťou celej rady špecifikácií – *JavaServer Pages Specification 1.0* (1999) a *1.1*[18] (1999), *1.2*[19] (JSR-53, 2001), *2.0*[20] (JSR-152, 2003) a *2.1*[21] (JSR-245, 2006). Okrem toho existujú ďalšie štandardy pre technológie súvisiace s JSP, napr. *JSR-267*[22].

JSP je možné si predstaviť ako isté *inverzné servlety* – JSP stránka je textový súbor (*.jsp*), ktorý obsahuje HTML preložené miestami kúskami Java kódu (*skriptletmi*). Skriptlety sú podporované, ale kvôli zníženiu čitateľnosti kódu je výrazne nedoporučované umiestňovať do nich aplikačnú logiku.

So samotným štandardom JSP súvisia ďalšie dve technológie – *JSTL*[23] a *EL* (dnes *Unified Expression Language*). JSTL je štandardná knižnica značiek, ktoré môžme použiť pri písaní JSP stránok. Reálne pozostáva z niekoľkých oddelených knižníc, ktoré poskytujú množstvo užitočných funkcií a umožňujú nám v JSP napríklad pracovať s premennými, iterovať pomocou cyklov, používať vetvenie, manipulovať s URL, transformovať XML, apliko-

17. HyperText Markup Language, `http://www.w3schools.com/html/`.
18. JavaServer Pages Specification 1.0, 1.1, `http://java.sun.com/products/jsp/archive.html`.
19. JavaServer Pages Specification 1.2, `http://www.jcp.org/en/jsr/detail?id=53`.
20. JavaServer Pages Specification 2.0, `http://www.jcp.org/en/jsr/detail?id=152`.
21. JavaServer Pages Specification 2.1, `http://www.jcp.org/en/jsr/detail?id=245`.
22. JSP Tag Library for Web Services, `http://www.jcp.org/en/jsr/detail?id=267`.
23. A Standard Tag Library for JavaServer Pages (JSTL) 1.1, `http://www.jcp.org/en/jsr/detail?id=52`.

vať rôzne funkcie na reťazce či zobrazovať lokalizované údaje [5]. EL nám zase zjednodušuje prístup k dátam (zo stránky, aplikácie, požiadavku i relácie) a umožňuje nám použiť ich v jednoduchých výrazoch.

Výhodou JSP je určite tiež podpora šablón umožňujúcich oddelenie statického a dynamického obsahu a jednoduchého pridávania dynamicky získaných dát do nich, platformová nezávislosť, oddelenie rolí vývojára a autora, kvalitné nástroje pre vývoj i podpora skriptovania, akcií a výrazov) [6].

IEP Manager (modul IEP Manager Portlets) využíva JSP na prezentáciu dát z portletov spolu s prostriedkami JSTL a EL. Skriptlety sú použité v duchu zachovania čistoty kódu výhradne na vkladanie konštánt. Jednotlivé stránky sú umiestnené v adresári *WEB-INF/jsp*.

Technológia JSP bola vybraná pre použitie v aplikácii hlavne preto, lebo je štandardne používaná, podporovaná zo strany už spomínaných nástrojov, jednoducho použiteľná na zobrazenie dynamicky generovaného obsahu a zároveň poskytuje dostatočnú funkcionalitu.

7.2.5 jQuery

jQuery[24] je voľne dostupná knižnica v jazyku JavaScript distribuovaná pod licenciami MIT a *GPL*[25]. Poskytuje jednoduchý mechanizmus pre prechádzanie HTML dokumentu, obsluhu udalostí, animácie a interakciu prostredníctvom technológie AJAX (Asynchronous JavaScript and XML) [13]. Tá v zásade pracuje na princípe umiestnenia nového článku (*AJAX engine*) medzi užívateľa a server a nahradzuje tak klasickú štart-stop-štart-stop povahu interakcie užívateľa na webe novým modelom asynchrónnej komunikácie, ktorá nám umožňuje predovšetkým dynamicky meniť obsah na stránke bez obnovenia celej stránky [9]. To je veľmi užitočné obzvlášť v prostredí portálu.

Medzi hlavné výhody tejto knižnice patrí veľká rozšírenosť, vysoký výkon, podpora všetkých najpoužívanejších webových prehliadačov a veľké množstvo dostupných rozšírení a zásuvných modulov.

Hlavným dôvodom použitia jQuery bola okrem spomínaných výhod aj podpora knižnice zo strany portálu Liferay, ktorý ju priamo obsahuje a je teda možné ju použiť bez toho, aby sme museli konkrétnu verziu zahŕňať vo vlastnej aplikácii.

IEP Manager (modul IEP Manager Portlets) využíva na prezentačnej

24. jQuery, `http://jquery.com/`
25. GNU General Public Licence, `http://www.gnu.org/licenses/old-licenses/gpl-2.0.html`.

vrstve zásuvné moduly *Autocomplete*[26] a *Datepicker*[27]. Autocomplete slúži na našepkávanie pri vypĺňaní textového políčka a použitý je napr. pri vypĺňaní zamestnanca počas tvorby vzdelávacieho plánu v portlete Administrator, kedy našepkáva mená užívateľov portálu. Datepicker zase slúži na zadávanie dátumu prostredníctvom kalendára, napr. pri vytváraní inštancie aktivity, opäť v portlete Administrator.

7.2.6 CSS

Kaskádové štýly (*CSS*[28]) je štandard prijatý W3C[29], pričom najnovšia verzia je *CSS 3*. Jedná sa o dnes bežne používaný jazyk na popis vzhľadu a formátovania dokumentu, ktorý je podporovaný väčšinou webových prehliadačov [4]. Jeho hlavným cieľom je oddelenie obsahu od prezentácie dokumentu, čo je prístup prinášajúci množstvo výhod – flexibilitu, zdieľanie vzhľadu medzi rôznymi dokumentmi, zvýšenú čitateľnosť dokumentu a pod.

Použitie kaskádových štýlov v portletoch je pomerne špecifická problematika a v zásade ide pri tvorbe štýlov o nájdenie vhodného kompromisu medzi jednotným vzhľadom, prenosnosťou a použiteľnosťou.

Na jednej strane máme samotný portál, pre ktorý vyvíjame aplikáciu. Ten prichádza s vlastnými kaskádovými štýlmi a vlastným vzhľadom, ktorý je navyše umožnené užívateľovi meniť (typicky už čerstvo nainštalovaná inštancia portálu ponúka výber z niekoľkých schém vzhľadu, šablón, motívov či tém, pričom je možné nové vytvoriť či stiahnuť z internetu). Keďže portál by mal poskytovať jednotné a užívateľsky prívetivé rozhranie k informáciám, cieľom dizajnéra portletu je vytvoriť aplikáciu, ktorá sa svojím vzhľadom bude do portálovej stránky hodiť. Dizajnér pritom nemá kontrolu nad celou stránkou, nevie, čo je na nej umiestnené a nevie ani to, aký vzhľad bude mať stránka, na ktorú užívateľ portlet dá. Riešením tohto problému je nedefinovať portletu vlastné kaskádové štýly a použiť pri zobrazovaní priamo štýly poskytované portálom a jeho prvky vzhľadu (typicky sa pri prepnutí vzhľadu portálu nemenia názvy vystavených tried či identifikátory elementov, mení sa len ich vzhľad).

Na strane druhej máme iné portály. Portlety sú často vyvíjané podľa portletových štandardov a predpokladá sa teda možnosť nasadiť portlet aj na iný portál ako ten, ktorý používa vývojár pri vývoji či dizajnér pri tvorbe dizajnu. Iný portál samozrejme označuje svoje elementy inak a poskytuje iné

26. jQuery Autocomplete, `http://docs.jquery.com/Plugins/autocomplete`.
27. jQuery UI Datepicker, `http://jqueryui.com/demos/datepicker/`.
28. Cascading Style Sheets, `http://www.w3.org/Style/CSS/`.
29. World Wide Web Consortium, `http://www.w3.org/`.

štýly. Preto ak sa spoľahneme výhradne na štýly poskytované konkrétnym portálom, môže mať náš portlet pri nasadení na iný produkt nepekný, často až úplné rozbitý vzhľad. Z hľadiska prenosnosti a platformovej nezávislosti by teda malo najväčší význam zahrnúť v portálovej aplikácii vlastné súbory *.css*, ktoré by portlety využívali. Dôsledkom by teda bolo samozrejme to, že by sa naše portlety zrejme nehodili do celkového vzhľadu portálu.

Samozrejme netreba zabúdať, že isté rozdiely v zobrazovaní môžu byť aj na strane užívateľa. Väčšina bežne používaných prehliadačov síce podporuje CSS v rozumnej miere, napriek tomu však môže dochádzať k rozdielom v zobrazovaní niektorých prvkov a pre zachovanie maximálnej použiteľnosti je potrebné si na to dávať pozor obzvlášť vtedy, ak používame najnovšie technológie.

V zásade existuje niekoľko prístupov k riešeniu uvedeného problému. Je možné pokúsiť sa navrhnúť pre portlet vlastný štýl, ktorý je dostatočne skromný na to, aby celkový vzhľad portálovej stránky nerušil bez ohľadu na to, aký vzhľad to je. Dôsledkom je samozrejme to, že portlet s veľkou pravdepodobnosťou nebude vyzerať veľmi pekne na žiadnom portáli. Ďalšou variantou je vytvoriť štýlov viac a použiť konkrétny podľa aktuálnej šablóny, prípadne dať užívateľovi možnosť výberu z niekoľkých štýlov a nechať tak rozhodnutie o celkovej vhodnosti do prostredia na ňom. Je tiež možné využívať štýl konkrétneho portálu, o ktorom predpokladáme, že na ňom bude aplikácia nasadená a poskytnúť alternatívny pevný štýl obsiahnutý v portlete v prípade, že portlet bude nasadený na inom portáli a my nejakým testom (napríklad testom na prítomnosť istých elementov) zistíme nedostupnosť očakávaného štýlu. Všetky tieto riešenia sú pomerne pracne a obvykle zahŕňajú vytvorenie viacerých štýlov, čo nás vedie k poslednej alternatíve – k portálovej aplikácii pribaliť jednoduchý štýl, ktorý sa postará o základné, relatívne zmysluplné zobrazenie portletu za každých podmienok. Tento štýl obsahuje len vzájomné odsadenie elementov, okraje a pod. Hlavné prvky vzhľadu (písmo, farby a pod.) sú načítané z portálu.

Keďže predpokladáme nasadenie aplikácie IEP Manager primárne na portáli Liferay 5 používanom v IBA CZ, bola zvolená práve posledná varianta. Primárne sú využité prvky vzhľadu portálu, ale modul IEP Manager Portlets obsahuje v adresári *css* aj vlastné kaskádové štýly pre základné vlastnosti a rozloženie elementov.

Poznamenajme, že okrem dodržiavania štandardných zásad dizajnu webu je tiež potrebné si uvedomiť isté špecifiká v súvislosti s použitím kaskádových štýlov v prostredí portálu. Hlavne je nutné si uvedomiť to, že portlet nemá kontrolu nad celou stránkou, zväčša bude na stránke umiestnený spolu s inými portletmi a bude mať k dispozícii len malú plochu na zobrazenie

obsahu. Je preto dôležité, aby bol dizajn minimalistický a elementy neboli umiestnené na stránke absolútne. Taktiež je veľmi vhodné používať menné priestory, keďže nechceme, aby dochádzalo ku konfliktom medzi štýlmi našej a iných aplikácií. Nesmieme tiež zabudnúť na to, že portlet negeneruje priamo hlavičku stránky, čo je práve miesto, kde sa zvykne cesta k CSS súboru deklarovať. V prípade aplikácie IEP Manager je to riešené tak, že portlety obsahujú cestu k CSS v súbore *liferay-portlet.xml* a o umiestnenie príslušnej deklarácie do hlavičky stránky sa stará samotný portál.

7.2.7 Display tag library

Display tag library[30] je knižnica značiek s otvoreným zdrojovým kódom distribuovaná pod licenciou *Artistic License*[31]. Knižnica je určená na použitie na prezentačnej vrstve v aplikácii založenej na MVC a typicky sú jej značky využívané v JSP stránkach. Má jediný účel – šikovne zobrazovať tabuľky.

Display tag library v zásade poskytuje všetky základné funkcie, ktoré môže vývojár pri zobrazení dát vo forme tabuľky potrebovať. Stačí jej dať k dispozícii zoznam objektov a knižnica sa automaticky postará o zobrazenie dát vo forme stĺpcov, ich zoskupenie, export, triedenie, stránkovanie a nastavenie vzhľadu tabuľky, pričom všetko je konfigurovateľné. Je možné napríklad definovať vlastný vzhľad či popisy údajov lokalizovať. Nastaviteľné je tiež triedenie a stránkovanie. Knižnica pri východzom nastavení jednoducho prijíma zoznam všetkých dát, zoradí ich podľa istého kritéria a zobrazí príslušnú časť. Získavať pri každom dotaze z databázy všetky údaje vo väčšine prípadov nie je z pohľadu výkonu samozrejme vhodné riešenie, knižnica ale poskytuje API, pomocou ktorého je možné definovať externé triedenie či stránkovanie a vyriešiť to na úrovni aplikácie.

IEP Manager umožňuje správu niekoľkých entít a zobrazovanie údajov v tabuľke je v tomto prípade vhodné. Display tag library poskytuje niekoľko užitočných funkcií pre tento účel a je možné ju nastaviť pre použitie v portálovom prostredí, čo bolo hlavným dôvodom jej použitia v aplikácii.

7.3 Možnosti rozšírenia

Očakáva sa, že IEP Manager bude ďalej vyvíjaný a existuje hneď niekoľko oblastí, kde má veľký potenciál na zlepšenie. Vhodnými možnosťami rozšírenia tejto aplikácie sú napríklad:

---

30. Display tag library, `http://www.displaytag.org/`.
31. *Artistic License*, `http://www.opensource.org/licenses/artistic-license-1.0`.

- Generovanie reportov pomocou externých nástrojov – dôležitou súčasťou aplikácie IEP Manager je analýza dát a generovanie reportov týkajúcich sa vzdelávacích plánov, ktoré môže využiť manažér pre vzdelávanie napríklad pri plánovaní rozpočtu. O generovanie týchto reportov sa stará portlet Reporter, ktorý v súčasnosti zobrazuje niekoľko základných štatistík vhodných pre plánovanie. Aktuálne zobrazované štatistiky slúžia takpovediac na ukážku. Podrobnejšie analýzy neboli implementované, pretože v súčasnosti už prebieha vývoj univerzálneho nástroja na báze *BI*[32], ktorý bude schopný vykonávať pokročilú analýzu dát a generovať príslušné reporty. Očakáva sa, že tento nástroj bude využívať aj IEP Manager, čo mu bude umožňovať robenie podrobnejších analýz, získanie väčšieho množstva relevantných informácií a v konečnom dôsledku zobrazenie nových štatistík.

- Export správ – v súčasnosti sú správy týkajúce sa vzdelávacích plánov zobrazené priamo v portlete Reporter predovšetkým vo forme tabuliek. Očakáva sa, že s príchodom nového nástroja na tvorbu správ bude možné získané údaje exportovať do dokumentov (HTML a PDF), napríklad aj s využitím grafov.

- Konvertor mien – cena aktivít a ich inštancií nemusí byť zadaná v rovnakej mene. Pri tvorbe správ a generovaní štatistík týkajúcich sa rozpočtu je však potrebné ceny sčítať a teda do rovnakej meny konvertovať. Bolo by ideálne prepojiť portlet s nejakým nástrojom na konverziu mien tak, aby si portlet nemusel tieto hodnoty vôbec uchovávať a využíval automaticky aktualizovaný kurz z externého zdroja.

- Zvýšenie integrácie s projektom Skill Manager – IEP Manager a Skill Manager sa zaoberajú podobnou oblasťou. Správu plánov a znalostí by bolo vhodné prepojiť.

- Užívateľsky prívetivá konfigurácia – v aktuálnej implementácii je viacero vlastností nastavených cez konfiguračné súbory (*.properties*). Bolo by vhodné presunúť viac nastavení do portletov a vytvoriť napríklad samostatný portlet, v ktorom by mohol administrátor nastaviť konfiguráciu jednotlivých portletov – napríklad cestu k databáze a východziu konfiguráciu zobrazenia dát.

32. Business Intelligence, `http://en.wikipedia.org/wiki/Business_intelligence`.

8 Záver

Cieľom tejto práce bolo popísať problematiku vývoja aplikácií v prostredí podnikového portálu a aplikovať popísané postupy pri návrhu a implementácii aplikácie umožňujúcej správu vzdelávacích plánov. Práca bola vypracovaná v rámci Sdružení průmyslových partnerů Fakulty informatiky pre spoločnosť IBA CZ a implementovaná aplikácia je prispôsobená potrebám tejto spoločnosti.

V práci predstavujeme koncept a uznávanú klasifikáciu portálov s osobitým dôrazom kladeným na podnikové portály. Použitie podnikových portálov vysvetľujeme na príklade a na základe skúseností s vývojom pre túto platformu analyzujeme výhody a nevýhody ich nasadenia v organizácii. Stručne popisujeme aktuálnu situáciu na trhu podnikových portálov a predstavujeme základnú terminológiu súvisiacu s vývojom portálových aplikácii. Podrobnejšie prechádzame hlavné špecifikácie popisujúce vývoj portletov nad platformou Java – JSR-168 a JSR-286. Dôraz je pritom kladený na rozdiely medzi tvorbou portálových a klasických webových aplikácií a podrobnejšie popisujeme tie aspekty vývoja, ktoré boli využité pri tvorbe aplikácie pre správu vzdelávacích plánov. Snažili sme sa popísať spomínané metódy a princípy tak, aby aj človek, ktorý priamo portlety nevyvíjal, získal nielen pomerne komplexnú predstavu o fungovaní týchto aplikácií podľa špecifikácií, ale bol tiež oboznámený s poznatkami plynúcimi zo skúseností s vývojom pre platformu portálov. Myslíme si, že tento cieľ sa nám podarilo splniť.

Dôležitou súčasťou práce je vlastná analýza problematiky správy vzdelávacích plánov v IT firme a ich významu pre zamestnancov i zamestnávateľa. Vysvetľujeme druhy vzdelávacích plánov a kritériá, ktoré by plány mali spĺňať. Špeciálna pozornosť je venovaná popisu osobitostí správy plánov v spoločnosti IBA CZ predovšetkým na základe informácií z konzultácií. Analyzujeme požiadavky kladené na aplikáciu, výstupom čoho je špecifikácia a popis prípadov užitia aplikácie. Popisujeme systémy fungujúce v IBA CZ, ktoré s aplikáciou spolupracujú a formulujeme návrh s dôrazom na architektúru a dátový model aplikácie. Uvádzame tiež popis rozhrania a samotnej implementácie aplikácie, kde prinášame prehľad základných rámcov a technológií, ktoré sme použili. Vždy zdôrazňujeme, kde konkrétne v aplikácii sme jednotlivé technológie uplatnili a aký bol hlavný dôvod ich výberu. Voľba technológií sa aj s postupom času ukázala ako dobrá.

Výsledná aplikácia IEP Manager bola rozdelená na moduly IEP Manager Backend a IEP Manager Portlets, pričom jej funkcionalita je poskytovaná v podobe troch navzájom komunikujúcich portletov. Portlet Administrator

je určený manažérovi pre vzdelávanie a jeho asistentovi, ktorým umožňuje pridávať, odoberať a upravovať základné entity vystupujúce vo vzdelávacích plánoch – samotné plány, tréningové aktivity, inštancie aktivít, typy aktivít a voliteľnosti položiek plánov. Portlet Viewer umožňuje zamestnancovi spoločnosti zobrazenie jeho plánu. Líniový manažér tu môže vidieť a schvaľovať plány svojich podriadených, zatiaľ čo manažér pre vzdelávanie a jeho asistent môžu vidieť a spravovať položky plánov ľubovoľného zamestnanca. Portlet Reporter zase umožňuje manažérovi pre vzdelávanie generovať štatistiky týkajúce sa plánov zamestnancov firmy. Tým boli splnené všetky funkcie požadované zadaním.

Aj keď aplikácia IEP Manager je použiteľná na správu individuálnych vzdelávacích plánov, nie je to uzavretý projekt. V práci spomíname niekoľko možností rozšírenia súvisiacich predovšetkým s integráciou aplikácie s ďalšími nástrojmi a implementácia bude pokračovať v spolupráci so spoločnosťou IBA CZ. Očakáva sa, že aplikácia bude následne v jej prostredí nasadená.

Literatúra

[1] ABDELNUR, A., HEPPER, S. *JSR 168: Portlet Specification* [online]. 2003. 07.10.2003 [cit. 18.05.2011]. Dostupné na internete: <http://www.jcp.org/en/jsr/detail?id=168>.

[2] ADAMS, P. et al. *IBM Workplace Collaborative Learning 2.6*. 1st ed. 2006. 378 p. ISBN 0738495905.

[3] BURGESS, S., DAVISON, A., TATNALL, A. *Internet technologies and business*. Melbourne, Australia : Data Publishing, 2003. 214 p. ISBN 1875415823.

[4] Cascading Style Sheets. In *Wikipedia* [online]. 18.05.2011 [cit. 18.05.2011]. Dostupné na internete: <http://en.wikipedia.org/wiki/Cascading_Style_Sheets>.

[5] DELISLE, P. *JSR 52: A Standard Tag Library for JavaServer Pages* [online]. 2003. November 2006 [cit. 18.05.2011]. Dostupné na internete: <http://www.jcp.org/en/jsr/detail?id=52>.

[6] DELISLE, P., LUEHE, J., ROTH, M. *JSR 245: JavaServer Pages 2.1* [online]. 2006. 08.05.2006 [cit. 18.05.2011]. Dostupné na internete: <http://www.jcp.org/en/jsr/detail?id=245>.

[7] EJB 3.0 Expert Group. *JSR 220: Enterprise JavaBeans,Version 3.0* [online]. 2006. 02.05.2006 [cit. 18.05.2011]. Dostupné na internete: <http://www.jcp.org/en/jsr/detail?id=220>.

[8] FOWLER, M. *Inversion of Control Containers and the Dependency Injection pattern*. 2004. [cit. 18.05.2011]. Dostupné na internete: <http://martinfowler.com/articles/injection.html>.

[9] GARRETT, J.J. *Ajax: A New Approach to Web Applications*. 18.02.2005 [cit. 18.05.2011]. Dostupné na internete: <http://www.adaptivepath.com/ideas/e000385>.

[10] HEPPER, S. *JSR 286: Portlet Specification 2.0*. 2008. 25.01.2008 [cit. 18.05.2011]. Dostupné na internete: <http://www.jcp.org/en/jsr/detail?id=286>.

[11] Java Persistence Expert Group. *JSR 317: Java Persistence API, Version 2.0* [online]. 2009. 10.11.2009 [cit. 18.05.2011]. Dostupné na internete: <http://www.jcp.org/en/jsr/detail?id=317>.

[12] JOHNSON, R. et al. *Spring Framework Reference Documentation 3.0*. 2010. 20.10.2010 [cit. 18.05.2011]. Dostupné na internete: <http://static.springsource.org/spring/docs/3.0.5.RELEASE/reference/>.

[13] *jQuery* [online]. [cit. 18.05.2011]. Dostupné na internete: <http://jquery.com/>.

[14] KAČALA, J. et al. *Krátky slovník slovenského jazyka*. 4. vydanie. Bratislava : Veda, 2003. 988 p. ISBN 80-224-0750-X.

[15] KING, G. et al. *Hibernate Reference Documentation*. 2011. 06.04.2011 [cit. 18.05.2011]. Dostupné na internete: <http://docs.jboss.org/hibernate/core/3.6/reference/en-US/html/>.

[16] Liferay Portal [online]. [cit. 18.05.2011]. Dostupné na internete: <http://www.ibacz.eu/Liferay-Portal>.

[17] SCHELPT, J., WINTER, R. Enterprise portals und enterprise application integration. In *HMD?Praxis der Wirtschaftsinformatik*. 2002. pp. 6-20.

[18] SIEBER, S., VALOR, J. Horizontal portal strategies: Winners, losers and survivors. In *15th Bled Electronic Commerce Conference – eReality: Constructing the eEconomy*. Bled, Slovenia, 2002. pp. 166-178.

[19] TATNALL, A. Portals, portals everywhere. In *Web portals: The new gateways to internet information and services*. 2005. pp. 1-14.

[20] *The Difference Between Strategic and Tactical Planning* [online]. 29.02.2009 [cit. 18.05.2011]. Dostupné na internete: <http://www.morebusiness.com/strategic-planning>.

[21] Web portal. In *Wikipedia* [online]. 07.05.2011 [cit. 18.05.2011]. Dostupné na internete: <http://en.wikipedia.org/wiki/Web_portal>.

Register