Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Bakalářská práce

Rich Internet Applications

Petr Cihelka

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "Rich Internet Applications" jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu literatury na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil(a) autorská práva třetích osob.

V Praze dne 30.4.2009 _____

Poděkování

Rád bych touto cestou poděkoval vedoucímu práce Ing. Petru Bendovi za odborné vedení a cenné připomínky při zpracování této práce. Dále bych rád poděkoval Ing. Janu Vondrusovi a Ing. Tomáši Třebickému za odborné konzultace a pomoc při orientaci v rozebírané problematice.

Rich Internet Applications

Rich Internet Applications

Souhrn

Námětem této práce jsou moderní technologie pro tvorbu aplikací souhrnně označované pojmem Rich Internet Applications (RIA). Vysvětlena je historie vzniku tohoto pojmu společně s výčtem organizací, které se na jejich vývoji podílely a stále podílejí. Důraz je kladen na jednotlivé směry, které lze při vývoji aplikací uplatnit, na jejich zhodnocení a užití v konečných RIA. Následuje popis a vyhodnocenní platformem, které jsou využívány pro tvorbu RIA. Kromě platformy Adobe Flex je zde také představeno vývojové prostředí Adobe Flex Builder a muliplatformní běhové prostředí Adobe AIR. V neposlední řadě jsou zde vysvětleny základní principy programování pro tuto platformu a na praktickém přikladu demonstrován postup tvorby jednoduché RIA.

Klíčová slova: Rich Internet Applications, Adobe Flash, Adobe Flex, Microsoft Silverlight, Adobe AIR, OpenLazslo, Google Web Toolkit, AJAX

Summary

The work presented in this thesis has been motivated by better understading and subsequent utilization of modern tools for application development called Rich Internet Applications (RIAs). The history of the name as well as the pioneering companies and organizations will be introduced. Both mainstream technological trends are discussed and evaluated with respect to end-user applications. RIAs can be implemented using various platforms that are throroughly described and assessed. Among these are Adobe Flex, a comprehensive development environment Adobe Flex Builder, and a multiplatform runtime environtment Adobe AIR. With regards to Adobe Flex, the basic and most important programming rules and principles are explained on a real example.

Keywords: Rich Internet Applications, Adobe Flash, Adobe Flex, Microsoft Silverlight, Adobe AIR, OpenLazslo, Google Web Toolkit, AJAX

1 Obsah

1 Obsah 3
2 Úvod 4
3 Cíl práce a metodika 6
RIA aplikace - vývoj a historie 8
3.1 Definice a vysvětlení pojmu 8
3.2 Historie RIA 9
3.2.1 Společnost Macromedia 9
3.2.2 Java 10
3.2.3 Microsoft 10
3.2.4 HTML a DHTML 10
3.3 Základní rysy a charakteristika RIA 11
3.3.1 Výhody a nevýhody RIA technologií 14
3.3.2 Volba vhodné technologie 16
3.4 Web 2.0 a RIA 18
4 Platformy pro vývoj RIA aplikací 20
4.1 Adobe Flash Platform 21
4.1.1 Hodnocení 23
4.2 Microsoft Silverlight 23
4.2.1 Hodnocení 24
4.3 OpenLaszlo 25
4.3.1 Hodnocení 25
4.4 ExtJS 25
4.4.1 Hodnocení 26
4.5 Bindows 26
4.5.1 Hodnocení 26
4.6 Curl 27
4.6.1 Hodnocení 27
4.7 Google Web Toolkit 27
4.7.1 Hodnocení 28
5 Architektura RIA aplikací 29
5.1 Adobe Flash Platform 29
5.1.1 Adobe Flex 29
5.1.2 Zápis kódu 30
5.1.3 Adobe Flex Builder 33
5.1.4 Adobe AIR 35
5.1.5 Serverová řešení 35
6 Závěr 36
7 Slovníček pojmů 39
8 Seznam literatury 44

2 Úvod

Je tomu již téměř 40. let, kdy agentura ARPA položila základy k vytvoření experimentální sítě ARPANET, která se později transformovala do celosvětového fenoménu, dnes známého pod pojmem Internet. V prvopočátku byla síť ARPANET určena pro vojenské účely armády USA a její velikost byla omezena na několik počítačů a směrovacích prvků, které řídily provoz síťové komunikace. Svou topologií byla síť decentralizovaná, neměla tedy žádné centrum a případný útok na jeden uzel neznamenal žádné závažné narušení funkčnosti.

Stejně jako velké události, bitvy či myšlenky mají své významné osobnosti, tak i při vytváření a vývoji technologií, jež dopomohly ke zrodu Internetu do podoby, v jaké jej známe dnes, se podílelo několik významných lidí. Prvními, kteří vytvořili důležitý technologický zlom ve zrodu Internetu, byli Vinton Gray Cerf a Bob Kahnem. Navrhli a vytvořili protokol TCP/IP, který měl zajišťovat, a do dnešní doby zajišťuje, provoz internetové komunikace [1]. Dalším důležitým mužem v technologickém rozvoji byl Paul Mockapetris, který vyvinul a spustil protokol DNS, jenž měl a dodnes má za úkol obousměrný překlad číselných IP adres na doménová jména [2]. Poslední krok spočíval v zabránění komerčního využití vojenské části ARPANETu a vyústil k vytvoření samostatného výhradně pro-vojensky-orientovaného systému zvaného MILNET.

Samotný vznik pojmu Internet se datuje k roku 1987. V této době bylo připojeno do sítě již 27 000 počítačů, jednalo se především o univerzitní a vědecká pracoviště; komerční využití sítě bylo v této době zanedbatelné. O dva roky později přichází další významná osobnost, Tim Berners-Lee, který představuje koncept WWW a o dva roky později, tj. roku 1991, jej prakticky realizuje v laboratořích CERN [1]. Toto lze považovat za poslední technologický krok, který byl nutný pro rozvoj Internetu, tak jak ho známe dnes. Zavedením konceptu WWW započala nová epocha Internetu a došlo ke komercializaci internetu. Uživatelům bylo umožněno publikovat své texty na síti, komerční subjekty, i přes počáteční nedůvěru v toto médium,

začínaly vytvářet své prezentace. Zprvu se jednalo o jednoduché internetové stránky, ve kterých převažoval text. Obsah byl přizpůsoben tehdejším datovým rychlostem, ale s postupem času a zvyšováním průchodnosti datových spojů bylo nasazováno více grafických prvků. Zvyšování průchodnosti bylo také zapříčiněno velkým zájmem uživatelů, který překonal všechna očekávání. Nárůst uživatelů rok od roku řádově narůstal, pro představu: v roce 1996 měl internet 56 milionů uživatelů, o 4 roky později již 250 milionu uživatelů a o další 2 roky později již 600 milionů uživatelů. V současnosti se křivka počtu připojených uživatelů exponenciálně zvyšuje a odhadovaný počet je 1,5 bilionu počítačů.

Tento enormní zájem uživatelů o služby Internetu vyvolal velký tlak na možnosti internetových technologií. Uživatelé zvyšovali nároky na služby internetových prezentací a původní, převážně statické textové internetové stránky přestaly dostačovat potřebám. Statické stránky byly postupně vytlačovány technologiemi, umožňujícími generování dynamických stránek (např. PHP, .NET). Samotný jazyk HTML byl doplněn o možnosti stylování pomocí CSS a skriptování pomocí JavaScriptu. Někteří výrobci nebyli i tak spokojeni s možnostmi, které poskytoval jazyk HTML, a započali s vývojem vlastních proprietárních řešení.

Od internetových stránek byla vyžadována větší interaktivita, zvyšovaly se nároky na generování obsahu stránek dle požadavků či vstupů uživatele. Vývojáři na tyto požadavky reagovali vytvářením technologií a internetových aplikací, které poskytovaly uživatelům služby a komfort desktopových aplikací a v mnohých případech je, z hlediska funkčnosti, překonávaly. Vývojáři a uživatelé si začali uvědomovat výhody aplikací běžících v rozhraní běžného internetového prohlížeče. Uživatelé mohli s aplikací pracovat téměř kdekoli, kde byl dostupný internetový prohlížeč a připojení k síti. Díky centralizaci takových technologií měli vývojáři usnadněn vývoj, správu a distribuci nových verzí. Nový technologický směr dal vzniknout nové oblasti internetových aplikací, které jsou dnes známé pod označením RIA. Uživatel si jejich existenci možná ani neuvědomuje, ale každodenně je využívá; jedná se například o aplikace typu YouTube, GMail, Microsoft Outlook Online,

Microsoft Hotmail, FaceBook, MySpace atp. Pojmenováním tohoto směru bylo docíleno jasného vymezení rysů a charakteristik těchto aplikací.

3 Cíl práce a metodika

Cílem bakalářské práce je uvedení pojmů RIA a RIA aplikace. Dále bude uvedena historie a vznik tohoto moderního technologického směru. Práce si klade za cíl, popsat historii vzniku těchto aplikací, vymezit jejich základní rysy, představit technologie, pomocí kterých lze tyto aplikace vyvíjet. Osvětlit možné přístupy, k tvorbě RIA aplikací a vyjmenovat klady a zápory. Posledním cílem je zhodnocení jednotlivých přístupů a jsou předložena doporučení, která lze použít při volbě vhodného technologického směru, při návrhu a vývoji RIA.

V první části bude nejprve představen a vysvětlen samotný pojem RIA. Budou představeny jednotlivé organizace a technologie, které se na rozvoji podílely a podpořili tak vznik tohoto odvětví. Dále budou vysvětleny základní rysy a charakteristiky RIA aplikací. Představeny budou dva základní směry vývoje těchto aplikací. Bude představena technologie AJAX a její princip a výhody budou demonstrovány na příkladu. Taktéž budou představeny výhody a nevýhody RIA technologií jako celku a dojde i k porovnání výhod a nevýhod jednotlivých technologických směrů. Na závěr budou definována doporučení, která mohou pomoci při volbě vhodné technologie pro tvorbu RIA aplikací.

V druhé části bude představena široká škála platforem, které poskytují nástroje a umožňují tak tvorbu RIA aplikací. U každé z jednotlivých technologií budou vyjmenovány klady a zápory.

Ve třetí části bude představena zvolená dominantní technologie Adobe Flash/Flex. Bude představena architektura frameworku. Bude představena architektura frameworku a základních principů využitelných při vývoji aplikací

pomocí této technologie. Představeno bude vývojové prostředí, základní syntaxe jazyka a přehled základních komponent, použitelných pro tvorbu RIA aplikací.

V závěrečné části bude provedeno zhodnocení jednotlivých technologických přístupů, které lze při vývoji RIA aplikací zvolit, bude provedeno i ekonomické zhodnocení a zamyšlení se nad budoucím vývojem těchto technologií.

V bakalářské práci bude autor vycházet z informací nabytých během studia na České zemědělské univerzitě v Praze a dále ze studia odborné literatury a účasti na konferencích, které byly na toto téma pořádány. Také bude využívat článků publikovaných v odborných internetových magazínech.

RIA aplikace - vývoj a historie

3.1 Definice a vysvětlení pojmu

RIA – tato zkratka znamená **Rich Internet Applications**, což lze do českého volně přeložit jako „Bohaté Internetové Aplikace“, bohaté především kolekcemi funkcí, které uživateli přinášejí komfort pro práci.

Vznik RIA aplikaci byl logickým vyústěním takzvaného internetového boomu, který nastal kolem roku 1990. V této době začalo být internetové medium dostupné široké veřejnosti a postupem času se obyčejné statické stránky měnily na dynamické, které uživateli přinášely jistou úroveň interaktivity. V průběhu těchto změn začínalo docházet k jevu, kdy internetové aplikace běžící v běžném internetovém prohlížeči byly schopny plně nahradit desktopové aplikace. Aplikace v internetovém prohlížeči přinášejí oproti desktopovým aplikacím mnoho výhod od jednoduché správy, přes jednoduchost úprav funkčnosti aplikace a multiplatformností konče.

RIA aplikace využívají ke svému běhu standardních prostředků internetového prohlížeče, jako je HTML, CSS, JavaScript, AJAX a dále také prostředků, které nabízejí proprietární moduly třetích stran, např. dnes již známý Adobe Flash či Microsoft Silverlight. Aplikace postavené na těchto základech poskytují potřebný prostor pro vývoj aplikací, které dokáží uspokojit nároky dnešního uživatele.

Je nutné podotknout, že v dnešní době rozvoje mobilních zařízení, které svými parametry začínají dosahovat stavu, kdy bude možné a uživatelsky přijatelné provozovat RIA aplikace i na těchto zařízeních.

3.2 Historie RIA

Pojem Rich Internet Applications (RIA) spatřil světlo světa již v roce 1990 a jeho autorem byla firma Macromedia. Protože bylo jasné, že současné HTML nebude v budoucnu dostačovat požadavkům, tak již v této době začaly firmy pracovat na proprietárních prohlížečových modulech, které měly v budoucnu odstranit limity statického HTML.

3.2.1 Společnost Macromedia

U firmy Macromedia byl výsledkem tohoto snažení Macromedia Flash, který byl v prosinci roku 1996 uvolněn ve verzi 1.0. Tato verze byla přijata s rozpaky a zprvu poskytovala pouze nástroje pro animaci obrázků a vektorových objektů. V počátečních verzích byl Macromedia Flash využíván převážně pro tvorbu grafických prezentací a reklamních bannerů. Jeden ze zásadních zlomů přichází v roce 2002, kdy je vydána klíčová verze produktu Macromedia Flash Player, která nyní podporuje protokoly (AMF, podpora pro SOAP) pro komunikaci se vzdálenými službami, podporuje streaming on-demand/live videa a byla zavedena podpora pro komponenty a sdílené knihovny. Zároveň s uvedením nové verze Macromedia publikuje dokument, ve kterém specifikuje základní charaktery RIA aplikaci a požadavky pro jejich vývoj. Firma Macromedia pokračuje i nadále s vývojem svého Macromedia Flash, jenž díky novým vlastnostem nabývá na oblibě. V roce 2003 je představen nový Macromedia Flash ve verzi 7, který kromě jiných novinek podporuje i jazyk Action Script 2.0, jenž po dlouhých očekáváních přináší podporu objektově orientovaného programování. O dva roky později nastává další významný milník, krom vydání nové verze Macromedia Flash je dne 3. prosince 2005 odkoupena firmou Adobe Inc. Dále je v tomto roce vydán, ještě pod hlavičkou firmy Macromedia, nový produkt s názvem Macromedia Flex, který má být primárně určen pro tvorbu RIA aplikací. O další dva roky později, v roce 2007, je vydána nová verze Flash Playeru č. 9.0 avšak nyní již pod názvem Adobe Flash, která přináší kromě zlepšení integrovatelnosti s produkty Adobe také podporu pro Action Script 3.0.

V tomto roce Adobe přináší další revoluci v RIA aplikacích a prezentuje nový produkt Adobe AIR, který umožňuje přenést aplikace z internetového prohlížeče na uživatelské desktopy jakožto plnohodnotné aplikace, se všemi výhodami, které běžné aplikace mají (přístup k souborovému systému, přístup ke zdrojům systému, atp.) [11] [13] [15].

3.2.2 Java

Nebyla to však pouze firma Macromedia, později Adobe Inc., která se snažila o vytvoření prostředí pro vývoj RIA aplikací. Společně s Macromedia Flash existovaly i další proprietální moduly, např. Java Applety, avšak na poli RIA aplikaci se příliš neprosadily. Jednalo se o malé aplikace psané v jazyce Java, které byly vkládány do HTML stránky a pomocí virtuálního stroje JVM (Java Virtual Machina) spuštěny v prohlížeči. Nutnou podmínkou byla přítomnost JRE (Java Runtime Environment)

3.2.3 Microsoft

Firma Microsoft také nezůstala pozadu a ve snaze zmírnit dominanci Adobe Flash představila v roce 2007 vlastní technologii jménem Microsoft Silverlight 1.0. V současné době je k dispozici Microsoft Silverlight ve verzi 2.0. Produkt firmy Microsoft se těší veliké oblibě, zejména na poli vývojářů na platformě .NET, avšak nese s sebou břemeno plynoucí z menšího rozšíření mezi uživateli [12].

3.2.4 HTML a DHTML

Vývojem prošel i samotný jazyk HTML, který ve spojení se skriptovacím jazykem JavaScript poskytoval a poskytuje i nyní, dobré zázemí pro tvorbu RIA aplikací. Vývoj jazyka HTML ustrnul na verzi HTML 4.1 a v současné době se čeká na finální podobu budoucího HTML 5, které by mělo nahradit současný HTML standard. Dalším krokem byl vznik DOM (Document Object Model), který představuje objektově orientovanou reprezentaci internetové stránky v podobě stromové struktury jejích HTML tagů. Tato struktura muže být dále modifikována

skriptovacím jazykem JavaScript. Spojení technologií HTML, CSS, JavaScript a DOM dalo vzniknout DHTML, což je pouze rozšířená verze jazyka HTML doplněná o možnosti změny obsahu a struktury stránek za běhu. Na základě tohoto vývoje bylo možné vytvořit frameworky (např. ExtJS), jež poskytují bohaté kolekce pro tvorbu RIA aplikací a jsou tak schopné odstranit omezení plynoucí, ze samotného jazyku a HTML.

3.3 Základní rysy a charakteristika RIA

Jak uvedeno výše, firma Macromedia definovala vlastnosti, které by měly RIA aplikace splňovat, aby je bylo možné opravdu nazvat RIA aplikacemi. Přestože tato definice obsahuje doporučení a popis vlastností ke vztahu k tehdejšímu produktu firmy Macromedia, tj. Macromedia Flash (pozdeji Adobe Flash), lze většinu doporučení zobecnit [16]:

- komplexnost uživatelského rozhraní,
- uživatelsky přívětivé chování,
- získávat a odesílat data pouze pokud je třeba,
- komfort funkcí odpovídající klasickému desktopovému řešení (implementace funkcí, na které je uživatel zvyklý z operačního systému, tj. drag&drop, nápověda, klávesové zkratky).

Z předchozího textu je patrné, že k realizaci RIA aplikací lze přistoupit ze dvou směrů:

- využití proprietárního prohlížečového modulu
- využití současných technologií, které poskytují prohlížeče

Půjde-li tvůrce RIA aplikací cestou využití současných technologií, které nabízejí prohlížeče, musí se připravit na zdolání několika překážek. První překážkou, na kterou vývojář narazí, jsou omezenější možnosti prezentačních technologií, jako je HTML, CSS. Druhý a zásadnější problém představuje samotný protokol HTTP, který pracuje s modelem žádost/odpověď. Sebemenší změna stavu (nahrání dat, validace, aktualizace, uložení dat) u klienta musí vyvolat požadavek na server, který

jej musí obsloužit a zpětně vrátit předešlá data, což způsobí obnovení stavu stránky. Tento problém lze odstranit použitím technologie AJAXu.

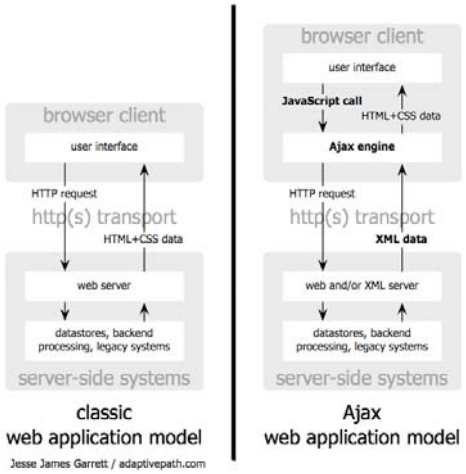
AJAX znamená Asynchronous JavaScript and XML. Ve skutečnosti není technologií nebo softwarovým produktem, jde o koncept, resp. návrhový vzor pro RIA aplikace, ale nejen pro ně [8].

Tento návrhový vzor využívá tří technologií:

- Dokument Object Model (DOM)
- XMLHttpRequest (JavaScriptová kolekce funkcí)
- HTML, CSS, JavaScript

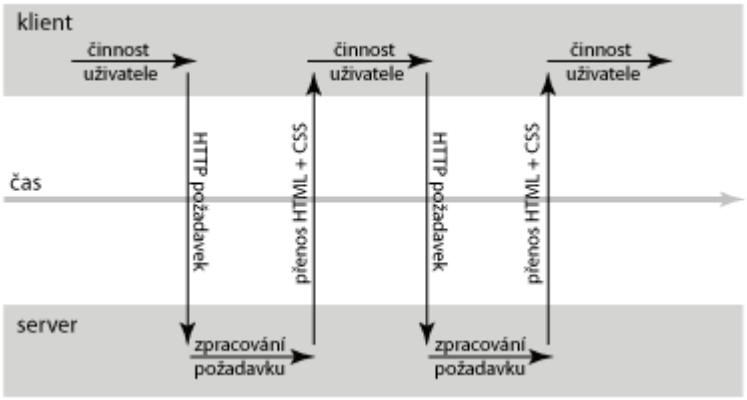
Při obecném pohledu na využití AJAXu je možno tvrdit, že celá jeho funkčnost (a tím i kouzlo) spočívá ve využití kolekce XMLHttpRequest, kterou poskytuje skriptovací jazyk JavaScript.

U odborné i laické veřejnosti může vyvstat otázka, co je tedy na AJAXu tak revolučního a proč je řešením problému, který způsobuje protokol HTTP. Využití AJAXu totiž poskytuje možnost asynchronního volání serveru a zpracování odpovědi aniž by bylo nutné překreslovat celou stránku, která akci vyvolala. Samotná technologie je znázorněna na obrázku č. 1.



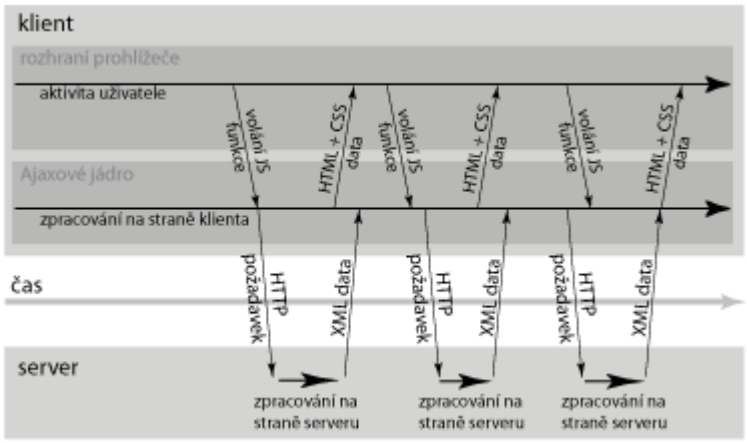
Obrázek 1 - rozdíl mezi využitím klasického modelu a AJAX modelu

Vysvětlení lze znázornit na příkladu. Vývojář potřebuje vytvořit jednoduchý formulář s několika prvky a tlačítko, které vyvolá akci, jež odešle formulářová data na server. Pokud bude realizovat formulář pomocí běžných nástrojů (tj. bez využití AJAXu) bude samotné odesílání formuláře probíhat takto. Po stisku tlačítka jsou data odeslána jako požadavek na server. Server data zpracuje a požadavek vyřídí, celá operace končí zasláním kompletního uživatelského rozhraní včetně dat. Všechny kroky tohoto procesu jsou vzájemně synchronizovány.



Obrázek 2 - Synchronní zpracování požadavku bez použití AJAXu

Pokud bude odeslání formuláře realizováno pomocí AJAXu, bude samotný proces odeslání velmi podobný s předchozím řešením, avšak s tím rozdílem, že samotné rozhraní formuláře může zůstat po dobu zpracování požadavku beze změn. Po stisknutí tlačítka, které slouží k odeslání formuláře na server, je vyvolána funkce či metoda AJAXového jádra, která vygeneruje požadavek a pomocí XMLHttpRequest jej odešle na server. Server požadavek zpracuje a vrátí odpověď. AJAXové jádro, na základě odpovědi ze serveru provede patřičné změny uživatelského rozhraní. Celé zpracování formuláře proběhlo v tomto případě asynchronně. V tomto případě je dokonce možné, po stisknutí tlačítka, vyslat na server požadavků několik a následně je zpracovat zcela odděleně.



Obrázek 3 - Asynchronní zpracování požadavku s použitím AJAXu

3.3.1 Výhody a nevýhody RIA technologií

Nespornou výhodou aplikací, které běží v běžném internetovém prohlížeči, je jejich dostupnost uživateli. Vzhled aplikace je stažen ze vzdáleného serveru, není tedy nutná instalace nativní desktopové aplikace a k běhu je potřeba pouze internetový prohlížeč, který má v dnešní době předinstalován každý systém. Díky této centralizaci se výrazně zjednodušuje vývoj a správa takovýchto aplikací a není potřeba další systém, který by zajišťoval distribuci nových verzí. Jako další výhodu lze označit snahu jednotlivých RIA platforem maximálně se přiblížit vzhledu a chování klasických aplikací, což výrazně ulehčuje uživatelům pochopit základní princip práce s takovýmto systémem.

Na základě informací, které zde byly doposud předloženy, je možné soudit, že RIA aplikace přinášejí pouze výhody. Z pohledu uživatele představují RIA aplikace problém v přístupnosti. V mnoho aplikacích nefungují klávesové zkratky, na které jsou uživatelé zvyklí, nefunguje tlačítko zpět a historie navštívených stránek se v takovýchto aplikacích mnohdy neukládá. Běžně lze narazit na problémy s indexací obsahu pro vyhledavače. Je-li na RIA aplikace nahlédnuto z pohledu vývojáře, lze opět narazit na problémy, tentokrát spojené se samotným vývojem aplikace. Přestože platformy pro tvorbu RIA aplikaci poskytuji bohaté kolekce funkcí, často vzniká problém s přílišnou komplexitou aplikace a tím i následným

vývojem, laděním a nasazením do produkčního prostředí. Dalším negativním faktorem je nemožnost pracovat v režimu Offline, tj. ve stavu, kdy je počítač odpojen od sítě. Mnohé aplikace jsou závisle na permanentním připojení k síti, a pokud není tento stav ošetřen tak jejich odpojení může způsobit problémy [8] [10].

Další výhody a nevýhody již nelze aplikovat dostatečně obecně, budou tedy vysvětleny dle výše zmíněného rozdělení a to dle aplikací:

- využívajících proprietárních zásuvných modulů
    - Výhody
        - Prohlížečový modul, na kterém je běh aplikace závislý, pochází od jednoho autoritativního dodavatele, čímž odpadá ladění aplikací pro různé prohlížeče,
        - Výkon aplikací, které využívají vlastních modulů je mnohdy daleko vyšší než u JavaScriptového interpretu prohlížeče,
        - Odpadají veškerá technologická omezení, která s sebou přináší spojení HTML/CSS/JavaScript,
        - Vývoj aplikací pomocí těchto zásuvných modulů bývá často efektivnější, protože jde o technologie vytvořené na míru vývojářům,
        - Bohaté kolekce nástrojů, které simulují rozhraní operačního systému,
        - Přítomnost kvalitních vývojářských nástrojů s podporou kontroly syntaxe a ladění.
    - Nevýhody:
        - Pro provoz aplikace je potřeba zásuvný prohlížečový modul, bez tohoto modulu nebude aplikace fungovat.
        - Ačkoli se aplikace tváří, že běží v prohlížeči, častokrát narazíme na problémy s ovládáním prohlížeče. Dochází k nefunkčnosti klávesových zkratek prohlížeče, není možné použít tlačítko „Zpět“, formulářová pole si nepamatují předchozí hodnoty, prohlížečový password manager taktéž nefunguje. Taktéž muže dojít ke zmatení uživatele, když dojde

k nahrání aplikace v nezašifrovaném spojení, avšak aplikace nadále komunikuje pomocí zabezpečených kanálů.

- využívajících technologií internetového prohlížeče
  - Výhody:
    - Multiplatformnost – lze tvrdit, že aplikace, které využívají běžných technologií poskytovaných prohlížečem, fungují na téměř všech operačních systémech,
    - Jednoduchost údržby a vývoje aplikací,
    - Aplikace není nutno kompilovat, vzhled aplikace je přenášen do internetového prohlížeče, který následně vykresluje grafiku aplikace.
  - Nevýhody:
    - Problémy s odladěním aplikace – vzhledem k velkému počtu prohlížečů, je velice náročné aplikace odladit tak, aby byl vzhled všude jednotný,
    - Problémy s výkonností JavaScriptu. Technologie využívající zásuvných prohlížečových modulů jsou v řádech výkonnější,
    - Technologická omezenost HTML/CSS. Tento problém vychází z historického zaměření těchto technologií,
    - Pomalý vývoj standardu, které by pomohly odstranit limity současných HTML/CSS technologií,
    - Slabá podpora AJAX vývoje ve vývojářských nástrojích

3.3.2 Volba vhodné technologie

Je patrné, jak jsou oba technologické přístupy, ač spadají pod společný termín RIA, v poměrně ostrém kontrastu. Při tvorbě RIA aplikací je nutné zamyšlení a stanovení cílů, které má aplikace v budoucnu splňovat. Je potřebné zvážit všechna pro a proti jednotlivých technologií, a dle toho vybrat vhodnou cestu pro budoucí aplikaci, popřípadě se vydat cestou kombinace obou technologií. Tato cesta bude pravděpodobně složitější, avšak umožní vývojářům využít maximum výhod z obou

technologických směrů. Toto rozhodnutí nemusí být vždy jednoduché a na první pohled jasné. Jako podporu při rozhodování lze využít jistých pravidel:

- technologický směr proprietárních zásuvných modulů použijeme za předpokladu, že aplikace je:
  - natolik specifická, že lze oželet ztrátu některých uživatelů, kteří nebudou ochotni zásuvný modul do svého internetového prohlížeče instalovat,
  - použita v intranetovém prostředí, ve kterém jsme schopni zajistit přítomnost zvoleného zásuvného modulu,
  - natolik komplikovaná, že by použití AJAXu bylo příliš složité nebo nákladné,
  - náročná na výkon.
- technologický směr využívající technologií internetového prohlížeče použijeme za předpokladu, že aplikace je:
  - použita pro širokou klientelu, ve které si nemůžeme dovolit odradit ztrátu uživatelů,
  - jednoduššího charakteru, pouze s jednoduchou aplikační logikou,
  - natolik specifická, že je prioritou funkčnost v jakémkoli prohlížeči.

Uvedená doporučení vycházejí ze zkušeností autora s vývojem aplikací a studiem literatury, která se tímto tématem zabývá. Na poli RIA aplikací lze najít výjimky, které se těmto doporučením vymykají. Je třeba si uvědomit několik pravidel, která platí všeobecně pro vývoj aplikací. I pro jednoduchou aplikaci s jedním formulářem lze vybrat robustní technologii, kterou například poskytuje Adobe Flex, avšak v tomto případě si bereme do ruky pomyslný kanón na vrabce. Technologie Adobe Flex je na takto jednoduché aplikace příliš robustní a vývoj takovýchto aplikací se nemusí vyplatit a to jak z časového tak z finančního hlediska. Nadruhou stranu, lze napsat komplexní a složitou RIA aplikaci, stavějící na základech technologií, poskytovaných internetovým prohlížečem. Za touto volbou musíme hledat kvalitní analýzu aplikace a opravdu dobrý aplikační návrh.

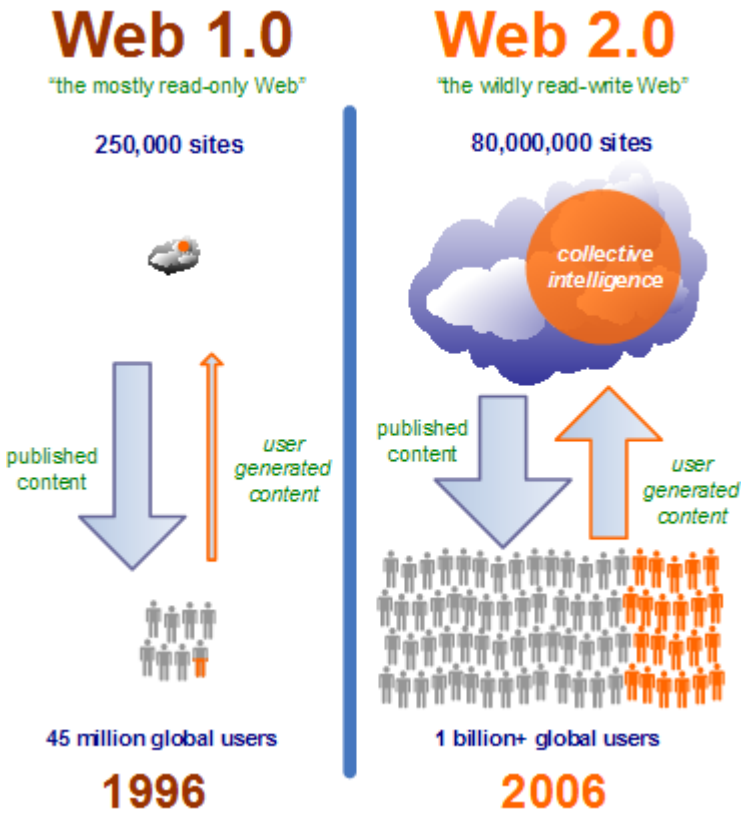
3.4 Web 2.0 a RIA

Ačkoli uživateli Internetu může připadat vymezení pojmu Web 2.0 velmi jednoduché, není tomu tak. Web 2.0 není technologií a není to ani přesně definovaný termín; lze říci, že jeho definice nemá ustálený charakter. Samotný pojem může vyvolávat představu, že jde o novou verzi WWW, avšak tento název nevychází z žádného vývoje technických specifikací, jde spíše o změnu přístupu k současným internetovým technologiím, zejména webu. Vznikl z názvu konference O'Reilly Media Web 2.0, pořádané v roce 2004, kde měl odkazovat na návrat internetového podnikání po krachu pádu internetových společností v roce 2001 [5].

Tvůrcem tohoto pojmu je Tim O'Reilly, který se pokusil definovat Web 2.0 takto: *Web 2.0 je revoluce podnikání v počítačovém průmyslu způsobená přesunem k chápání webu jako platformy a pokus porozumět pravidlům vedoucím k úspěchu na této nové platformě. Klíčovým mezi těmito pravidly je toto: tvořte aplikace, které budou díky síťovému efektu s přibývajícím počtem uživatelů stále lepší. (Což jsem jinde nazval „zapřažením kolektivní inteligence".) [4].*

Tim Berners Lee, tvůrce WWW popsal pojem Web 2.0 takto: „*Nikdo ve skutečnosti neví, co znamená .... Pokud pro vás Web 2.0 znamená blogy a wiki, tak to jsou lidé lidem. Což byla již původní myšlenka WWW." [6]*. Uvedený citát naráží na skutečnost, že veškeré technologie, které Web 2.0 využívá, byly dostupné již od prvopočátků WWW a nepřináší tak žádné technologické vylepšení a socializaci internetových aplikací vytvářejí především samotní uživatelé, nikoli technologie.

Web 2.0 tedy znamená především změnu přístupu k využívání internetových technologií, snaží se o zpřístupnění a sdílení informací mezi uživateli a poskytuje jim nástroje, pomocí kterých se mohou spolupodílet na tvorbě obsahu a vytvářet tak sociální sítě.



Obrázek 4 - znázornění přínosu Webu 2.0 [7]

Vztah Webu 2.0 a RIA je těsný, protože RIA technologie umožňují vytvářet aplikace pro tvorbu obsahu, sdílení informací mezi uživateli a poskytují tak nástroje pro vytváření sociálních sítí. Může docházet k mylnému dojmu, že RIA aplikace jsou Webem 2.0, popř. vznikly na základě potřeb Webu 2.0. Není tomu tak, stejně jako technologie WWW, tak i technologie RIA, zde byla dříve, než byl pojem Web 2.0 vymezen a definován. RIA jsou prostředkem pro vytváření aplikací, které lze nazvat aplikacemi Webu 2.0.

4 Platformy pro vývoj RIA aplikací

Jak již bylo napsáno dříve, RIA aplikace jsou aplikace bežící v běžném internetovém prohlížeči. Díky této vlastnosti vzniklo kromě zásuvných proprietárních modulů také množství frameworků, které využívají pro svůj běh právě technologií běžně dostupných v internetovém prohlížeči, a proto jejich počet přesahuje nabídku frameworků, které využívají zásuvných prohlížečových modulů.

Přehled platforem pro vývoj RIA aplikací:

- využívající proprietálních zásuvných modulů
    - Adobe Flash Platform,
    - Microsoft Silverlight,
    - OpenLaszlo,
    - Curl,
- využívající technologií internetového prohlížeče
    - ExtJS,
    - Bindows,
    - Google Web Toolkit,
    - JavaFX,
    - OpenLaszlo,
    - a další …

4.1 *Adobe Flash Platform*

Je možné se setkat s tvrzením, že je v současné době firma Adobe diky své produktové řadě největším hráčem na poli RIA aplikací. Takové tvrzení není daleko od pravdy a to i přes fakt, že pro běh aplikací psaných v Adobe Flash potřebuje uživatel přítomnost zásuvného prohlížečového modulu.

Tato dominance je důsledkem mnoha faktorů:

- *dominantní zastoupení prohlížečového modulu Adobe Flash v klientských systémech* – Adobe Flash je zastoupen na více než 5 miliardách počítačů, což představuje přibližně 95% počítačů, které jsou připojeny k Internetu.
- *dlouholeté zkušeností s vývojem RIA aplikací* – i nyní Adobe těží ze zkušeností a předpovědí, které byly nastoleny firmou Macromedia.
- *široká komunita vývojářů* – v dnešní době sociálních sítí je komunita vývojářů velice ceněným artiklem a firma Adobe si je tohoto faktu vědoma, proto vybudovala po celém světě síť lidí zvaných „Platform Evangelist“, kteří tvoří most mezi firmou Adobe a vývojáři využívající jejich produktů.
- *uvolnění zdrojových kódů pod licencí OpenSource* – firma Adobe uvolnila zdrojové kódy frameworku Flex a serveru BlazeDS pod licencí OpenSource. Tento krok umožnil nezávislým vývojářům zasahovat do zdrojových kódů a zlepšovat tak funkčnost celého frameworku,
- *nástroje pro vývoj jsou dostupné zdarma* – základní nástroje, pomocí kterých lze začít vyvíjet RIA aplikace, jsou poskytovány zdarma

Přestože je pojem Adobe Flash Platform relativně mladý, zahrnuje celou řadu již dávno známých produktů. V bakalářské práci však budou uvažovány pouze produkty, které přímo souvisejí s tvorbou RIA aplikací. Pro orientační přehled bude uveden seznam celé produktové řady s krátkým popiskem.

Přehled Adobe Flash Platofrm:

- *Adobe Flash* – prostředí pro tvorbu vektorových animací, tvorbu skinů pro RIA aplikace, zásuvných modulů, atp.
- *Adobe Flash Player* – zásuvný prohlížečový modul potřebný pro běh aplikací psaných pro Adobe Flash, v současné době zastoupen na všech majoritních operačních systémech.
- *Adobe Flex* – framework pro tvorbu RIA aplikací, uvolněn pod OpenSource licencí.
- *Adobe AIR* – prostředí, které umožňuje spuštění Flash aplikace jako plnohodnotne desktopová aplikace.
- *Adobe Flash Catalyst* – nedávno představený nástroj umožňující skinování Flex/Air aplikací.
- *Adobe Creative Suite* – soubor produktů pro práci s bitmapovou a vektorovou grafikou.
- *Serverová řešení* – výše bylo uvedeno, že pokud se hovoří o RIA aplikacích tak je myšlen uživatelský front-end. Avšak pro funkčnost aplikace je nutné zajistit kanály pro získávání a přenos dat do klientského rozhraní a zpět. K tomuto účelu nabízí Adobe Flash Platform trojici serverových řešení, jsou to:
    - *Adobe ColdFusion* – plnohodnotný webový server, který je schopen poskytovat uživatelům i obyčejné internetové stránky.
    - *Adobe LiveCycle* – serverové řešení určené primárně pro RIA aplikace, umožňuje jednoduchou definici datových kanálů, přímé napojení do SQL databází, přímé generování PDF dokumentů a mnoho dalších výhod.
    - *Adobe BlazeDS* – OpenSource varianta výše uvedeného produktu Adobe LiveCycle, která však nedisponuje takovým portfoliem možností.

4.1.1 Hodnocení

Klady	Zápory
- vysoké procento rozšíření na uživatelských stanicích - propojení s ostatními produkty Adobe Inc. - základní vývojové nástroje jsou poskytovány zdarma - vysoký výkon aplikací - vývojové prostředí Flex Builder, umožňující komplexní správu projektů a editor pro návrh uživatelského rozhraní - možnost přenosu aplikace z internetového prohlížeče na desktop (Adobe AIR)	- spoléhá na zásuvný prohlížečový modul - možné problémy s přístupností aplikací

4.2 Microsoft Silverlight

Jedná se o velice mladou technologii firmy Microsoft, jež byla představena v roce 2007 ve verzi 1.0. V současné době je Microsoft Silverlight dostupný ve verzi 2.0.

Technologie Adobe Flex a Microsoft Silverlight mají podobné základy:

- pro běh je nutný zásuvný prohlížečový modul, velikost v řádech megabajtů,
- uživatelské rozhraní je definováno ve značkovacím jazyku, v případě Microsoft Silverlight jde o jazyk XAML, který opět vychází z jazyka XML. Na rozdíl od Adobe Flex není XAML kompilován,
- pro programování funkčnosti slouží objektove orientovaný jazyk,

- zajištěna multiplatformnost v rámci systému Windows, Mac OS a pro nejrozšířenější prohlížeče jako je Internet Explorer, Firefox a Safari. Verzi pro systém Linux zajišťuje běhové prostředí Moonlight společnosti Novell.

Ačkoli je Microsoft Silverlight stále ještě nováčkem na poli RIA aplikací, nedají se mu upřít nesporné výhody při využití vývojových nástrojů firmy Microsoft. Běhové prostředí Microsoft Silverlight totiž využívá oblíbeného prostředí .NET, což přináší nesporné výhody [10]:

- pro vývoj máme k dispozici širokou škálu programovacíh jazyků. Kromě oblíbeného a dobře známého jazyka C# lze využít jazyků jako je Visual Basic, PHP, Ruby, Python.
- využití OOP konceptů na vyšší úrovni
- možnost vývoje serverové i klientské části se znalostí jednotného programovacího modelu a jazyka
- využití vyspělých technologií jako je LINQ či kompoziční model WPF
- početná komunita .NET vývojářů

4.2.1 Hodnocení

Klady	Zápory
- využívá silné pozice firmy Microsoft - propojení se stávajícími vyvojářskými nástroji, které se používají při vývoji pro .NET - široká škála programovacích jazyků	- spoléhá na zásuvný prohlížečový modul - přes dominantní zastoupení firmy Microsoft je zastoupen na malém počtu uživatelských stanic

4.3 OpenLaszlo

Jedná se o OpenSource platformu pro vývoj RIA aplikací. Pro zápis rozhraní a funkcí je použit jazyk LZX. O provoz a kompilaci se stará OpenLazslo Server, což je Java servlet, který umožňuje kompilovat výslednou aplikaci buď do DHTML, nebo binárního SWF souboru pro Adobe Flash Player. Díky této vlastnosti lze používat aplikace psané v OpenLazslo i bez nutnosti instalace dodatečného prohlížečového modulu [17].

4.3.1 Hodnocení

Klady	Zápory
- možnosti kompilace aplikace do dvou formátů, SWF a DHTML - možnost využití obou technologických směrů	- chybí vývojové prostředí a nástroje pro návrh uživatelského prostředí - slabší pozice na trhu

4.4 ExtJS

Tento framework, který je uvolněn pod GNU GPL licencí, patří do kategorie frameworků, které pro svůj běh nevyžadují přítomnost dodatečného prohlížečového modulu. Již z názvu frameworku lze odvodit, že je postaven na skriptovacím jazyce JavaScript. Framework je psán tak, aby byl vývojář oproštěn od implementačních problémů jednotlivých prohlížečů. Framework nabízí základní kolekce formulářových prvků, dobrou skinovatelnost všech komponent a obsahuje zapouzdření funkcí Adobe AIR [18].

4.4.1 Hodnocení

Klady	Zápory
- není závislý na zásuvném prohlížečovém modulu - využívá technologií, které poskytují internetové prohlížeče - poskytován zdarma - k vývoji nepotřebuje vlastní vývojový nástroj.	- silná závislost na JavaScriptu - nižší výkon - možnost využití v méně složitých aplikacích

4.5 Bindows

Stejně tak jako předchozí framework i Bindows nevyžaduje ke svému běhu dodatečný prohlížečový modul. Taktéž je založen na skriptovacím jazyce JavaScript, avšak k popisu UI aplikace je použit jazyk XML, zvaný ADF (Application Description File) [19].

4.5.1 Hodnocení

Klady	Zápory
- nevyžaduje vlastní prohlížečový modul - vlastní jazyk pro popis vzhledu aplikace - odpadají problémy s přístupností	- nízké zastoupení - silná závislost na JavaScriptu - nízký výkon u složitých aplikací - není poskytován zdarma

4.6 Curl

Jedná se o plně objektově orientovaný programovací jazyk určený pro RIA aplikace. Kombinuje funkčnost značkovacího jazyka (např. HTML) a skriptovacího jazyka (např. JavaScript) v jednom jednotném frameworku. Curl má vlastní značkovací jazyk, podobný HTML. Pro zápis aplikací není nutné oddělení definice informací, stylu a chování, vše lze zapsat do jednoho souboru. Pro běh aplikace je nutný zásuvný prohlížečový modul, který se nazývá Curl RTE. V současné době jsou podporované platformy Microsoft Windows, Linux a Mac OS. Taktéž je možná kompilace aplikace jako platné desktopové aplikace, podobně jako tomu je u Adobe AIR [20].

4.6.1 Hodnocení

Klady	Zápory
- vlastní vývojové nástroje - zaměřen na bezpečnost aplikací - dobrý výkon aplikací - možnost přenosu aplikací z internetového prohlížeče na desktop	- spoléhá na zásuvný prohlížečový modul - nízké zastoupení v uživatelský - výhradní zaměření na enterprise aplikace - drahé vývojové nástroje - problémy s přístupností

4.7 Google Web Toolkit

Google Web Toolkit je OpenSource framework, psaný v populárním jazyce Java, který umožňuje vývojářům vytvářet a spravovat komplexní RIA aplikace. Výsledné rozhraní je realizováno pomocí HTML a JavaScriptu. Pro komunikaci se serverem je primárně použita technologie Ajax [21].

4.7.1 Hodnocení

Klady	Zápory
- maximální využití technologií, které poskytují současné internetové prohlížeče. - odpadají problémy s přístupností aplikací - připraven pro vývoj enterprise aplikací	- chybí nástroje pro návrh uživatelského rozhraní - úzké propojení na funkce JavaScriptu

5 Architektura RIA aplikací

5.1 Adobe Flash Platform

Pro vývoj RIA aplikací pod technologií Adobe Flash je určen framework Adobe Flex, který poskytuje bohatou kolekci formulářových a UI prvků. Samotný kód aplikace lze psát buď v libovolném textovém editoru, nebo je možné využít komerční varianty, kterou je Adobe Flex Builder (dále Flex Builder) jež je určen pro vývoj Adobe Flex aplikací. Pro spuštění aplikace je nutné provést kompilaci zdrojových souborů. Pokud je použít editor Flex Builder, je kompilace záležitostí několika kliknutí. Je-li k vývoji použito jiného editoru, lze kompilaci provést pomocí příkazové řádky. Samotná aplikace může být zkompilována do dvou různých formátů, a to:

- SWF – formát souboru, který lze spustit v zásuvném modulu Adobe Flash Player,
- AIR – v tomto případě nelze mluvit o formátu souboru, ale spíše o balíčku, který je určen pro instalaci na systému, který umožňuje instalaci a spouštění AIR aplikací. Tento balíček obsahuje kromě souboru SWF také soubor s popisem vlastností aplikace.

5.1.1 Adobe Flex

Framework, který byl představen již v roce 2004, tehdy ještě firmou Macromedia, prodělal za svých pět let vývoje mnoho změn. V současné době je dostupný ve verzi 3.2. Samotný framework je distribuován pomocí balíčku zvaného Flex SDK. Tento balíček obsahuje kompletní zdrojové soubory frameworku Flex a soubory potřebné pro kompilaci aplikací.

Adobe Flex využívá jako programovacího jazyku jazyk ActionScript, který je hlavním programovacím jazykem pro aplikace spouštěné v zásuvném prohlížečovém modulu Adobe Flash Player. Jazyk ActionScript, vychází

z ECMAScript standardu, je tedy velice podobný jazyku JavaScript, který vychází ze stejných základů.

Základní nabídka komponent a aplikační logika frameworku je psána v programovacím jazyce ActionScript, což umožňuje vývojářům využít všech výhod objektově orientovaného přístupu při rozšiřování funkčností základních komponent či úpravou samotné aplikační logiky frameworku.

Pro zápis struktury a vzhledu aplikace se zpravidla používá jazyk MXML. Jde o značkovací jazyk, který vychází z definice XML a striktně vyžaduje validitu dokumentu. Funkčnost aplikace, interní logika a komunikace s externími datovými zdroji je zapisována pomocí jazyku ActionScript. Vzhled aplikace lze ovlivnit pomocí zjednodušené implementace CSS.

5.1.2 Zápis kódu

Základním stavebním kamenem Adobe Flex aplikací je tag <*Application*>, popř. <*WindowedApplication*> v případě AIR aplikace. Tento tag vytváří kontejner pro celý zbytek aplikace. Do tohoto kontejneru jsou umísťovány ostatní komponenty, které se mají v aplikaci zobrazit. Do kontejneru lze přímo vpisovat instrukce v jazyce ActionScript. Adobe Flex podporuje vytváření vlastních komponent, které mohou upravovat funkci již vestavěných komponent. Všechny dostupné MXML komponenty lze vytvářet a měnit i pomocí jazyka ActionScript.

Pro znázornění struktury jazyka MXML a ActionScript, bude prezentována jednoduchá aplikace, která po stisknutí tlačítka „*Příklad*“ zobrazí okno s textem „*Hello world*“. Příklad bude prezentován ve dvou variantách. V první variantě bude příklad zapsán běžným způsobem, tj. vzhled aplikace bude popsán v jazyce MXML a logika aplikace bude zapsána v jazyce ActionScript. Druhá varianta příkladu znázorní možnost zapsat celou aplikaci jako ActionScript, bez použití MXML

definice vzhledu. Pro demonstraci možností změny vzhledu pomocí CSS bude text tlačítka změněn z výchozí černé barvy na červenou.

Příklad zápisu MXML dokumentu s vloženou aplikační logikou pomocí jazyka ActionScript:

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" width="300"
height="150" verticalAlign="middle" horizontalAlign="center">

  <mx:Script>
    <![CDATA[
       import mx.controls.Alert;

       private function sampleButton_ClickHandler():void
       {
         Alert.show( "Hello world", "Hello world alert!" );
       }
    ]]>
  </mx:Script>

  <mx:Button label="Příklad" id="sampleButton"
    click="sampleButton_ClickHandler()"
    fontWeight="bold" color="0xFF0000" />
</mx:Application>
```

V této variantě příkladu je pomocí MXML tagů definován vzhled aplikace a aplikační logika je implementována pomocí jazyka ActionScript. Na úvod dokumentu je uvedena XML hlavička s informací o verzi a kódování dokumentu, protože jak již bylo zmíněno, jazyk MXML vychází z definice jazyka XML a ten tuto definici předepisuje. Jako další následuje tag *<mx:Application>*, který vytváří hlavní kontejner aplikace, do kterého jsou vkládány další komponenty. V tomto tagu je předepsán výchozí vzhled aplikace, v tomto případě je nastavena šířka aplikace na 300px a výška na 150px. Výchozí zarovnání vložených komponent je nastaveno na střed, a to jak v horizontálním směru (parametr *horizontalAlign*) tak i ve vodorovném směru (parametr *verticalAlign*). Za úvodní definicí aplikačního kontejneru následuje tag, umožňující vložení ActionScript kódu, v tomto případě je zde uveden zápis funkce *sampleButton_ClickHandler*, která obslouží událost po kliknutí na tlačítko „Příklad“. V popisu funkce je využita komponenta Alert, která poskytuje statickou metodu *show()* jež zobrazí varovné okno, v našem případě bude okno obsahovat text „*Hello Word*“. Posledním prvkem dokumentu je MXML definice tlačítka. V této definici je předepsán text tlačítka (parametr label)

a nastaveno obsloužení události „*click*", která nastane po kliknutí. Dále je změněn výchozí vzhled tlačítka s požadavkem na změnu tučnosti (parametr *fontWeight*) a barvy (parametr *color*).

Příklad zápisu vzhledu aplikace bez použití MXML:

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" width="300"
height="150" verticalAlign="middle" horizontalAlign="center"
creationComplete="init()">
  <mx:Script>
    <![CDATA[
      import mx.controls.Button;
      import mx.controls.Alert;

      private function init():void
      {
        var btn:Button = new Button();
        btn.label = "Příklad";
        btn.id = "sampleButton";
        btn.addEventListener( MouseEvent.CLICK,
                              sampleButton_clickHandler );
        btn.setStyle("fontWeight", "bold" );
        btn.setStyle("color", "0xFF0000");

        this.addChild( btn );
      }

      private function sampleButton_ClickHandler( event:MouseEvent ):void
      {
        Alert.show( "Hello world", "Hello world alert!" );
      }
  ]]>
  </mx:Script>
</mx:Application>
```

V druhém případě je pomocí jazyka MXML definován pouze základní kontejner aplikace. Oproti první variantě se objevil v definici tagu <*Application*> nový parametr *creationComplete*, který zajistí spuštění funkce *init()* ve chvíli, kdy byla dokončena inicializace a základní rámec aplikace byl zobrazen v prohlížeči. Funkce *init()*, provede vytvoření tlačítka „Příklad" pomocí jazyku ActionScript. Nejprve definována proměnná *btn*, která ponese instanci požadovaného tlačítka. Po vytvoření objektu *Button*, je nutné nastavit požadované vlastnosti jako je popisek tlačítka (parametr *btn.label*), ID prvku (parametr *btn.id*) a obsluha události, která nastane při kliknutí na tlačítko. Tuto událost je obsloužena vložením „listeneru události" (*btn.AddEventListener(...)*). Tento listener bude mít za úkol odposlech událostí, které

na tlačítku nastanou a při události *Click*, vyvolané kliknutím na tlačítko, spustí funkci *sampleButton_clickHandler*. Nakonec je nastavena tučnost textu tlačítka (parametr *btn.setStyle(„fontWeight“, „bold“)*) a barva (parametr *btn.setStyle(„color“, „0xFF0000“)*). Tímto jsou nadefinovány základní parametry, obsluha událostí a vzhled tlačítka. Jako poslední krok je vložení tlačítka do aplikačního kontejneru pomocí metody *this.addChild (btn)*.

V obou výše uvedených případech byla vytvořena stejná aplikace, kterou znázorňuje Obrázek 5.

Obrázek 5 - ukázka výstupu vzorové aplikace psané v Adobe Flex

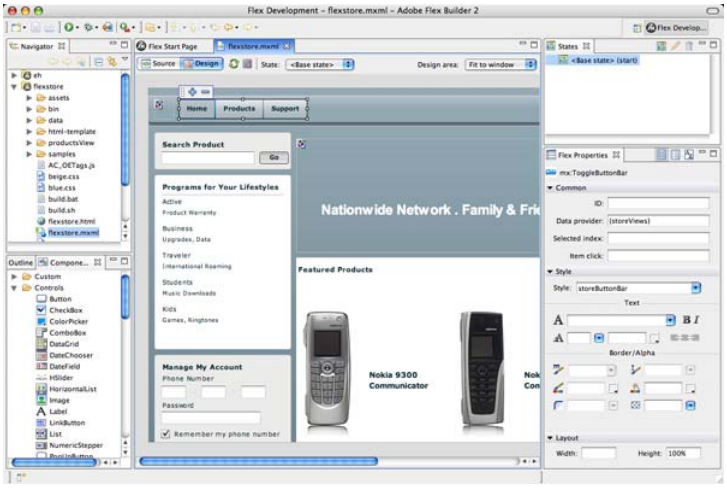
5.1.3 Adobe Flex Builder

Jde o komerční vývojové prostředí pro vývoj aplikací v Adobe Flex. Pro běh využívá obecného prostředí Eclipse, které je vyvíjeno neziskovou společností Eclipse Foundation, jejímiž členy jsou firmy jako IBM, Oracle, Sun, Zend, SAP, Iona, Nokia [22].

Adobe Flex Builder poskytuje editor ActionScript kódu s podporou vysvícení syntaxe jazyka. Psaný kód je průběžně kontrolován na správnost syntaxe a v případě nalezení chyby je vývojář okamžitě upozorněn.

Vývojář má také k dispozici vizuální editor aplikace. Tento editor umožňuje jednoduchý návrh uživatelského rozhraní, které lze skládat jak ze základních komponent, které poskytuje framework, tak komponent vlastních. Veškeré změny, provedené ve vizuálním návrhu se ihned promítají do kódu aplikace. Mezi vizuálním návrhem aplikace a kódem se lze libovolně přepínat.

Vývojové prostředí obsahuje nástroje pro odhalování chyb (debugger), k dispozici je zde také Profiler, který umožňuje odhalit paměťové chyby vznikající při běhu aplikace.



Obrázek 6 - rozhraní aplikace Adobe Flex Builder

5.1.4 Adobe AIR

Společnost Adobe Inc. představila v roce 2007 běhové prostředí AIR, které poskytlo možnost přenést RIA z běžného internetového prohlížeče na pracovní stanice jako plnohodnotnou aplikaci. AIR (*Adobe Integrated Runtime)* je multiplaformní běhové prostředí určené pro vývoj RIA, postavených na technologiích Adobe Flash, Adobe Flex, HTML nebo Ajax, které mohou být spuštěny jako plnohodnotné desktopové aplikace [14].

Oproti aplikacím, běžícím v rámci internetového prohlížeče, je nutné AIR aplikace instalovat do místního uživatelského systému. Instalací získá aplikace možnost přístupu k místním diskům a službám systému.

5.1.5 Serverová řešení

Aby mohla být RIA aplikace opravdu užitečná a plnila svou funkci, je potřeba zajistit propojení se serverovými skripty. V této oblasti je Adobe Flex velmi otevřený. Poskytuje širokou škálu komponent, které jsou určeny komunikaci a manipulaci s daty. Základním komunikačním protokolem jsou XML, SOAP a AMF. Jako server zpracovávající požadavky lze použít libovolný server, který je schopen přijmout požadavky ve zvoleném komunikačním protokolu a odpovědět. Vzhledem k otevřenosti všech zmíněných protokolů lze nalézt velké množství implementací a to jak zdarma dostupných (např. AMFPhP, ZendAMF) tak i komerčních (např. Adobe Live Cycle Data Services, WebSphere).

6 Závěr

Internet – pojem, který je dnes provázán prakticky s každým odvětvím lidské činnosti, prošel od počátku svého vytvoření (před přibližně 40 lety) velice rychlým rozvojem. Zásadním vývojem prošli i jeho uživatelé, jejichž počet rapidně narostl z počátečních stovek a tisíců do dnešních stovek milionů. Úměrně s dobou využívání služeb Internetu se mění chápání, struktura a způsob využívání (kladené nároky) jeho funkcí a nástrojů. Tlak uživatelů s požadavkem na vyšší interaktivitu a intuitivnost webových aplikací dal vzniknout novému odvětví a pojmu – RIA.

Objasnění pojmu RIA a představení hlavních platforem a jejich architektury pro tvorbu zmíněných aplikací bylo cílem bakalářské práce. Pojmem RIA jsou označovány aplikace s vysokou mírou interaktivity uživatele a zahrnují jak administrační nástroje, tak i internetové portály a nově i desktopové aplikace. Zejména v oblasti administračních aplikací nastal velký rozvoj. Firmy si uvědomují výhody těchto aplikací, plynoucí z faktu, že samotný internetový prohlížeč, který je v dnešní době součástí každodenního počítače, může fungovat jako základ pro aplikace, ve kterých lze například spravovat finance firmy, evidovat skladové zásoby či monitorovat pohyby na burze.

Cílem první části práce bylo představení a definování samotného pojmu RIA a také byl věnován prostor pro uvedení okolností vzniku tohoto pojmu. Byly představeny jednotlivé organizace a technologie, které ke vzniku RIA aplikací přispěly. Dále byl proveden rozbor základních rysů a popsána charakteristika RIA s uvedením dvou technologických směrů či přístupů, na základě kterých lze aplikace vytvářet. Detailně byl rozebrán přístup, využívající technologie AJAX. Rozdíl oproti běžným internetovým aplikacím byl demonstrován na jednoduchém příkladu. Taktéž byly vyjmenovány výhody a nevýhody RIA technologií jako celku a byly porovnány i jednotlivé technologické směry. Byla provedena analýza a rozbor volby vhodného technologického směru pro RIA aplikace. V poslední kapitole první části bylo provedeno zamyšlení nad vztahem RIA aplikací a pojmu Web 2.0.

S příchodem pojmu Web 2.0 nabyly RIA aplikace obliby i u internetových portálů, pro které byly technologie RIA odpovědí na změny v chápání a nakládání s informacemi na Internetu. Uživatelé v současné době sociálních sítí vyžadují vyšší míru interaktivity, mají potřebu sdílet informace a společně nové informace vytvářet. Na všechny tyto požadavky jsou RIA technologie odpovědí.

Ačkoliv RIA jsou v práci uváděny jako velmi mocný nástroj vývoje moderních aplikací, nelze však hovořit pouze o kladech, jež jsou s RIA aplikacemi spojeny. Zvolení špatného směru při vývoji aplikací může vést k problémům, které na první pohled nebudou tak jasné. Například mohou vzniknout problémy s přístupností aplikace. Při analýze a návrhu budoucí RIA aplikace je tedy nutné klást důraz na účel a funkci, a dle těchto kriterií volit vhodný technologický směr, popř. zvolit cestu kombinace obou technologických směrů, tj. směru, který využívá současných technologií, které poskytují prohlížeče a směru, který využívá proprietárních zásuvných modulů. Při volbě je nutné počítat i s rozdílnou ekonomickou náročností jednotlivých platforem pro vývoj RIA.

Dílčím cílem práce bylo představení platforem pro vývoj aplikací RIA. Byl uveden seznam nástrojů, s jejichž pomocí lze RIA vyvíjet. Každý nástroj byl představen a zhodnocen tabulkou, ve které byly vypsány klady a zápory jednotlivého nástroje. Některé frameworky, jako je například ExtJS, jsou poskytovány zcela zdarma. Alternativou je platforma Adobe Flex, jež je poskytována taktéž zdarma, avšak pro efektivní vývoj a využití všech možností, které tento framework poskytuje, je vhodné zakoupit vývojové prostředí Adobe Flex Builder. V nabídce jsou dále platformy, jako je například Curl či Microsoft Silverlight, které jsou zpoplatněny včetně základního frameworku.

Cílem bakalářské práce také bylo představení architektury RIA, konkrétně platformy Adobe Flex, s popisem jeho základních charakteristik. Dále byly podány informace o způsobu vývoje uvažovaných aplikací. Pro představu byl demonstrován příklad tvorby aplikace v Adobe Flex, a to ve dvou variantách: pro znázornění dvou možných přístupů tvorby aplikací v tomto frameworku. Bylo představeno vývojové prostředí Adobe Flex Builder s popisem základních vlastností tohoto nástroje.

Nakonec bylo pojednáno o nové technologii Adobe AIR, umožňující přenos RIA aplikací na klientské desktopy jako plnohodnotné aplikace. Za účelem dosažení komplexního přehledu byly podány také informace o možnostech propojení Adobe Flex s různými serverovými technologiemi.

RIA aplikace se s rozvojem Internetu, a především webových aplikací, staly dobrým obchodním artiklem. Společnosti, které na tomto trhu podnikají, nabízejí kvalitní produkty, a s využitím dostupných RIA technologií, dokáží plnit různorodá přání klientů. Na druhé straně se RIA, jakožto symbol interaktivity, přívětivosti a moderních technologických přístupů, nedokázaly ubránit zneužívání. Označení RIA mnohdy využívají produkty, které ani zdaleka nedosahují základních parametrů, jimiž se tyto aplikace charakterizují. Zneužívání komerčně úspěšných značek, které je v dnešní době populární, RIA škodí a je pouze na tvůrcích aplikací, aby uživatelům dokázali poskytnout dostatek vodítek k rozpoznání opravdových RIA od náhražek, které se tímto označením pouze chlubí, ale zdaleka nedosahují takové kvality.

Problematika RIA aplikací a technologií v blízké budoucnosti zaznamená další dynamický rozvoj, kdy jejich uplatnění bude rozšiřováno nejen na současných klientských stanicích (počítačích uživatelů), ale také na moderních mobilních zařízeních, které postupně začínají disponovat dostatečným výkonem k běhu takových aplikací. Dostupná technologie, umožňující tvorbu RIA, je pouze jednou ze součástí celého komplexu tvorby a zavádění moderních interaktivních systémů. Velmi potřebná je míra uplatnění (potřebnost) uvažovaných aplikací a jejich atraktivita z pohledu uživatelů. V bakalářské práci byla uvedena zmínka o velmi rychlém a masivním rozšíření Internetu a internetových technologií, kdy během několika málo let zaznamenaly požadavky uživatelů na podobu internetových prezentací, komunikačních sítí, sdíleného obsahu a jejich chápání rapidního rozmachu a obliby, což podporuje nasazování a rozvíjení RIA aplikací a vyzdvihuje jejich kladné stránky. Bakalářská práce poskytla základní přehled o problematice RIA a možných nástrojích jejich tvorby.

7 Slovníček pojmů

ActionScript

Objektově orientovaný programovací jazyk využitelný v aplikacích vyvíjených pro platformu Adobe Flash. Vychází ze standardizované verze jazyka ECMAScript.

ADF

Application Description File – jedná se o jazyk založený na značkovacím jazyce XML. Slouží k popisování vzhledu RIA aplikaci.

AJAX

Asynchronous JavaScript And XML – obecné označení pro technologii vývoje interaktivních webových aplikací, které mění obsah svých stránek bez nutnosti jejich celkového znovunačítání.

AMF

Action Message Format – binární formát založený na protokolu SOAP. Je primárně určen pro výměnu dat mezi aplikacemi psanými na platformě Adobe Flash a databází.

API

Application Programming Interface – rozhraní pro programování aplikací. Jedná se o soubor procedur, funkcí či tříd určité knihovny, které muže programátor využívat. API určuje, jakým způsobem se funkce knihovny mají volat ze zdrojového kódu programu.

CSS

Cascading Style Sheets – jazyk pro popis způsobu zobrazení stránek napsaných ve značkovacích jazycích. Je primárně navržen k tomu, aby umožnil oddělení obsahu dokumentu od vzhledové prezentace. Ve většině případů je používán pro popis vzhledu webových stránek psaných v HTML a XML, ale může být taktéž aplikován na jakýkoli dokument založený na standardu XML, včetně SVG a XUL.

DHTML

Dynamic HTML – Pojmenován pro kombinaci technologií používaných ke tvorbě dynamických a interaktivních webových stránek. Těmito technologiemi se obvykle myslí HTML, JavaScript, CSS, DOM.

DOM

Document Object Model – objektově orientovaná reprezentace XML nebo HTML dokumentu. DOM je API umožňující přístup či modifikaci obsahu, struktury, nebo stylu dokumentu, či jeho částí.

ECMA

Ecma International – mezinárodní soukromá nezisková organizace, založená roku 1961 s cílem standardizace informačních a komunikačních systémů v Evropě.. Dříve také známá pod názvem *European Computer Manufacturers Association*. Její název byl v roce 1994 změněn, aby zdůraznil mezinárodní zaměření organizace.

ECMAScript

skriptovací jazyk, standardizovaný asociací Ecma International ve specifikaci ECMA-262. Jazyk je především určen pro použití v internetových stránkách, kde jej většina uživatelů zná pod názvem JavaScript.

Framework

Softwarová struktura, která slouží jako podpora při programování, vývoji a organizaci jiných softwarových projektů. Cílem je převzetí typických problémů dané oblasti, čímž se usnadní vývoj tak, aby se návrháři a vývojáři mohli soustředit pouze na své zadání.

JavaScript

multiplatformní, objektově orientovaný skriptovací jazyk. Zpravidla je používán jako interpretovaný programovací jazyk pro internetové stránky. Syntaxí patří do rodiny jazyků C/C++/Java. Slovo Java bylo použito v názvu pouze z marketingových důvodů a s programovacím jazykem Java jej vedle názvu spojuje pouze podobná syntaxe. Roku 1997 byl standardizován asociací ECMA.

JVM

Sada počítačových programů a datových struktur, která využívá modul virtuálního stroje ke spuštění dalších počítačových programů a skriptů vytvořených v jazyce Java.

JSON

JavaScript Object Notation – způsob zápisu dat nezávislý na počítačové platformě, určený pro přenos dat, která mohou být organizována v polích nebo objektech.

HTML

HyperText Markup Language – jde o značkovací jazyk pro internetové stránky. Poskytuje prostředky pro popis struktury textových informací v dokumentu. Je psán ve formě značek (tagů), jež jsou uzavřeny do úhlových závorek („<“ a „>“).

LZX

Popisovací jazyk využívající pro popis rozhraní XML a JavaScriptu.

RIA

Rich Internet Application – internetové/intranetové aplikace s charakterem běžných desktopových aplikací.

SVG

Scalable Vector Graphics – značkovací jazyk a formát souboru, který popisuje dvojrozměrnou vektorovou grafiku pomocí XML.

SOAP

Simple Object Access Protocol – protokol pro výměnu zpráv založených na XML. Tvoří základní vrstvu komunikace mezi webovými službami a poskytuje prostředí pro tvorbu složitější komunikace.

skin/skinování

Pojem „skin“ je možné do českého jazyka přeložit jako kůže. Je to vlastní volitelný grafický vzhled aplikace nebo webové stránky. Skin má vliv pouze na vzhled, obsah zůstává nezměněn

UI

User Interface – rozhraní, pomocí kterého uživatel komunikuje s aplikací a aplikace s uživatelem. Toto prostředí umožňuje uživateli manipulaci s aplikací (např. zadávání příkazů) a aplikaci na tyto příkazy reagovat.

XHTML

Extensible Hypertext Markup Language - jde o značkovací jazyk, vycházející z HTML. Důraz je kladen na správnou (validní) syntaxi jazyka XML.

XUL

XML User Interface Language - jedná se o formát pro tvorbu multiplatformního grafického rozhraní, používaný v produktech firmy Mozilla, která je i jeho autorem. Zápis je prováděn formou XML.

XML

eXtensible Markup Language – obecný značkovací jazyk, který byl vyvinut a standardizován konsorciem W3C. Umožňuje snadné vytváření konkrétních značkovacích jazyků pro různé účely a je vhodný pro komunikaci a výměnu dat mezi aplikacemi. Jazyk je určen pro popis struktury dokumentu z hlediska věcného obsahu jednotlivých částí, nezabývá se sám o sobě vzhledem dokumentu nebo jeho částí.

.NET

dot NET – název zastřešující soubor technologií v softwarových produktech, které tvoří celou platformu. Základní komponentou je Microsoft .NET Framework. Platforma .NET nepředepisuje použití žádného programovacího jazyka. Bez ohledu na to, v čem byla aplikace původně napsána, se vždy přeloží do mezijazyka CIL (*Common Intermediate Language*). Nejpoužívanějšími jazyky pro vývoj .NET aplikací jsou C#, Visual Basic, Delphi, ale také PHP, Python.

8 Seznam literatury

[1] ROSS Shannon, *The History of the Net* [online] [cit. 2009-4-26].
Dostupné z: http://www.yourhtmlsource.com/starthere/historyofthenet.html

[2] History of the Domain Name System - A.K.A. DNS [online] [cit. 2009-4-26].
Dostupné z: http://www.lagunainternet.com/techsupport/history_of_dns.htm

[3] GRIBBLE Cheryl, *History of the Web Beginning at CERN* [online] [cit. 2009-4-26].
Dostupné z: http://www.hitmill.com/internet/web_history.html

[4] O'REILLY Tim, *Web 2.0 Compact Definitiv: Trying Again* [online] 2006-10-12 [cit. 26.4.2006].
Dostupné z: http://radar.oreilly.com/archives/2006/12/web-20-compact.html

[5] LOGANBILL Scott, *The End of Web 2.0, Beginning of Web Infinity* [online] [cit. 2009-4-26].
Dostupné z:
http://www.webmonkey.com/blog/The_End_of_Web_2DOT0__Beginning_of_Web_Infinity

[6] developerWorks, *Tim Berners-Lee* [online] 2006-9-22 [cit. 2009-4-26].
Dostupné z:
http://www.ibm.com/developerworks/podcast/dwi/cm-int082206txt.html

[7] Dion Hinchcliffe's Web 2.0 Blog [online] 2006-9-4 [cit. 2009-4-26].
Dostupné z:
http://web2.wsj2.com/all_we_got_was_web_10_when_tim_bernerslee_actually_gave_us_w.htm

[8] PICHLÍK Roman, *Rich Internet Application* [online] 2005-6-14 [cit. 2009-4-26].
Dostupné z: http://interval.cz/clanky/rich-internet-application/

[9] NORÅS Anders, *The Past, Present and Future of Rich Internet Applications* [online] [cit. 2009-4-26].
Dostupné z:
http://www4.java.no/javazone/2005/presentasjoner/AndersNordas/Anders_Noras-Rich_Internet_Applications-Past_Present_and_Future.pdf

[10] BOREK Borek, *Rich Internet Applications v roce 2008* [online] 2008-4-25 [cit. 2009-4-26].
Dostupné z: http://interval.cz/clanky/rich-internet-applications-v-roce-2008/

[11] FundingUniverse [online] [cit. 2009-4-26].
Dostupné z:
http://www.fundinguniverse.com/company-histories/Macromedia-Inc-Company-History.html

[12] Microsoft Silverlight - Overview [online] [cit. 2009-4-26].
Dostupné z: http://www.microsoft.com/silverlight/overview/default.aspx

[13] TARIQ Ahmed, *What Is Adobe/Macromedia Flex?* [online] [cit. 2009-4-26].
Dostupné z: http://www.cflex.net/about_adobe_flex.cfm

[14] HARLAND Gil, *What Is Adobe AIR?* [online] [cit. 2009-4-26].
Dostupné z: http://www.webglossary.co.uk/article-what-is-adobe-air.asp

[15] Adobe Flash Player [online] [cit. 2009-4-26].
Dostupné z: http://www.adobe.com/products/flashplayer/

[16] ALLAIRE Jeremy, *Macromedia Flash MX—A next-generation rich klient* [online] [cit. 2009-4-26].
Dostupné z:
http://download.macromedia.com/pub/flash/whitepapers/richclient.pdf

[17] OpenLazslo [online] [cit. 2009-4-26].
Dostupné z: http://www.openlaszlo.org/taxonomy/term/14

[18] ExtJS - Documentation [online] [cit. 2009-4-26].
Dostupné z: http://extjs.com/learn/Ext_FAQ

[19] Bindows Features [online] [cit. 2009-4-26].
Dostupné z: http://www.bindows.net/features.html

[20] Introducing Curl [online] [cit. 2009-4-26].
Dostupné z: http://www.curl.com/products/feature/

[21] PICHLÍK Roman, *Google Web Toolkit* [online] [cit. 2009-4-26].
Dostupné z: http://interval.cz/clanky/google-web-toolkit/

[22] About the Eclipse Foundation [online] [cit. 2009-4-26].
Dostupné z: http://www.eclipse.org/org/

[23] Adobe AIR [online] [cit. 2009-4-26].
Dostupné z: http://www.adobe.com/products/air/