

Vysoká škola ekonomická v Praze

Fakulta informatiky a statistiky

Katedra systémové analýzy

Student: **Pavel Toušek**

Vedoucí bakalářské práce: **Ing. Libor Krsek**

TÉMA BAKALÁŘSKÉ PRÁCE

**Srovnání uživatelských rozhraní
webových a desktopových aplikací
a otázka jejich konvergence**

ROK: 2009

Prohlášení

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně a že jsem uvedl všechny použité prameny a literaturu, ze kterých jsem čerpal.

V Praze dne 30. 6. 2009

.....

podpis

Poděkování

Tímto bych rád poděkoval panu Ing. Liboru Krskovi především za možnost pracovat na tomto tématu, za jeho pomoc a cenné rady v začátcích, za trpělivost a v neposlední řadě i za vstřícnost, se kterou do celé spolupráce se mnou vstupoval.

Abstrakt

Práce srovnává uživatelská rozhraní webových a desktopových aplikací, hledá příčiny jejich odlišnosti z teoretického pohledu a ověřuje jejich platnost v praxi. Praktická část zahrnuje vlastní výzkum metodou heuristické analýzy pro porovnání použitelnosti rozhraní vybraných dvojic aplikací. Poslední část na základě dosavadního vývoje vztahu těchto dvou rozhraní stanovuje odhad, jak daleko povede jejich sbližování.

Abstract

This paper compares and contrasts user interfaces of web-based and desktop applications. It analyzes reasons of their differences theoretically and then verifies the resulting hypotheses using real-world examples. Usability of chosen pairs of applications is compared using methods of heuristic evaluation. In last part, the paper estimates the future development of the interfaces convergence.

Obsah

Obsah.....	6
1 Úvod	8
2 Vymezení základních pojmů	11
2.1 Uživatelské rozhraní	11
2.2 Desktopová aplikace	11
2.3 Webová aplikace	12
2.4 Použitelnost	13
3 Rozdíly mezi uživatelskými rozhraními v obecných prvcích.....	15
3.1 Konvence desktopových aplikací.....	15
3.2 Situace webových aplikací	16
3.2.1 Webová aplikace a webová prezentace	16
3.3 Hranice mezi rozhraním webové aplikace a rozhraním webového prohlížeče.....	17
3.3.1 Specifické aspekty práce v prohlížeči.....	19
3.4 Shrnutí původu obecných rozdílů	22
4 Srovnání UI vybraných aplikací	23
4.1 Postup	24
4.2 Textové procesory	29
4.2.1 Představení.....	30
4.2.2 Přímé srovnání	34
4.2.3 Heuristická analýza použitelnosti	35
4.2.4 Shrnutí	36
4.3 Komunikační klienti	37

4.3.1	Představení.....	38
4.3.2	Přímé srovnání	41
4.3.3	Heuristická analýza použitelnosti	42
4.3.4	Shrnutí	43
5	Konvergence	44
5.1	Příklady vývoje aplikací v čase	44
5.2	Posun významu webového prohlížeče.....	49
5.3	Mezistupeň mezi webovou a desktopovou aplikací.....	51
5.3.1	Site-specific browser	51
5.3.2	Adobe AIR.....	53
5.4	Predikce budoucího vývoje.....	55
5.4.1	Teoretické předpoklady.....	56
5.4.2	Technologické předpoklady	57
6	Závěr	59
	Zdroje.....	61
	Seznam aplikací.....	63
	Seznam obrázků.....	64
	Seznam příloh	65
	Přílohy	66

1 Úvod

World Wide Web byl od počátku své existence místem dynamického vývoje. Stejně tak od jeho počátků se datuje koexistence prvních webových aplikací jako alternativ ke klasickým desktopovým. Adaptace nových technologií, prudké zvyšování konektivity a masivní nárůst zájmu uživatelů společně vedly k takovému rozmachu webových aplikací, že se stávají skutečnými konkurenty těm desktopovým. Uživatelské rozhraní webových aplikací má kořeny ve stylu hypertextových dokumentů, zatímco běžné dnešní desktopové aplikace nabízejí uživatelské rozhraní vycházející z modelů relativně stabilních od počátků éry grafických uživatelských rozhraní.

Tato práce se snaží odpovědět na otázku, jaké jsou rozdíly mezi uživatelskými rozhraními webových a desktopových aplikací a čím jsou způsobeny. Dále si klade za cíl na konkrétních případech zjistit, jaký mají tyto rozdíly dopad na použitelnost. Bude zjišťovat, do jaké míry se uživatelská rozhraní webových a desktopových aplikací vzájemně ovlivňují a nakonec se pokusí na základě dosavadního vývoje a současné situace predikovat budoucí vývoj tohoto jejich vztahu.

Předmětem práce má být samotné uživatelské rozhraní, a tak bude co možná nejvíc upuštěno od zabíhání hlouběji do technických aspektů vývoje aplikací. v některých případech je dopad dané technologie takový, že jí bude věnován určitý prostor, ale převážně se bude práce koncentrovat na svůj předmět, tedy tu část aplikace, která je vidět nebo ji lze jiným způsobem vnímat a zpátky je skrze ni možno aplikaci řídit a ovlivňovat.

Na základě skutečnosti, že desktopové aplikace jsou zavedenější, v rámci jedné platformy konzistentnější a v současnosti rozšířenější formou a webové aplikace jsou naproti tomu novějším modelem s méně ustálenými konvencemi, procházejícím dynamičtějšími změnami, bude při srovnávání zpravidla přikládán větší důraz na specifika webových aplikací a desktopové konvence budou často kladeny jako základ pro srovnávání. Domnívám se, že tento postup odpovídá reálné struktuře zkoumaného prostředí a umožní soustředit se na poznatky pro tuto práci podstatné.

Východiskem pro toto zkoumání bude literatura zabývající se chováním uživatelů, návrhem uživatelského rozhraní a zkoumáním použitelnosti. Jde o spektrum publikací a článků od teoretických až po těsně vázané na informatickou praxi. Součástí bude také vlastní výzkum s využitím metod heuristické analýzy, v rámci kterého proběhne přímé srovnání vybraných dvojic aplikací obou platforem. Závěry z výzkumu budou následně spolu s poznatky z použitých zdrojů aplikovány při formulaci predikovaného vývoje.

Práce je členěna do čtyř hlavních kapitol. První seznámí čtenáře s pojmy, se kterými bude další text pracovat, a jejich kontextem. Druhá se zaměřuje na kořeny rozdílů mezi uživatelskými rozhraními webových a desktopových aplikací. Další kapitola prozkoumá naopak rozdíly mezi rozhraními na konkrétních příkladech dvojic aplikací a prostřednictvím vlastního výzkumu bude porovnávat použitelnost uživatelských rozhraní těchto dvojic. Poslední kapitola se zabývá sbližováním popisovaných uživatelských rozhraní, novými technologiemi naznačujícími budoucí vývoj, zkoumá směr a příčiny současného směřování vztahu uživatelských rozhraní webových a desktopových aplikací a na jejich základě stanovuje jeho odhad v blízké budoucnosti.

Vzhledem ke specifičnosti a komplikovanosti vztahu zkoumané problematiky a podnikové praxe není možné se jí v omezeném rozsahu této práce dostatečně věnovat. Podnikové informační systémy a další typy aplikačního SW pro podnikové nasazení budou ponechány stranou. Tato práce bude dále počítat s uživatelem bez omezení ze strany podnikových pravidel či správců sítě.

Práce nalezne své využití pro kohokoliv, koho zajímají tendence ve vývoji uživatelských rozhraní aplikací, její poznatky mohou aplikovat vývojáři při návrhu uživatelského rozhraní a v neposlední řadě může sloužit jako zdroj pro další bádání, protože s pomocí relevantních zdrojů pokývá dosud velmi málo zdokumentované území mezi uživatelským rozhraním klasické desktopové a webové aplikace.

2 Vymezení základních pojmů

Protože jsou v dané problematice uživateli i některými autory klíčové pojmy běžně používány velmi volně, je potřeba věnovat zvýšenou pozornost jejich konkrétnímu vymezení pro další užití v této práci. Cílem tedy nejsou univerzální a přesné definice, nýbrž základní platforma pro porozumění významu následujícího textu ve shodě s autorem.

2.1 Uživatelské rozhraní

Uživatelské rozhraní (user interface, UI) je souhrnem aspektů počítačového systému či programu, které může člověk jako uživatel vidět nebo jiným způsobem vnímat, a příkazy a mechanismy, kterými uživatel řídí činnost systému či programu a vkládá data. Grafické uživatelské rozhraní klade důraz na užití grafických prvků pro výstup a polohovacího zařízení, jako je například myš, pro ovládání. [Howe]

Protože téměř všechny moderní aplikace nabízí jako primární grafické uživatelské rozhraní, bude i tato práce zaměřena na grafická uživatelská rozhraní (GUI – graphical user interface). Nebude-li uvedeno jinak, je obecnějším pojmem *uživatelské rozhraní* myšleno *grafické uživatelské rozhraní*.

2.2 Desktopová aplikace

Aplikace je takový software, který přímo interaguje s uživatelem a slouží k řešení určitého problému či úkolu. *Desktopovou aplikací* potom pro účely této práce budeme rozumět takový aplikační software, který je spouštěn na počítači koncového uživatele nejčastěji ve formě spustitelného souboru. Kód takové aplikace je

tedy typicky přítomen v souborovém systému koncového počítače a aplikace často vyžaduje instalaci na tomto počítači.

Pojem *desktopová aplikace* není v praxi příliš používán, často je obecnější výraz *aplikace* či *aplikační software* automaticky chápán jako samostatný (stand-alone) software běžící na koncovém počítači. Vzhledem k potřebám této práce ale bude pojem *desktopová aplikace* sloužit pro jasné rozlišení od *webové aplikace*.

2.3 Webová aplikace

Webovou aplikací rozumíme takovou aplikaci, ke které uživatel přistupuje prostřednictvím webového prohlížeče a jejíž zdrojový kód je typicky umístěn na serveru v internetu či v intranetu ve formě webové stránky a většinou je také vykonáván na straně serveru a výsledek interpretován na straně klienta.

Webová aplikace tedy nevyžaduje instalaci ani přítomnost velkého objemu dat na koncovém počítači a může být dostupná z kteréhokoliv počítače v dané síti vybaveného kompatibilním¹ webovým prohlížečem.

Existují také platformy, které leží na pomezí webové a desktopové aplikace. Příkladem může být runtime Adobe AIR. Těmto případům se bude věnovat podkapitola 5.3.

¹ Webové aplikace většinou vyžadují od prohlížeče podporu skriptovacího jazyka *JavaScript* či přítomnost *Flash* pluginu. Složitější aplikace jsou testovány pro konkrétní sadu webových prohlížečů a u těch nepodporovaných se může stát, že nebude aplikace fungovat korektně nebo bude přímo zakázán přístup k této aplikaci.

2.4 Použitelnost

„Použitelnost: lehkost, s jakou se uživatel naučí ovládat, připravovat vstupy pro a interpretovat výstupy ze systému nebo jeho části.“
[IEEE 90]

Použitelnost² tedy zahrnuje širokou škálu faktorů jako rozvržení prvků, funkcionalitu, strukturu a mnohé další. Jakob Nielsen, světová kapacita v oboru použitelnosti, zdůrazňuje nutnost chápat použitelnost jako mnohorozměrnou vlastnost složenou z více komponent a popisuje pět tradičně rozlišovaných rysů použitelnosti, viz Box 2.1.

Box 2.1 Atributy použitelnosti

- *Naučitelnost*: Systém by měl být jednoduchý na naučení, aby uživatel brzy dokázal jeho prostřednictvím provádět požadovanou práci.
- *Efektivita*: Systém by měl být použitelný efektivně, aby uživateli, který se už se systémem naučil pracovat, umožnil dosáhnout vysokého stupně produktivity.
- *Zapamatovatelnost*: Systém by měl být jednoduchý na zapamatování, aby se občasný uživatel mohl k práci se systémem vrátit i po určitém období, kdy se systémem nepracoval, aniž by se musel učit vše od začátku.
- *Chyby*: Systém by měl mít nízkou chybovost, aby uživatelé dělali málo chyb při užívání systému a aby případná chyba byla snadno odstranitelná. Ke katastrofickým chybám dojít nesmí vůbec.
- *Uspokojení*: Práce se systémem by měla být příjemná, aby byli uživatelé subjektivně spokojeni s jeho užíváním, aby se jim líbil.

Zdroj: [Nielsen1]

² *Usability*; lze se setkat i s překladem *uživatelnost*

Z těchto pěti rysů jsou poslední čtyři běžně testované. *Testování použitelnosti* je nejpřesnější metodou, jak použitelnost měřit. Vedle něj lze použít i *inspekci použitelnosti* (usability inspection), která je levnější a je možno ji použít i v počátečních fázích vývoje, ovšem nepřináší tak přesné výsledky a nemusí odhalit všechny chyby. Jednoduchá forma inspekce použitelnosti bude použita pro potřeby této práce.

3 Rozdíly mezi uživatelskými rozhraními v obecných prvcích

Přes rostoucí soupeření webových a desktopových aplikací naznačené v úvodu existuje řada prvků, v nichž se jejich uživatelská rozhraní tradičně liší. Než se dostaneme ke konkrétním příkladům aplikací, přistoupíme nejprve teoreticky ke kořenům těchto rozdílů. Některé z nich mají původ historický v zavedených zvyklostech, některé jsou ještě dnes způsobeny použitím rozdílných technologií.

3.1 Konvence desktopových aplikací

Oproti webovým aplikacím mají ty desktopové za sebou delší dobu vývoje a fungují na rozdíl od webových v prostředí pro běh aplikací přímo stvořeném. Různé operační systémy dlouhou dobu koexistovaly relativně odděleně bez možnosti přenositelnosti softwaru a aplikace byly primárně vyvíjeny pro konkrétní systém. Tyto podmínky daly vzniknout určitým sadám konvencí, které se staly pro aplikace jednotlivých platforem zažitými. Uživatel tak od všech desktopových aplikací očekává v obdobných situacích také obdobné chování. Jde logicky zejména o často se vyskytující prvky, jako jsou dialogová okna, hlavní menu aplikace, respektive jeho struktura, či tlačítka pro nejčastější operace, jako jsou například otevření nového dokumentu, vrácení poslední úpravy nebo zavření aplikace. Dále uživatel automaticky očekává podporu různých zažitých postupů, jako je copy & paste, drag & drop, nebo použití standardních klávesových zkratk. Uživatelé jsou také zvyklí, že různé aplikace řešící stejný problém se ovládají obdobně, existují tedy i konvence pro ovládání jednotlivých kategorií aplikací.

3.2 Situace webových aplikací

Běžný vzorec práce ve webovém prohlížeči byl – a ve většině případů stále je – kliknutí, odeslání požadavku a čekání na překreslení stránky. Moderní webové aplikace využívající technologií jako Ajax či Flash vyžadují po uživateli prolomení starých zvyků a přivyknutí novým, nebo alespoň adaptaci starých zvyků v novém prostředí. Přestože se vzhled pokročilých webových aplikací do značné míry blíží známým vzorcům uživatelských rozhraní klasických desktopových aplikací, uživatel často neočekává možnost dvojkliku, použití pravého tlačítka myši, klávesových zkratk či přímé manipulace s objekty kupříkladu stylem drag & drop [Powell]

K původu tohoto uživatelova tápání, tedy absence jednoznačných naučených postupů, lze přistupovat ze dvou pohledů, které se ovšem ve svých příčinách prolínají. Jde za prvé o rozdíl mezi prací s webovou aplikací a s webovou prezentací, za druhé o stanovení hranice mezi rozhraním webové aplikace a rozhraním webového prohlížeče.

3.2.1 Webová aplikace a webová prezentace

Webový prohlížeč je prostředím, ve kterém je uživatel zvyklý přistupovat k obsahu hypertextového typu, tedy k tradičním stránkám převážně pasivního charakteru. v takovém modelu je stránka skutečně pouhým vykresleným obsahem a uživatel postupuje podle navyklých vzorů pro práci v tomto prostředí, tedy prostřednictvím jasně rozlišených hypertextových odkazů a formulářů, případně s pomocí nástrojů webového prohlížeče. Důležité je, že takové ovládání je standardní, funguje na všech webových stránkách stejně. Oproti tomu přítomnost nestandardních ovládacích prvků – byť třeba standardních v prostředí desktopových

aplikací – v tomto prostředí často neočekává a tedy ani nezkouší a snaží se hledat řešení prostřednictvím zavedených vzorů ovládání. Na neobvyklé možnosti ovládání, jako zmíněný drag & drop, musí být uživatel upozorněn viditelnou nápovědou či někdy až tutoriálem při prvním použití.

Ovšem ani aktivní a dynamické prvky nejsou ve všech webových aplikacích v identickém složení. Uživatel se tedy nenaučí, že jakmile se ocitl v prostředí webové aplikace, má k dispozici pevně stanovenou sadu nástrojů pro práci s touto aplikací, protože žádná taková sada neexistuje a uživatel je nucen u každé nové aplikace znovu zjišťovat, co daná aplikace umožňuje a co ne. Problémem je i rozpoznání hranice mezi pasivní prezentací a aktivní aplikací, tyto modely se volně prolínají, čímž opět přidávají uživateli práci s hledáním v hypertextu neobvyklých ovládacích prvků.

Důsledkem nemožnosti přesného vymezení prostředí webové aplikace od prostředí běžné webové prezentace je skutečnost, že uživatelské rozhraní webové aplikace musí vycházet z konvencí webových prezentací, nelze například zcela aplikovat konvence desktopových aplikací, protože uživatel přicházející z Webu očekává uvnitř webového prohlížeče naučené ovládací prvky. [Powell]

3.3 Hranice mezi rozhraním webové aplikace a rozhraním webového prohlížeče

Druhý pohled také vychází z modelu webového prohlížeče jako aplikace a webové stránky jako touto aplikací prezentovaného obsahu. Problémem je, že samotný webový prohlížeč má vlastní uživatelské rozhraní a ve chvíli, kdy se vykreslovaným obsahem stává další aplikace, najednou zde paralelně koexistují uživatelská rozhraní dvě. Potom v některých situacích není jasné, které ovládací

prvky ještě patří do uživatelského rozhraní prohlížeče a které do uživatelského rozhraní aplikace. Jde například o pravé tlačítko myši, které za normálních okolností slouží pro nabídku prohlížeče, ale mnohé aplikace (za všechny kancelářský balík *Google Docs*) ho využívají a tím mění standardní chování prohlížeče.

Dalším takovým ovládacím prvkem je klávesnice. Ať už jde o jednoklávesové zkratky, používané například v široké škále aplikací Googlu, či kombinace kláves. Jednoklávesové písmenné zkratky převážně s ničím nekolidují, protože většina prohlížečů na stisk samotného písmene nijak nereaguje. Některé webové prohlížeče ale lze nastavit tak, aby při psaní okamžitě začaly hledat v textu stránky, nebo lze přiřadit těmto klávesám funkce. v žádném z těchto případů ale uživatel neočekává reakci na stisk klávesy od samotného obsahu, tedy ani od webové aplikace. K výraznějším konfliktům dochází v případě zavedených kombinací kláves, jako nejlepší příklad si můžeme vzít zkratku Ctrl+S pro uložení. v uživatelském prostředí webového prohlížeče tato klávesová zkratka zpravidla slouží pro uložení celé aktuální stránky do souborového systému počítače³, zatímco v některých webových aplikacích se stává součástí uživatelského rozhraní aplikace a je použita pro uložení dokumentu rozpracovaného uvnitř této aplikace.

Obecně řečeno je klávesnice i práce s myší mimo jednoduché kliknutí primárně považováno za součást uživatelského rozhraní prohlížeče a v opačných případech je vhodné na to uživatele upozornit, protože se pohybuje v prostředí, kde takové ovládání neočekává.

³ z ovládání Microsoft Internet Exploreru byla tato zkratka odstraněna ve verzi 7 (Od verze 7 přejmenován na *Windows Internet Explorer*), ve všech ostatních z pěti nejpožívanějších prohlížečů v aktuálních verzích funguje shodně k uložení celé stránky

3.3.1 Specifické aspekty práce v prohlížeči

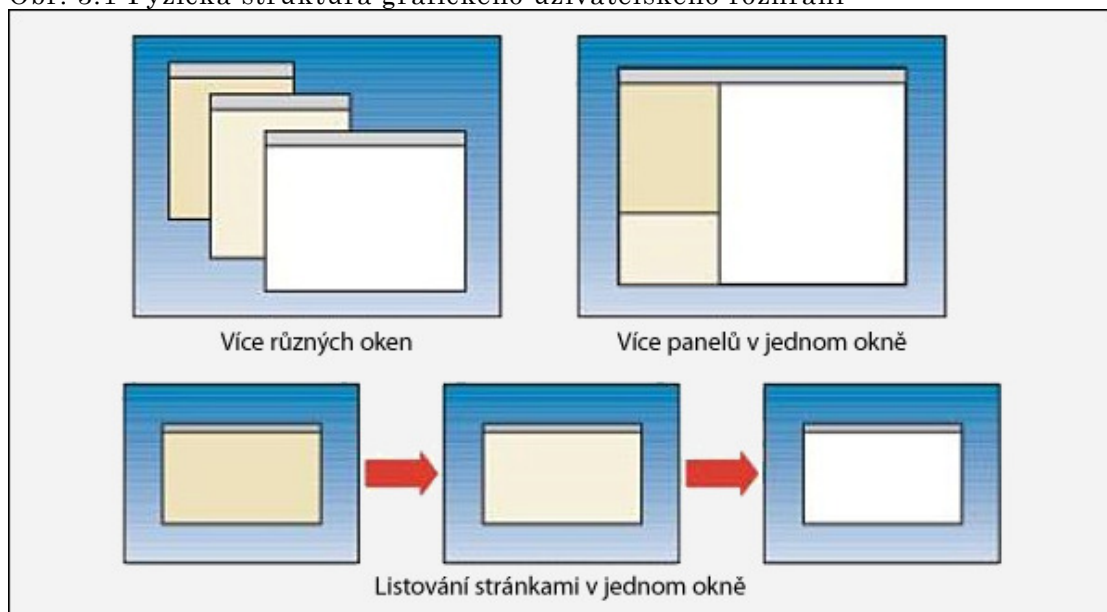
I webová aplikace si stále zachovává některé podstatné vlastnosti běžné webové stránky a může s ní být zacházeno prostřednictvím nástrojů prohlížeče jako s jinou stránkou. Problémy může přinášet například uživatelská manipulace s URI (unified resource identifier), která je v prostředí Webu běžná. Typickou manipulací je uložení záložky nebo zkopírování adresy pro odeslání odkazu jinému uživateli. v takových případech mohou nastat dvě zcela protikladné nechtěné situace. První problém nastane, když chce uživatel například v aplikaci napsané v Ajaxu nebo ve Flashi odkázat na konkrétní stránku uvnitř aplikace, řekněme na určitou fotku uvnitř alba, ale aplikace není v tomto směru ošetřena a její URL se při navigaci uvnitř této aplikace nemění. Uživatel potom nechtěně dostane adresu na úvodní obrazovku aplikace, což vůbec nepožadoval. Zcela opačný případ, přestože ne tak běžný, může nastat ve chvíli, kdy uživatel naopak chce získat odkaz na celou aplikaci a místo toho dostane URI určitého jejího stavu. [Proctor]

S URI přímo souvisí i problematika tlačítka *zpět*, která je při návrhu webových aplikací kapitolou sama o sobě. Toto tlačítko je jedním z hlavních navigačních prvků a primární uživatelova „úniková cesta“. Navyklým vzorcem chování na obsahově zaměřeném webu je únik prostřednictvím tohoto tlačítka, jakmile uživatel usoudí, že se dostal jinam, než si představoval. [Powell] Návrh webové aplikace tedy musí nejen počítat s přítomností tlačítka zpět v prohlížeči, ale hlavně s návykem uživatele používat v prostředí webového prohlížeče tento způsob ovládání. První požadavek není tak složitý, jde zejména v případě ajaxových a flashových aplikací, kde často existuje riziko nechtěného opuštění aplikace, o nutnost nějakým způsobem ošetřit funkci tohoto tlačítka. Běžně využívanou možností

je zcela zabránit jeho funkčnosti otevřením aplikace v novém okně, respektive záložce, a tedy startem s prázdnou historií. Další alternativou je upozornění uživatele před opuštěním aplikace. Nejvhodnějším řešením ovšem je, pokud je to možné, implementace funkce tlačítka zpět do aplikace tak, aby skutečně vracelo aplikaci do předchozího stavu. Takovým postupem je nejen ošetřeno tlačítko zpět proti nechtěnému opuštění aplikace, ale vyhovuje i druhému požadavku, tedy na chování v souladu s uživatelským očekáváním. Ovšem takto jednoznačné uživatelské očekávání významu tlačítka zpět existuje pouze u uživatelského rozhraní založeného na listování ve více stránkách, v jinak koncipované struktuře buď tlačítko nemá smysl, nebo musí být jeho význam pozměněn.

Tím se dostáváme k rozdílům ve fyzické struktuře grafického uživatelského rozhraní. Fyzická struktura znamená způsob rozložení obsahu a ovládacích prvků na obrazovce, Základní možnosti fyzické struktury aplikací dle Tidwella jsou znázorněny na Obr. 3.1.

Obr. 3.1 Fyzická struktura grafického uživatelského rozhraní

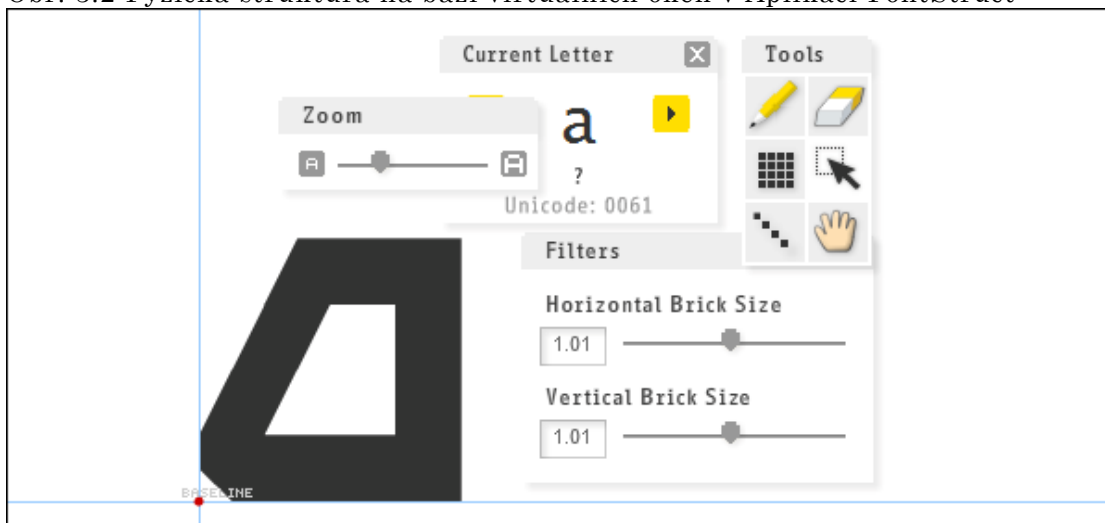


Zdroj: [Tidwell]

Desktopové aplikace využívají všech těchto struktur dle potřeb konkrétní aplikace. Struktura s více okny je typická pro složitý

software s velkým množstvím nástrojů či různých zobrazovaných informací, který zároveň klade vyšší požadavky na uživatele. Ten se musí v oknech dobře orientovat a dle vlastních potřeb si je nastavit. Na zcela opačném konci leží struktura založená na listování stránkami v jednom okně. v případě desktopových aplikací je adaptována pro maximální snížení nároků na uživatele, například v podobě klasických průvodců (wizardů). U webových aplikací je situace jiná, v prostředí World Wide Webu je listování stránkami v jednom okně strukturou prvotní a pevně zakořeněnou, takže ve zvýšené míře bývá zachována i ve webových aplikacích. Pokud je použita struktura na bázi více různých oken, často je dnes místo více skutečných oken prohlížeče použito oken „virtuálních“ uvnitř samotné stránky. Příklad webové aplikace s takovou strukturou je na Obr. 3.2.

Obr. 3.2 Fyzická struktura na bázi virtuálních oken v Aplikaci FontStruct



Zdroj: <http://fontstruct.fontshop.com/>

Struktura s více panely byla ve webových aplikacích dříve běžně uplatňována prostřednictvím *framů*, ale pro mnoho problémů z tohoto postupu plynoucích byl tento prvek nahrazen jinými způsoby implementace, nejčastěji s využitím CSS a Ajaxu.

Vazba na prostředí prohlížeče ale neznamena pro webové aplikace jen omezení, je třeba zmínit, že v prostředí webového prohlížeče je uživatel ochoten tolerovat několikanásobně pomalejší reakce systému, než by považoval za únosné u desktopové aplikace.

3.4 Shrnutí původu obecných rozdílů

V této kapitole popsané obecné rozdíly uživatelského rozhraní webové aplikace od uživatelského rozhraní aplikace desktopové mají společného jmenovatele, kterým je prostředí, ve kterém se dané aplikace nachází. U webových aplikací jde o vazbu na současný model webového prohlížeče a v něm vnořeného prostředí hypertextového obsahu. Přestože dnes již vývojáři mají technické možnosti k tomu, aby webové aplikace vybavili uživatelským rozhraním téměř identickým s těmi desktopovými, návrhy uživatelských rozhraní webových aplikací v těchto podmínkách musí stále respektovat, že webová aplikace není striktně oddělena od prostředí tradičního hypertextového webu a že je vnořena ve vlastní aplikaci webového prohlížeče. U desktopových aplikací zase musí být brán v potaz konzistentnější charakter uživatelských rozhraní na dané platformě a silněji zažitá uživatelská návyky a musí tedy být striktně dodrženy konvence a standardy dané platformy.

4 Srovnání UI vybraných aplikací

V minulé kapitole jsou rozdíly mezi uživatelskými rozhraními webových a desktopových aplikací rozebrány obecně a spíše teoreticky, tato kapitola se zaměří na konkrétní aplikace z praxe a bude zjišťovat, jestli skutečné rozdíly ve vybraných příkladech odpovídají popsaným obecným předpokladům a jaký je tedy vztah konkrétních uživatelských rozhraní ve skutečnosti. Kromě přímého srovnávání dále budou zkoumána uživatelská rozhraní jednotlivě z hlediska použitelnosti a budou porovnány jejich výsledky.

Pro praktické porovnání byly hledány aplikace na základě několika základních požadavků, které by měly zajistit dostatečnou relevantnost výsledků srovnávání. Hledané dvojice aplikací tedy musí splňovat následující podmínky:

- Aplikace musí řešit stejný problém, respektive pokrývat co nejpodobnější oblasti, a to pokud možno v obdobném rozsahu funkcionality.
- Aplikace musí být dostupná zdarma.
- Aplikace by měla fungovat na všech běžnějších operačních systémech, tedy Windows, Mac OS X, Linux, případně BSD a UNIX. Webová aplikace nesmí být vázána na konkrétní webový prohlížeč.
- Musí jít o kategorii aplikací určených pro každodenní či běžnou práci.
- Aplikace by měla být co možná nejvýznamnějším produktem v dané oblasti.

- Nesmí jít o aplikaci, se kterou je autor této práce zvyklý pracovat.

Tyto podmínky splnily textové procesory OpenOffice.org Writer 3 jako desktopová aplikace a Zoho Writer jako její webový konkurent, dále z kategorie aplikací pro instant messaging⁴ podporujících širší sadu protokolů byl vybrán Pidgin jako desktopová varianta a meebo jako webová alternativa.

4.1 Postup

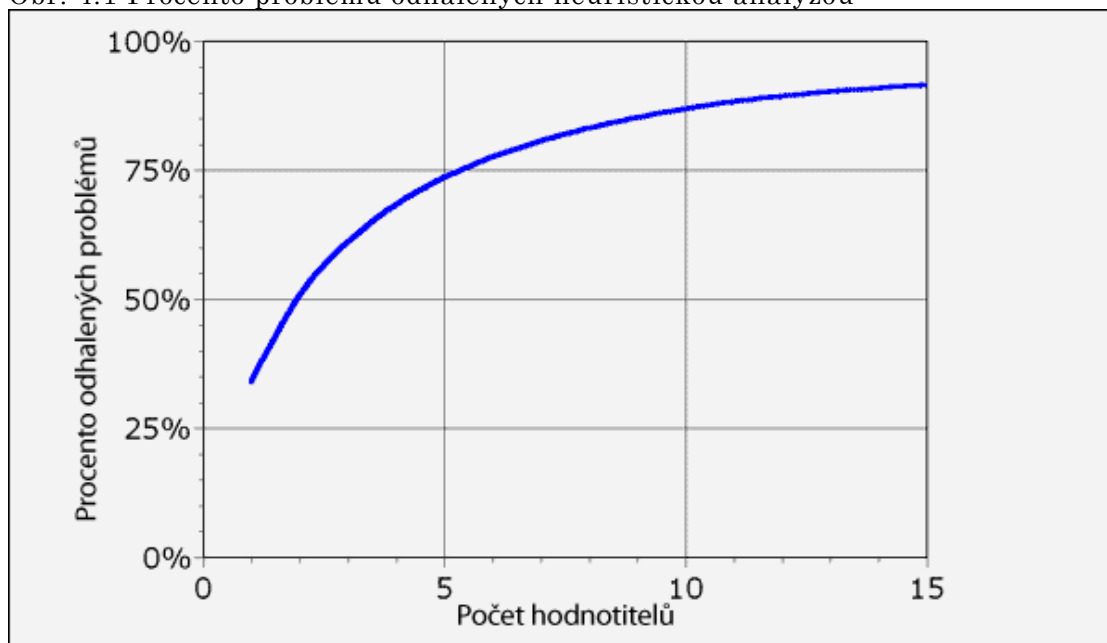
Pro hodnocení použitelnosti uživatelských rozhraní vybraných aplikací bude použita heuristická analýza použitelnosti. Tato metoda slouží k identifikaci problémů použitelnosti na základě zavedených ergonomických pravidel (human factor principles). Jde o nejméně formální metodu inspekce použitelnosti, a tedy metodu výhodnou pro svoji rychlost a nízké náklady na provedení ve srovnání například s uživatelským testováním. Další výhodou heuristické analýzy použitelnosti je aplikovatelnost již ve fázi návrhu, což pomáhá odhalit chyby v rané části vývoje a může snížit náklady na prototypování či v horším případě náročnější pozdější opravy. Heuristická analýza ani další metody inspekce použitelnosti ale nemohou u větších projektů plně nahradit uživatelské testování, tyto a další postupy je žádoucí při vývoji systému vhodně kombinovat.

Analytici provádějící heuristickou analýzu použitelnosti mohou být experti na ergonomii nebo HCI (*Human-computer interaction*,

⁴ Instant messaging je komunikace po internetu prostřednictvím krátkých převážně textových zpráv v reálném čase. Aplikace pro instant messaging se nazývá *instant messenger* neboli IM. v češtině bývá používán termín *komunikační klient*.

interakce člověka a počítače), ale metoda přináší relevantní výsledky i při provádění méně specializovanými hodnotiteli. Běžně analýzu provádí dva až pět hodnotitelů, protože různí členové hodnotící skupiny často nacházejí vzájemně různé problémy. Ještě vyšší počet hodnotitelů potom už zpravidla neznamena dostatečný přínos vzhledem k rostoucím nákladům na celou analýzu. Heuristickou analýzu použitelnosti může provádět i jeden hodnotitel, přestože výsledky takto získané bývají slabé. Vztah mezi počtem hodnotitelů a podílem odhalených problémů použitelnosti ukazuje Obr. 4.1, jde o průměr dat získaných ze šesti případových studií heuristické analýzy. v těchto případech bylo samostatným hodnotitelem identifikováno průměrně jen 35 % problémů použitelnosti. [Nielsen2] Z těchto studií ovšem také vyplývá, že závažné chyby v použitelnosti mají vyšší pravděpodobnost objevení, zatímco méně závažné problémy jsou v absolutních číslech početnější. Proto je i takto nízké procento odhalených chyb pro zkoumání použitelnosti rozhraní zásadní.

Obr. 4.1 Procento problémů odhalených heuristickou analýzou



Zdroj: [Nielsen2]

Pro účel této práce přitom heuristická analýza neslouží jako nástroj pro identifikaci chyb za účelem jejich odstranění, ale jako pomůcka při komparaci uživatelských rozhraní srovnatelných aplikací v rozdílných prostředích. Proto zde zcela stačí velmi hrubé, orientační výstupy, jaké obdrží i jeden hodnotitel. Samotná komparace potom vychází z předpokladu, že stejný hodnotitel odhalí v různých aplikacích alespoň přibližně podobné procento existujících chyb.

Heuristická analýza použitelnosti je prováděna tak, že hodnotitel či hodnotitelé vykonávají ve zkoumaném systému simulované úlohy a srovnávají jejich průběh s předem stanovenou sadou heuristik, tedy obecně definovaných pravidel znamenajících spíše odhad „od oka“, než úzce specifikované směrnice. [Nielsen3] Hodnotitel zaznamenává své pozorování ve formě seznamu identifikovaných chyb. v případě více hodnotitelů spolu tyto během doby pro hodnocení nekomunikují a až po uplynutí jsou jejich seznamy porovnány. Jednotlivým položkám dále může být přiřazena hodnota priority dle závažnosti problému. Výstupem je zpráva obsahující zpracovaný seznam identifikovaných chyb.

Východiskem pro tuto analýzu bude *Deset heuristik použitelnosti*, které v původní verzi publikoval v roce 1990 Jakob Nielsen jako základ pro metodu heuristické analýzy použitelnosti, a které po dalším zkoumání upravil v roce 1994 do současné podoby (Box 4.1). Těchto deset heuristik je základním kamenem metody heuristické analýzy použitelnosti dodnes.

1 Viditelnost stavu systému

Systém by měl vždy svému uživateli poskytovat rychlou a srozumitelnou zpětnou vazbu o svém stavu - tedy informace o tom, co se s ním právě děje.

2 Soulad mezi systémem a reálným světem

Systém by měl k uživateli promlouvat jeho jazykem, jeho slovy. Měl by používat pojmy a fráze, které uživatel zná, a měl by se vyhýbat obrátům a slovům vlastním pouze samotnému systému nebo jeho tvůrcům. Měl by se řídit zvyklostmi z reálného světa a informace by měl zobrazovat v přirozeném a logickém pořádku.

3 Svoboda a vláda uživatele

Uživatel často zvolí nějakou funkci systému omylem - a proto potřebuje jasně vyznačený "nouzový východ". Systém musí uživateli umožnit rychlé opuštění nechtěného stavu, a to bez složitého domlouvání se s ním. Systém by měl nabízet funkce "vrátit zpět" a "znovu".

4 Konzistence a standardy

Uživatel by nikdy neměl být nucen přemýšlet, zdá různé situace, akce nebo slova neznamenaají náhodou totéž. Systém by se proto měl řídit konvencemi dané platformy.

5 Předcházení chybám

Ještě lepší řešení, než dobré hlášení o chybě, je uvážlivý návrh systému, který chyby dokáže nejen předvídat, ale především jim předcházet.

6 Rozpoznání má přednost před vybavováním si

Objekty a možnosti volby a akcí systému musí být viditelné. Uživatel by neměl být nucen pamatovat si informace z jedné části dialogu se systémem poté, co přejde do jiné části. Instrukce ohledně používání systému by měly být viditelné nebo snadno dosažitelné, kdykoli je to vhodné.

7 Flexibilita a efektivnost použití

Nástroje k urychlení práce se systémem by měly zůstat pro nováčky neviditelné, pro zkušenější uživatele by však měly být k dispozici. Systém by tak měl být přizpůsobitelný, co se rychlosti a snadnosti ovládání týče, jak pro začátečníky, tak pro časté a zkušené

uživatelé. Uživatel by měl mít rychlý přístup k těm funkcím systému, které právě on často používá.

8 Estetika a minimalistický design

Dialog s uživatelem by neměl obsahovat informace, které nejsou pro funkci systému podstatné, nebo je jich zapotřebí jen zřídka. Každá nadbytečná jednotka informace soupeří o pozornost uživatele s informacemi podstatnými, a snižuje tak jejich relativní viditelnost.

9 Pomoc při rozpoznání chyb, při stanovení jejich příčin a při návratu k normálu

Hlášení o chybě systému by mělo být vyjádřeno běžným jazykem, mělo by přesně popsat problém, vysvětlit, proč k němu došlo, a konstruktivně nabídnout jeho řešení.

10 Náповěda a dokumentace

Ideální stav je sice ten, kdy systém ke svému používání žádnou náповědu ani dokumentaci nepotřebuje, někdy je však nezbytné je do systému zařadit. Takové informace by měly být snadno dosažitelné, měly by se soustředit pouze na pomoc při úkolu, který uživatel právě provádí nebo chce provést, měly by vyjmenovávat konkrétní kroky, které je třeba provést, a nemělo by jich být příliš.

Zdroj: [Nielsen3], překlad Lukáš Vodička

Přestože pro účely specializovanějších inspekcí použitelnosti vznikají řádově několikrát rozsáhlejší sady detailních pravidel, běžně obsahující přes sto a v některých případech přes tisíc detailních položek, pro heuristickou analýzu je naopak vhodné použít velmi omezenou sadu obecných heuristik pokrývajících základní principy použitelnosti. [Nielsen1] Nielsenových deset heuristik navíc není vázaných na žádný konkrétní typ systému, a tak je lze snadno aplikovat i pro účel této práce, tedy srovnání uživatelských rozhraní aplikací různých platforem, pro který jsou složitější metody inspekce nepoužitelné.

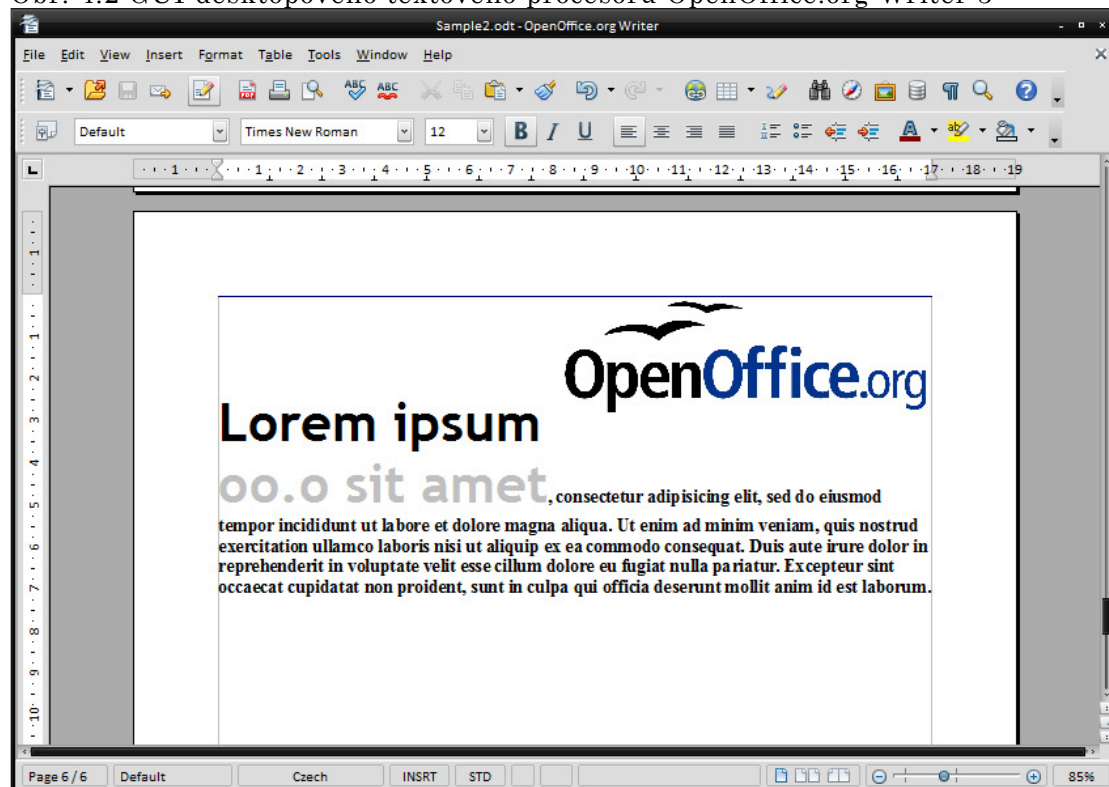
4.2 Textové procesory

Textové procesory mají mezi aplikačním SW významnou úlohu, patří k nejzákladnějšímu aplikačnímu SW a jsou nasazeny v každodenním užití. Při plnění požadavků na použitelnost vývojáři textových procesorů čelí několika náročným požadavkům. Pro mnoho počítačově málo gramotných uživatelů je textový procesor vstupní aplikací, a tak musí být jeho uživatelské rozhraní vyhovovat naprostému začátečníku. Zároveň je mnohým uživatelům základním pracovním nástrojem, se kterým tráví mnoho hodin denně, a tak musí tento SW splňovat i jejich požadavky na rychlost a efektivnost ovládání. Podobně protichůdné jsou nároky na širokou funkcionalitu a zároveň dostatečnou jednoduchost a přehlednost aplikace. a kvůli svému významu je kategorie kancelářských aplikačních balíčků, jichž je textový editor základní součástí, prostředím silně konkurenčním, bojují zde giganti jak Microsoft, Sun Microsystems či Google. Microsoft Word z placeného kancelářského balíku Office od roku 1989, kdy vznikla první verze pro Windows, donedávna významnou konkurenci neměl. Dnes je jeho hlavním soupeřem aplikace Writer z multiplatformního open source balíku OpenOffice.org vyvíjeného pod záštitou Sun Microsystems. Těmto desktopovým aplikacím roste zároveň další konkurence ve webových kancelářských balících. Těch existuje dlouhá řada, ale zřejmě nejvýznamnějšími představiteli jsou Google Docs, který je jednodušší a vsází spíše na spojení s dalšími službami Googlu, a Zoho Office Suite od společnosti Zoho. Google Docs jsou dostupné zcela zdarma, Zoho Office Suite nabízí zdarma základní kancelářské aplikace a některé rozšiřující, například Zoho CRM nebo Zoho Projects, jsou nabízeny ve velmi omezené podobě nebo za poplatek.

4.2.1 Představení

OpenOffice.org Writer 3

Obr. 4.2 GUI desktopového textového procesoru OpenOffice.org Writer 3



Zdroj: Vlastní zpracování autora

Zdarma distribuovaný open source kancelářský balík OpenOffice.Org vznikl v roce 2000, kdy Sun Microsystems uvolnil zdrojový kód balíku StarOffice, zakoupený od německé společnosti StarDivision, pod licencemi LGPL a Sun Industry Standards Source License (SISSL, tuto licenci od roku 2005 Sun nepoužívá, dnes je OpenOffice.org jen pod licencí LGPL). v roce 2005 vyšla významná verze 2.0, v říjnu 2008 byla vydána 3.0 a v květnu 2009 nejnovější 3.1

Vzhled uživatelského rozhraní je ovlivněn použitou knihovnou pro GUI, OpenOffice.org není pevně svázán s jednou, proto se v různých systémech a s využitím různých toolkitů může vzhled uživatelského

rozhraní výrazně lišit. To platí zejména na Linuxu, kde je možné například místo defaultní knihovny Qt využít GTK, která je dnes společná mnoha populárním linuxovým aplikacím. Pro tuto práci bude použito sestavení pro Windows z běžné distribuce, a to ve verzi 3.0.1; verze 3.1.0 nepřináší podstatné změny vzhledem k účelu této práce.

Na uživatelském rozhraní aplikace OpenOffice.org Writer se projevuje jeho multiplatformnost. Ta přináší problémy při snaze o dosažení požadavku na konzistenci uživatelského rozhraní se zvyklostmi platformy. Například zavedené pozice tlačítek u dialogových oken, které jsou velmi významné kvůli vysoké automatizaci uživatelské práce s nimi, se mohou lišit v závislosti na operačním systému. Tím, že je nabídnuto jedno defaultní rozložení těchto tlačítek, je na některých platformách poměrně závažně narušena konzistence s prostředím. Tlačítka dialogových oken jsou jedním z prvků, kde je dodržování konvencí platformy nejdůležitější. Vývojáři OpenOffice.org se ale evidentně snaží uzpůsobovat tyto významné prvky standardům prostředí Windows.

Zoho Writer

Obr. 4.3 GUI webového textového procesoru Zoho Writer



Zdroj: Vlastní zpracování autora

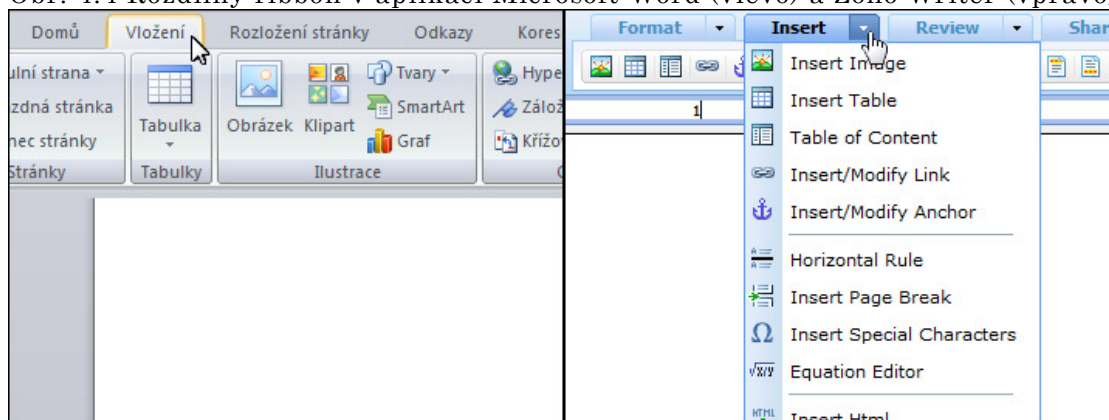
Zoho Writer je textovým procesorem z online kancelářského balíku Zoho Office Suite od společnosti Zoho (dříve AdventNet). Writer byl jako první z tohoto balíku webových aplikací představen v roce 2005, další aplikace přibývaly později.

Zoho Writer je i po čtyřech letech vývoje stále označen jako beta verze, tedy nefinální vývojová verze s možnými chybami. To však není u webových aplikací, byť nasazených v plném provozu s počty uživatelů v řádech milionů, natolik neobvyklé. Stejně tak stále nese označení „beta“ například populární webmailová aplikace Gmail, spuštěná v roce 2004 i mnoho dalších webových aplikací od Googlu. Vzhledem k charakteru vývoje webových aplikací nelze hovořit o verzích, v této práci tedy půjde o stav aplikace k červnu 2009; k aplikaci bude přistupováno webovými prohlížeči na jádře Gecko.

Aplikace je sice čistě webová, nicméně díky podpoře Gears (více viz 5.2) umožňuje práci offline. Dále podporuje všechny běžné formáty jako Microsoft Word (*.doc), Office Open XML (*.docx, *.docm), OpenDocument text (*.odt), OpenOffice.org text (*.sxw), Hypertext Markup Language (*.html, *.htm), Rich Text Format (*.rtf), JPEG, GIF či PNG. Kromě práce offline ovšem nabízí také výhody plynoucí z webového prostředí, především současnou spolupráci více uživatelů na jednom dokumentu, jednoduché sdílení dokumentů ale i mnohé vazby obsah dostupný online, ať už jde o vložení videa z YouTube do dokumentu či naopak přímé posílání příspěvků na blogy.

Je třeba zdůraznit, že uživatelské rozhraní využívá modelu *ribbon UI* známého například z rozhraní aplikace Microsoft Word 2007 (tam je pojem *ribbon* do češtiny překládán výrazem *pás karet*). Tento relativně moderní model patří mezi takzvané *task oriented* (zaměřený na plnění úkolů) a je vhodný pro zpřehlednění uživatelských rozhraní s širokou nabídkou funkcí. Vychází z analýz velkého množství dat o chování uživatelů, ze kterých vyplývá, že při provádění určitého úkolu uživatel potřebuje nejvíce po ruce jen užší sadu funkcí vázaných k dané činnosti. Model ribbon UI tedy sdružuje ovládací prvky, které budou často používány společně, do tematických skupin. v ribbonu je v jednu chvíli zobrazena právě jedna tato skupina ovládacích prvků a mezi skupinami lze přepínat formou záložek. Uživatelské rozhraní aplikace Zoho Writer navíc tento model doplňuje o možnost přímé volby kteréhokoliv ovládacího prvku z právě schované nabídky ve formě klasického rozevíracího menu (viz Obr. 4.4). Odstraňuje tím nevýhodu modelu ribbon UI v pomalejším přístupu k určité funkci, pokud ta není v aktuálně zobrazené nabídce, což bývá tomuto modelu často vytýkáno. Použitím popsané kombinace vzniká velmi efektivní uživatelské rozhraní.

Obr. 4.4 Rozdílný ribbon v aplikaci Microsoft Word (vlevo) a Zoho Writer (vpravo)



Zdroj: Vlastní zpracování autora

4.2.2 Přímé srovnání

Vzhledem ke skutečnosti, že obě aplikace se pohybují na špičce nabídky kancelářských balíků, a vzhledem k úsilí věnovanému v tomto oboru uživatelskému rozhraní nepřekvapí, že jsou rozhraní obou aplikací značně propracovaná. Liší se oproti předpokladům vyvozeným z kapitoly 3 spíše jiným použitým modelem UI, než vazbou na své prostředí. Tvůrci Zoho Writeru překonali potíže s implementací složitějších operací na webovou platformu a jejich aplikace bezproblémově podporuje například operace drag & drop se soubory, plně využívá pravé tlačítko myši a nabízí alespoň omezenou sadu obvyklých klávesových zkratk. Chybí zde jen podpora některých operací se soubory stylem copy & paste. Uživatelské rozhraní Zoho Writeru je spíše než s prostředím obsahově zaměřeného webu konzistentní s konvencemi desktopových aplikací. Také díky tomu je přebrání některých ovládacích prvků prohlížeče, jaké je popsáno v podkapitole 3.3, intuitivní a v žádném místě nevyvolává v uživateli pochybnosti.

OpenOffice.org Writer je oproti Zoho Writeru nepřehlednější a subjektivně působí těžkopádnějším dojmem. Částečně je to způsobeno větším rozsahem funkcionality, ale také zvoleným

modelem uživatelského rozhraní. Při přímém srovnání tedy nejvíce vyniká kontrast mezi klasickým modelem menu UI, doplněným o panel nástrojů ve formě nepříliš organizovaných ikon, a naproti tomu striktně strukturovaným modelem ribbon UI webového Zoho Writeru.

4.2.3 Heuristická analýza použitelnosti

Při analýze byly vykonávány simulované úlohy převážně charakteru základních úkonů, jako formátování a úpravy textu, práce s tabulkami a obrázky.

Odhalené rozpory se zvolenou sadou heuristik byly zaznamenány a byla jim přidělena priorita dle rozdělení popsaného v Box 4.2:

Box 4.2 Rozdělení priorit problémů v použitelnosti

- | |
|--|
| <ol style="list-style-type: none">1. Kritický problém: Závažný rozpor s heuristikami, který znemožňuje nebo výrazným způsobem komplikuje práci s aplikací2. Vážný problém: Chyba v použitelnosti uživatelského rozhraní, která ztěžuje práci s aplikací3. Lehký problém: Takový problém, který práci znatelně, ale jen mírně znepříjemňuje |
|--|

Zdroj: Stanovené autorem

OpenOffice.org Writer

V textovém procesoru z balíku OpenOffice.org byly identifikovány dva vážné a sedm lehkých problémů. Nejčastější porušenou heuristikou byl požadavek na konzistenci a standardy. Vážnými problémy bylo nepřehledné uživatelské rozhraní a systém pro automatickou opravu překlepů, který opravuje neznámé – byť správně napsané – výrazy na nezamýšlené, a navíc této úpravě chybí evidentní úniková cesta.

Zjednodušený zápis analýzy viz Příloha 1.

Zoho Writer

Webový textový procesor od společnosti Zoho skončil s výsledkem lehce horším, identifikovány byly čtyři vážné a tři lehké problémy. Vážné problémy se týkaly systému kontroly pravopisu, který se chová nevyzpytatelně, není jasný jeho stav zapnutí, vypnutí či průběhu kontroly a samotný průběh kontroly také není v souladu s konvencemi textových procesorů a chybí z něj evidentní úniková cesta. Dalším problémem bylo tlačítko *Undo* (zpět), které zřejmě v závislosti na průběžném ukládání často vrátí dva i více posledních kroků. Posledním závažným problémem je neschopnost provádět některé operace s více vybranými prvky zároveň, která vadí zejména při práci s více buňkami tabulek. Nejčastější problémovou heuristikou byla opět čtvrtá, tedy požadavek na konzistenci a standardy, dvakrát chyběla *svoboda a vláda uživatele*, tedy číslo tři.

V průběhu analýzy byl ve webové aplikaci Zoho Writer odhalen jeden kritický problém, kdy pohyb v historii prohlížeče (tlačítka zpět a vpřed) opakovaně vedl k zaseknutí aplikace – plátno s obsahem bylo prázdné. Žádná data nebyla ztracena, ale většinou bylo potřeba znovu načíst stránku. Při pozdějším pokusu tuto chybu zopakovat ale již aplikace fungovala korektně a z výsledků analýzy byl tedy tento záznam odstraněn.

Zjednodušený zápis analýzy viz Příloha 2.

4.2.4 Shrnutí

Výsledky obou heuristických analýz použitelnosti nejsou dostatečně rozdílné na to, aby bylo možno průkazně prohlásit jedno z uživatelských rozhraní co do použitelnosti za kvalitnější. Cenné je ale zjištění, že nejčastějším problémem je porušení konzistence

a konvencí, ať už v rámci aplikace či v rámci prostředí. Překvapivým výsledkem přímého srovnání i analýzy je skutečnost, že webový Zoho Writer není výrazně vázán konvencemi webového prostředí, ale spíše konvencemi textových procesorů. Vazba na Web tedy nedělá jeho uživatelskému rozhraní takové problémy, jak bylo očekáváno na základě teoretických předpokladů. Naopak desktopový OpenOffice.org Writer měl místy problémy s dodržáním konvencí v prostředí daného operačního systému.

4.3 Komunikační klienti

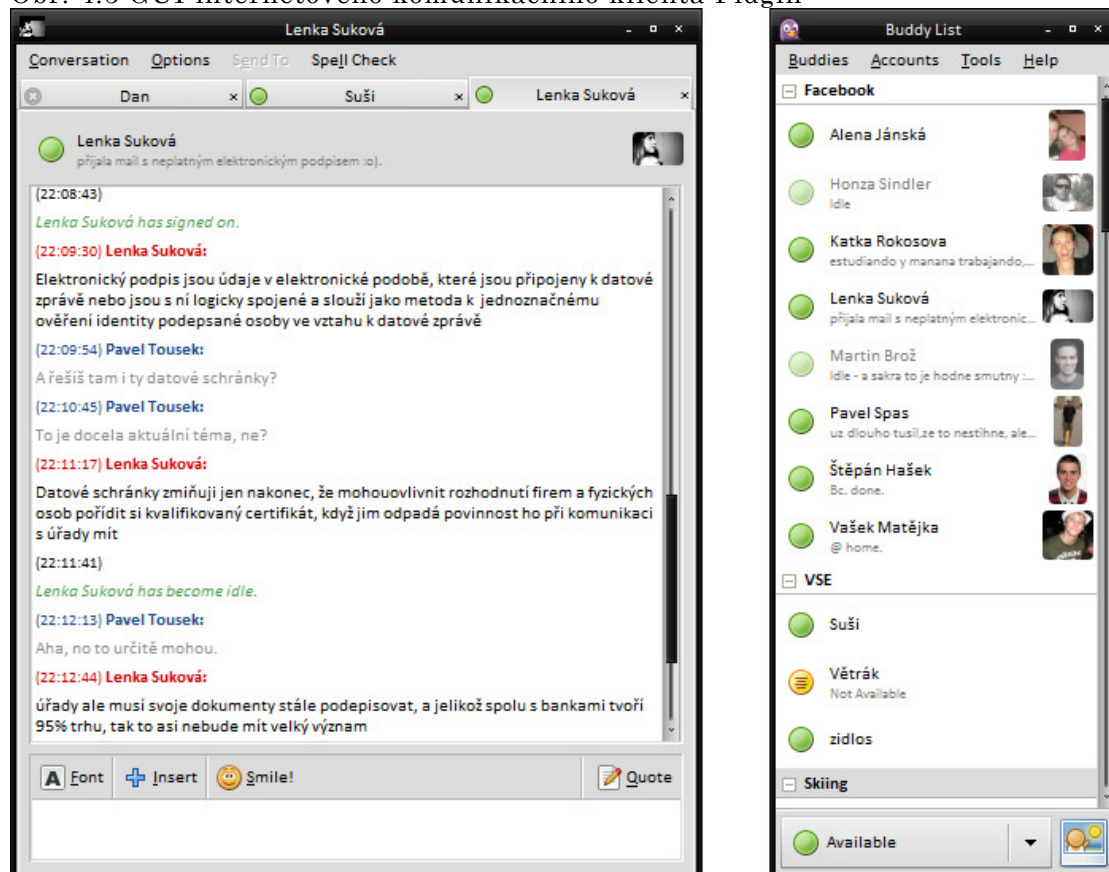
Internetový komunikační klient neboli *instant messenger* představuje relativně početnou skupinou aplikačního SW. Existuje totiž velmi vysoké množství různých protokolů a sítí pro instant messaging, pro různé sítě vznikají většinou proprietární klienti. Obvykle nezávisle na provozovateli těchto sítí jsou potom vyvíjeni klienti alternativní pro určitou síť či, dnes častěji, klienti pokrývající širší škálu komunikačních protokolů. k tomu vznikají různé aplikace pro různá prostředí, od mobilních telefonů přes všechny myslitelné operační systémy po čistě webové klienty.

Pro podrobnější prozkoumání byly vybrány aplikace *Pidgin* jako zástupce multiplatformní desktopové aplikace pod licencí GNU (General Public License) a webová aplikace *meebo* (skutečně s malým *m*), jedna z populárnějších webových aplikací pro instant messaging. Zajímavou okolností je, že tyto velmi rozdílné aplikace jsou postaveny na stejném komunikačním jádře *libpurple*. Testování probíhalo v sítích ICQ a Facebook,

4.3.1 Představení

Pidgin

Obr. 4.5 GUI internetového komunikačního klienta Pidgin



Zdroj: Vlastní zpracování autora

Pidgin, dříve *Gaim*, je komunikační klient původem z prostředí *nixových operačních systémů, kde je také nejrozšířenější. v prostředí Linuxu totiž nemá výrazného soupeře⁵, zatímco ve Windows je konkurence velmi tvrdá. Komunikační knihovna libpurple podporuje protokoly sítí AIM, MSN, Yahoo!, Jabber/XMPP, Facebook, ICQ, IRC, SILC, SIP/SIMPLE, Novell GroupWise, Lotus Sametime, Bonjour, Zephyr, MySpaceIM, Gadu-Gadu či QQ.

⁵ Kromě nativních IM pro Windows běžících v režimu emulace.

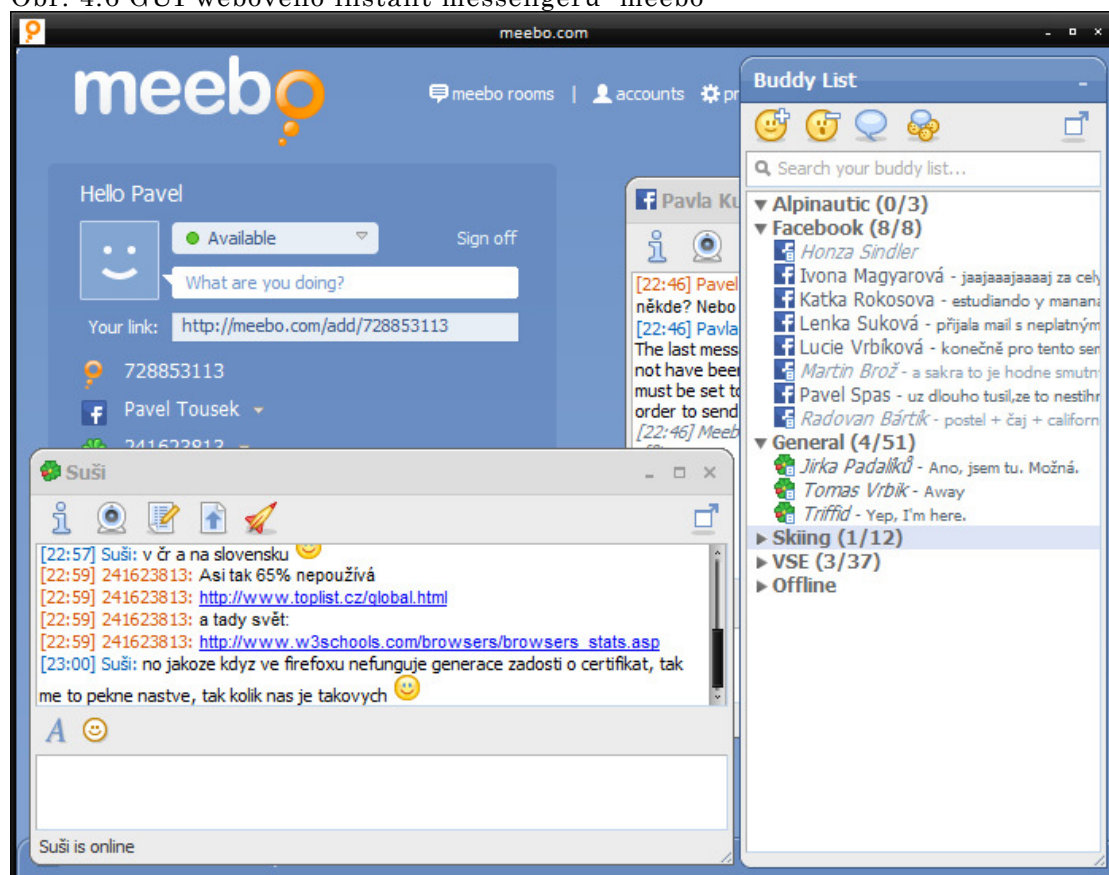
Uživatelské rozhraní je postaveno na toolkitu GTK, který je využíván pro mnoho dalších multiplatformních aplikací, například pro grafický editor GIMP, pro který byl toolkit původně vytvořen, nebo v Linuxu pro vývojové prostředí Eclipse. GUI tedy grafickými prvky, ale i rozvržením uživatelského rozhraní prozrazuje linuxové kořeny.

Uživatelské rozhraní Pidginu je podobné konkurenčním nativním aplikacím pro platformu Windows, ovšem oproti nim nabízí spíše slabé možnosti úprav. Členěné je tradičně na samostatné okno kontaktů, zde zvané *Buddy List*, a vlastní komunikační okno. Pidgin nabízí možnost otevírání více konverzací jak v samostatných oknech, tak v záložkách jednoho okna.

Testován byl Pidgin 2.5.7 v sestavení pro Windows.

meebo

Obr. 4.6 GUI webového instant messengeru meebo

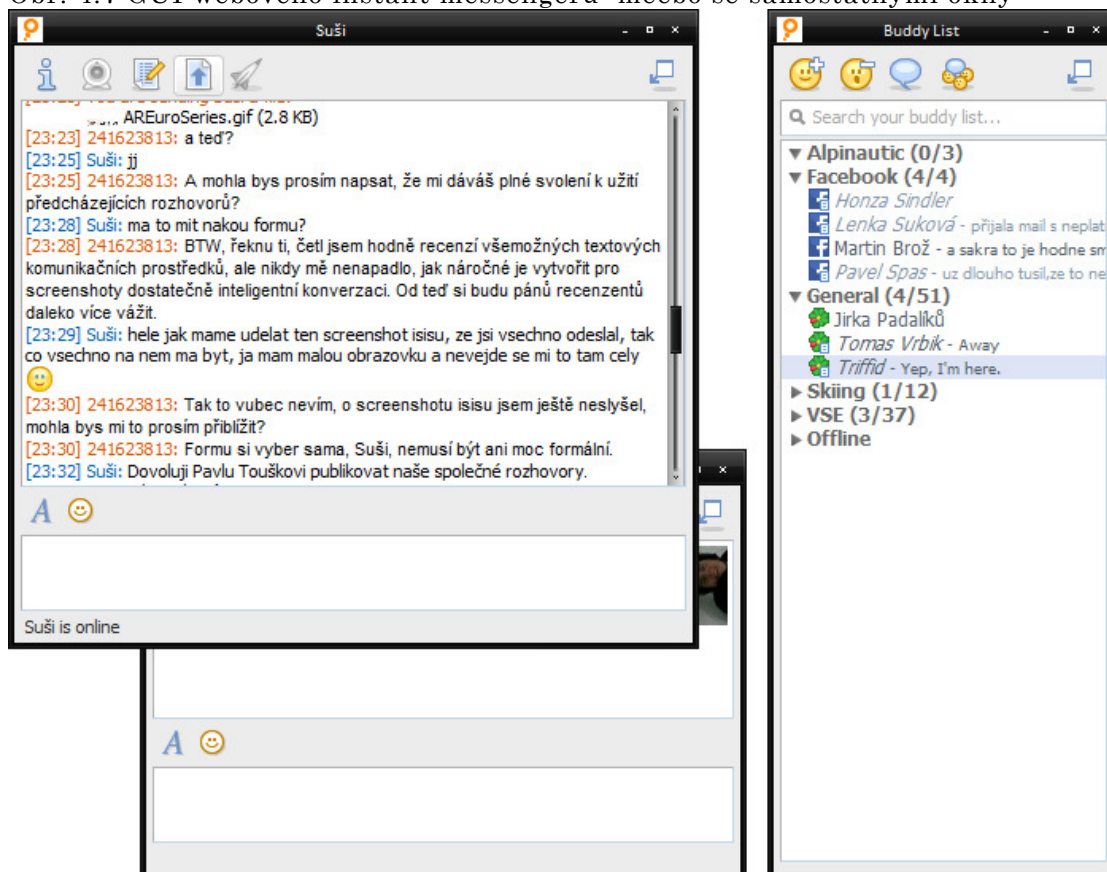


Zdroj: Vlastní zpracování autora

Aplikace *meebo* je, jak bylo zmíněno, postavena na stejné komunikační knihovně libpurple, nicméně jde o plně webovou aplikaci. Oproti Pidginu nabízí ještě širší paletu sítí, ke kterým se je schopna se připojit.

Uživatelské rozhraní postavené na Ajaxu je jednoduché s minimem možností nastavení. Defaultně má aplikace fyzickou strukturu více samostatných oken ve virtuální podobě uvnitř webového prohlížeče (Obr. 4.6), jednotlivá virtuální okna však lze osamostatnit do podoby reálných oken (Obr. 4.7).

Obr. 4.7 GUI webového instant messengeru meebo se samostatnými okny



Zdroj: Vlastní zpracování autora

Aplikace byla testována v červnu 2009 v prohlížečích na jádře Gecko.

4.3.2 Přímé srovnání

U této dvojice aplikací jsou na první pohled o něco evidentnější jejich kořeny – u Pidginu v původem linuxové vykreslovací knihovně GTK a u meebo ve webových technologiích. Zejména v základní podobě bez nových oken prohlížeče se meebo více blíží obsahovému pojetí webu uvnitř okna prohlížeče. To je podpořeno špatnou implementací pravého tlačítka myši a zcela chybějící podporou operací drag & drop se soubory. Tato dvojice aplikací tedy odpovídá předpokladům vyvozeným v kapitole Rozdíly mezi uživatelskými rozhraními v obecných prvcích.

V přímém srovnání vynikne ještě jeden podstatný rozdíl: Aplikace Pidgin nabízí výrazně větší možnosti nastavení a customizace, a to přesto, že mezi desktopovými instant messengery patří v tomto oboru mezi podprůměr.

4.3.3 Heuristická analýza použitelnosti

Při analýze byly vykonávány simulované úlohy charakteru běžné práce s internetovým komunikačním klientem, tedy textová komunikace, posílání souborů a manipulace s účty.

Odhaleným rozporům se zvolenou sadou heuristik byla opět přidělena priorita dle rozdělení v Box 4.2

Pidgin

Heuristická analýza použitelnosti uživatelského rozhraní aplikace Pidgin odhalila celkem sedm chyb, z toho tři vážné a čtyři lehké. Nejčastější porušenou heuristikou byla se třemi výskyty opět čtvrtá, tedy konzistence a standardy. Ve všech těchto případech šlo o konzistenci vnější, tedy s jinými instant messengery nebo prostředím operačního systému. Nekonzistenci s operačním systémem Windows lze připočítat na vrub linuxovému původu aplikace. Identifikovanými vážnými problémy byla zaměněná pozice tlačítek pro potvrzení a odmítnutí v různých dialogových oknech oproti konvencím aplikací v prostředí Windows, dále chybějící indikace stavu jednotlivých účtů při využití více protokolů současně a nakonec nekonzistence klávesových zkratk s jinými internetovými komunikačními klienty. Navíc tyto klávesové zkratky nejsou nastavitelné, což bývá u desktopových IM standardem.

Zjednodušený zápis analýzy viz Příloha 3

meebo

V rozhraní webové aplikace meebo byl identifikován jediný vážný problém a dva lehké. Vážným problémem je nekonzistentní chování pravého tlačítka myši v rámci aplikace. Pravé tlačítko funguje jako ovládací prvek webové aplikace pouze v seznamu kontaktů, kdekoliv jinde patří do uživatelského rozhraní webového prohlížeče a chová se tedy jako na jakékoliv běžné stránce. Navíc pro akci pravého tlačítka v seznamu kontaktů chybí alternativa bez použití tohoto tlačítka.

Lehké problémy v použitelnosti, tedy chybějící podpora operací drag & drop či copy & paste se soubory a příliš dotěrné prvky reklamního charakteru jsou běžnými atributy webových aplikací

Zjednodušený zápis analýzy viz Příloha 4

4.3.4 Shrnutí

Z porovnání počtů problémů identifikovaných heuristickou analýzou těchto dvou aplikací již lze vyvozovat určité závěry, protože rozdíl je dostatečně markantní. Konkrétně byly identifikovány tři závažné a čtyři lehké problémy u aplikace Pidgin proti jednomu vážnému a dvěma lehkým u aplikace meebo. Výsledkem tedy je, uživatelské rozhraní aplikace meebo vykazuje vyšší použitelnost. K tomuto závěru je třeba poznamenat, že rozsah funkcionality a možností nastavení webové aplikace meebo je výrazně užší. Ovšem vzhledem k tomu, že jde o webovou aplikaci relativně konzistentní s prostředím webu, není ani taková šíře možností na rozdíl od desktopové aplikace uživatelem očekávána.

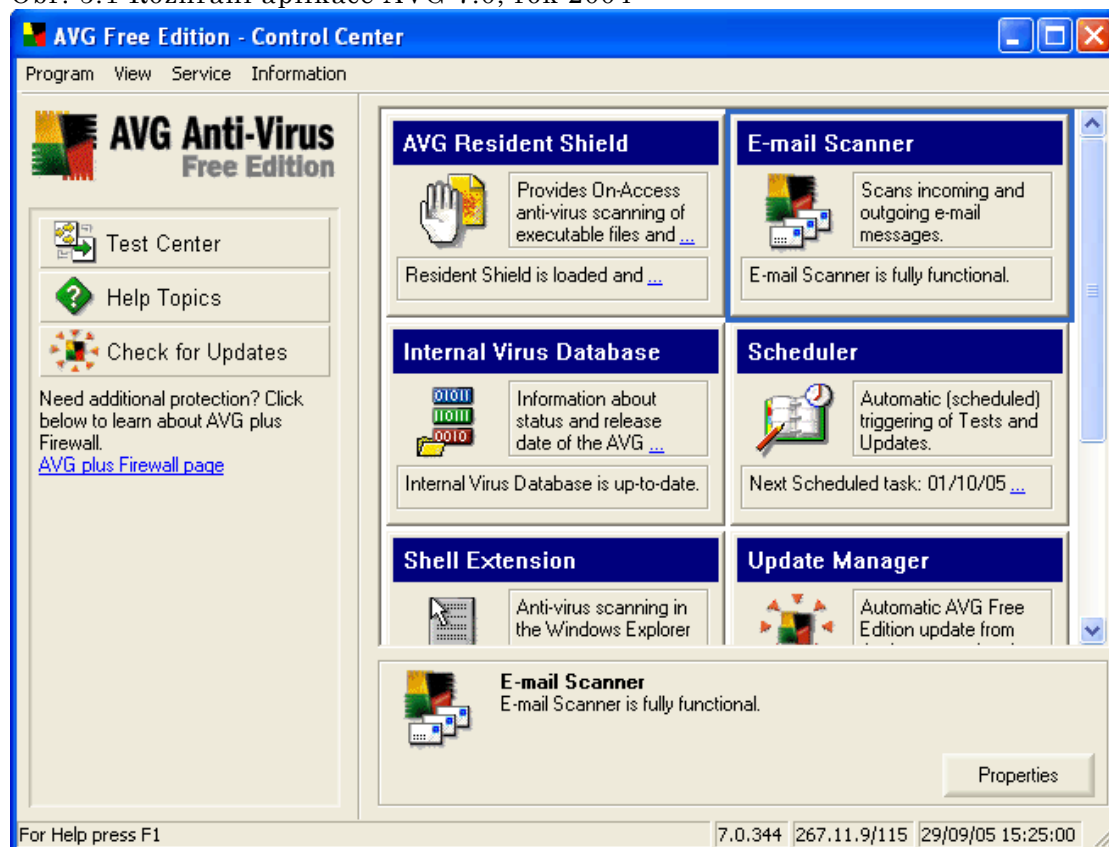
5 Konvergence

Konkurence mezi webovými a desktopovými aplikacemi nevyhnutelně vede k jejich vzájemnému ovlivňování a částečně přibližování. Tato kapitola se zaměří na krátkou analýzu příkladů takovýchto změn sbližujících uživatelské rozhraní aplikací obou platforem, dále na některé jevy a technologické směry vedoucí k takovému sbližování a v závěru dojde ke shrnutí do predikce budoucího vývoje vztahu uživatelských rozhraní webových a desktopových aplikací.

5.1 Příklady vývoje aplikací v čase

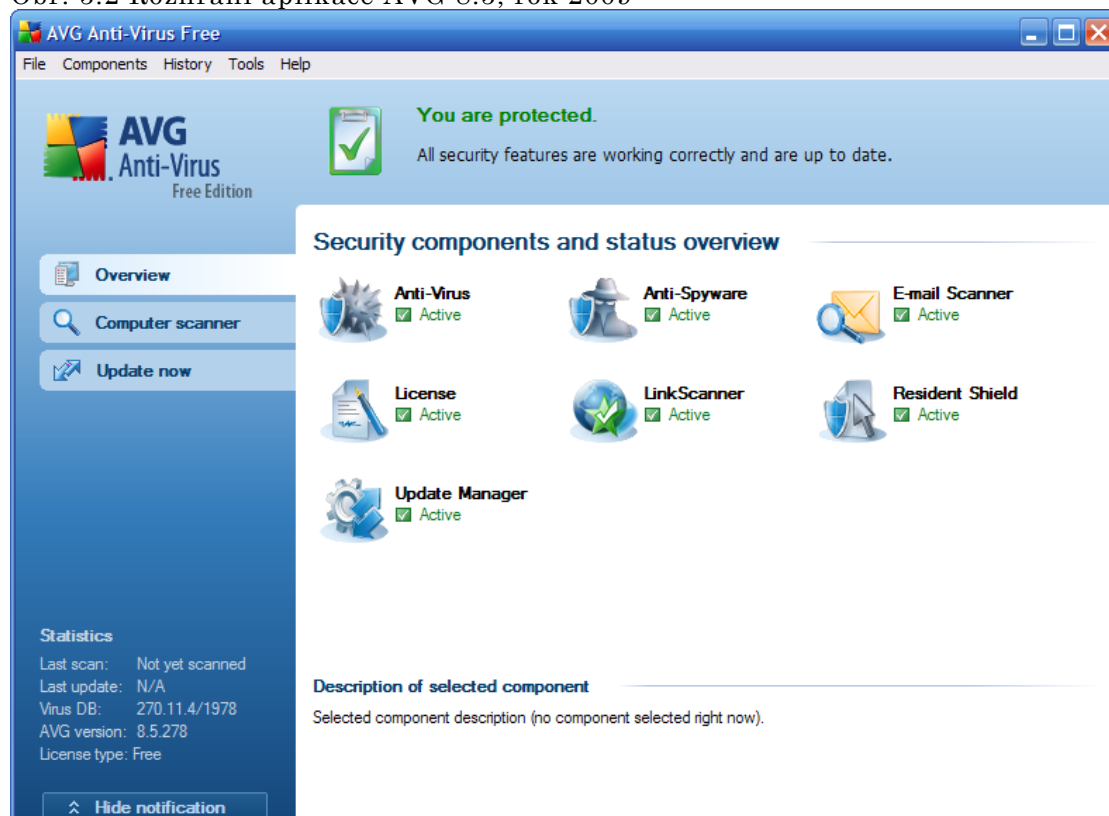
Na první pohled nejvíce viditelnou oblastí sbližování uživatelských rozhraní je grafický design a rozložení prvků (layout aplikace) obecně. Zde lze pozorovat velké posuny jak na ze strany webových, tak desktopových aplikací. Desktopové aplikace se často odklánějí od unifikované šedé grafiky a nabízejí líbivější či prostě jen nějak odlišný vzhled, více také podléhají módním trendům. Jako příklad poslouží rozšířený antivirový SW AVG Anti-Virus Free Edition. Obr. 3.1 ukazuje rozhraní této aplikace ve verzi 7.0 z roku 2004. Grafické uživatelské rozhraní je celé v režii nativních zobrazovacích knihoven. Přestože o pět let novější verze 8.5 (Obr. 5.2) jen lehce mění jednotlivé ovládací prvky a jinak zůstává u modelu Menu UI s doplňujícím levým panelem, a tedy nic zásadního se v uživatelském rozhraní nemění, vzhled této verze silně připomíná rozložení webové stránky a kromě zachovaného horního menu užívá pro vykreslení veškerých prvků vlastní grafiku.

Obr. 5.1 Rozhraní aplikace AVG 7.0, rok 2004



Zdroj: <http://howdo-i.com/hdipcsecurity.shtml>

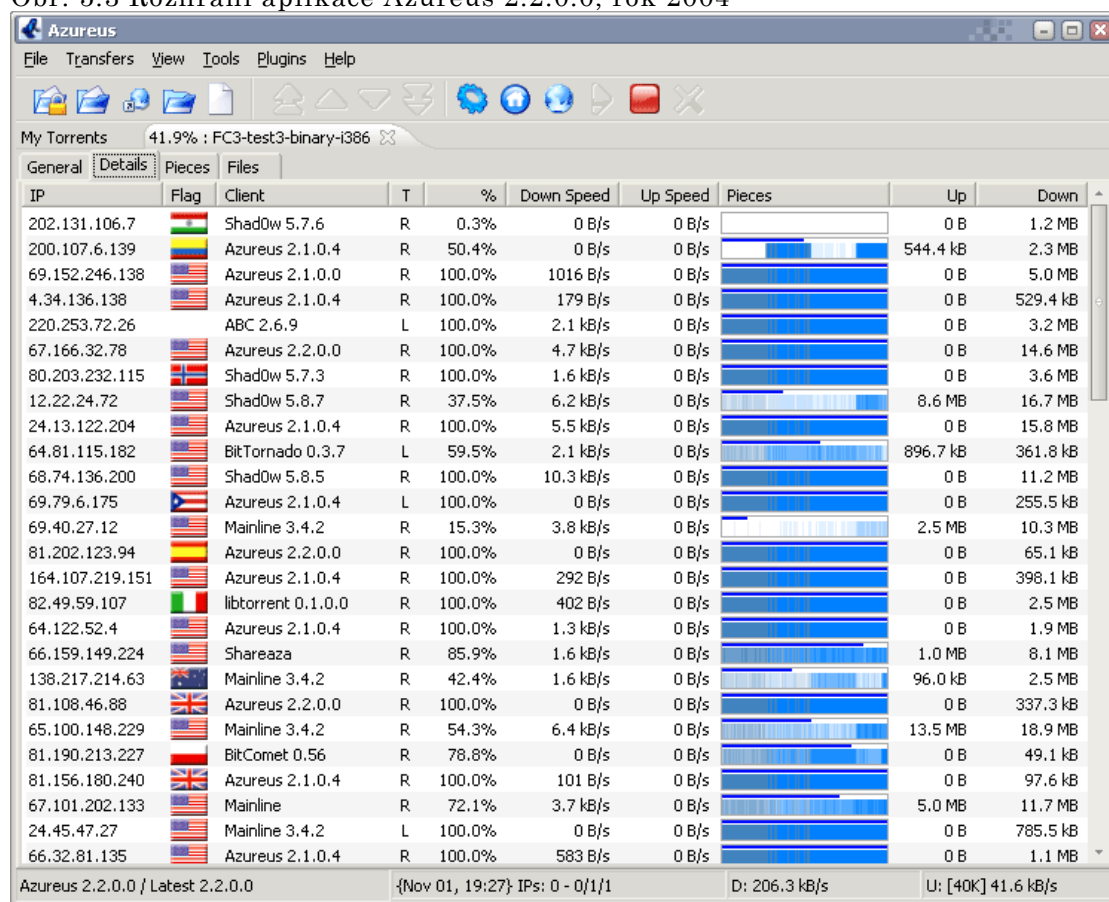
Obr. 5.2 Rozhraní aplikace AVG 8.5, rok 2009



Zdroj: <http://www.extrawindows.cz/novinky-avg-85-musicbee-dalsi>

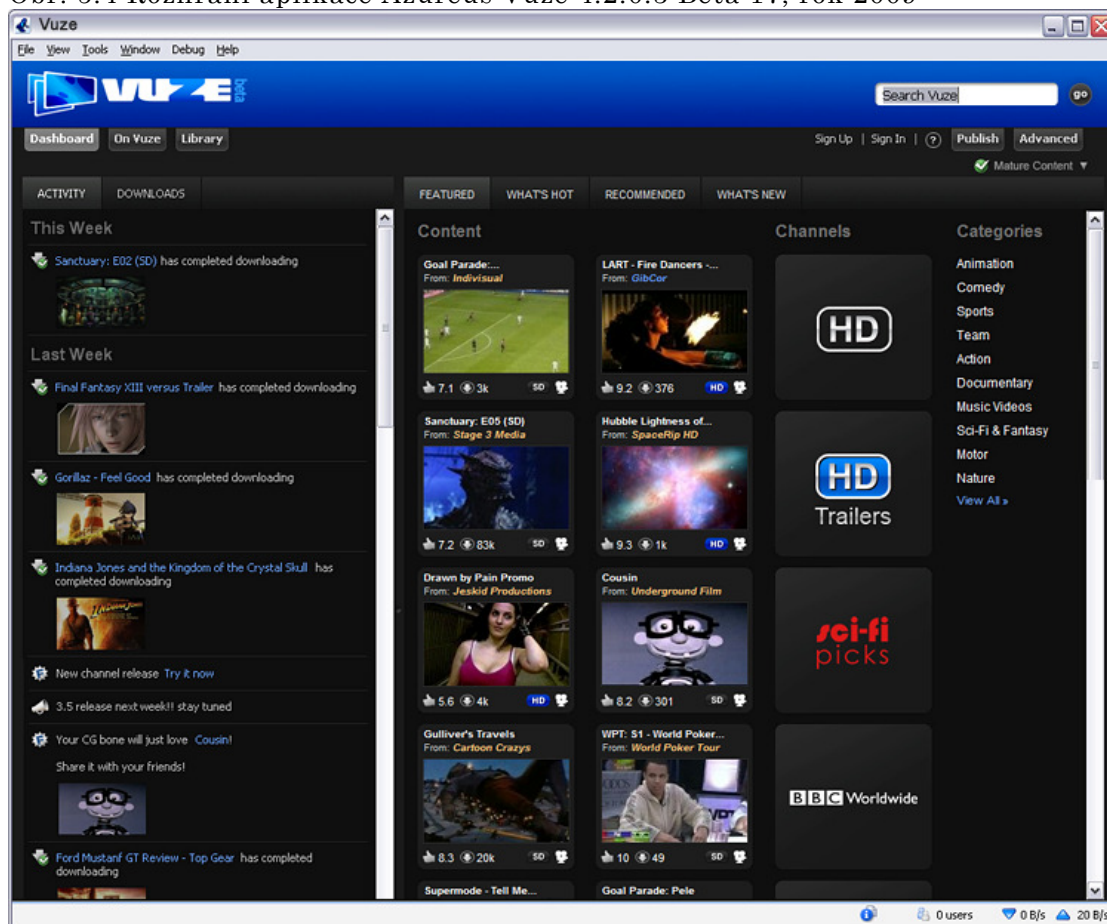
Ne ve všech případech ale dochází ke změnám čistě grafickým. Případ proměny aplikace Azureus pro sdílení v P2P síti BitTorrent je naopak z těch radikálnějších. Azureus prošel v roce 2007 radikální proměnou, kdy zcela změnil kromě jména také kompletní uživatelské rozhraní. Zatímco původní rozhraní (Obr. 5.3) ničím nevybočovalo z řady dalších BitTorrent aplikací, Vuze – jak se nově aplikace jmenuje – vypadá spíše jako webový videoportál (Obr. 5.4). Přitom základní funkcionality aplikace, tedy P2P sdílení, je stále stejná, jen s novou centrální stránkou, která je skutečně integrovaným webovým prohlížečem napojeným na portál Vuze nabízející sledování a stahování videa ve vysokém rozlišení. Nové rozhraní také výrazně zjednodušuje sadu ovládacích prvků.

Obr. 5.3 Rozhraní aplikace Azureus 2.2.0.0, rok 2004



Zdroj: http://www.svethardware.cz/forum/download/10833-azureus_Yul.gif

Obr. 5.4 Rozhraní aplikace Azureus Vuze 4.2.0.3 Beta 17, rok 2009



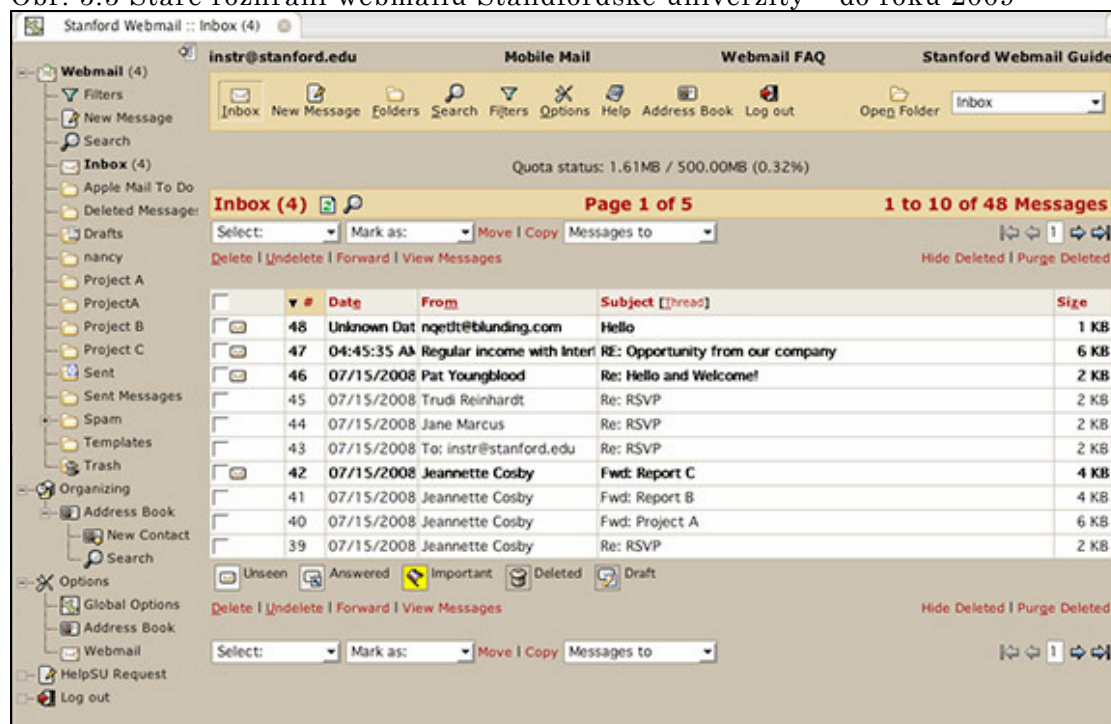
Zdroj: <http://www.sosej.cz/Screenshot/Azureus-Vuze-beta.html>

Tato aplikace dobře demonstruje tendence, které se projevují u více desktopových aplikací: důraz na multimédia a zjednodušování uživatelského rozhraní.

Ale konvergence prokazatelně existuje i druhým směrem, webové aplikace postupně přejímají konvence a prvky desktopových a pomalu se zbavují vazeb na zvyklosti prezentačně orientované prostředí Webu. Dobrou ukázkou je nová verze webového poštovního klienta na Stanfordské univerzitě, spuštěná letos. Ve srovnání s původní verzí (Obr. 5.5), která představuje zástupce prvních komplexních webových aplikací a je většinou prvků konzistentní s hypertextovými stránkami, dobře vyniká rozhraní nové verze (Obr. 5.6), které se od běžného desktopového poštovního klienta liší na

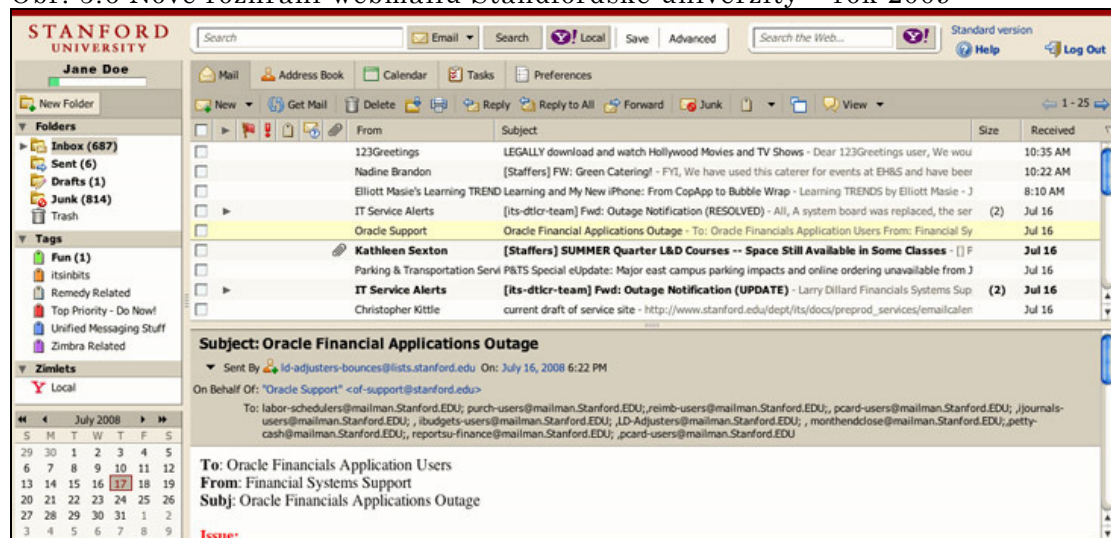
první pohled jen tlačítkem pro odhlášení vpravo nahoře. Obdobně směřuje vývoj mnoha dalších webových aplikací.

Obr. 5.5 Staré rozhraní webmailu Standfordské univerzity – do roku 2009



Zdroj: http://www.stanford.edu/services/emailcalendar/web/w_whatsnew.html

Obr. 5.6 Nové rozhraní webmailu Standfordské univerzity – rok 2009



Zdroj: http://www.stanford.edu/services/emailcalendar/web/w_whatsnew.html

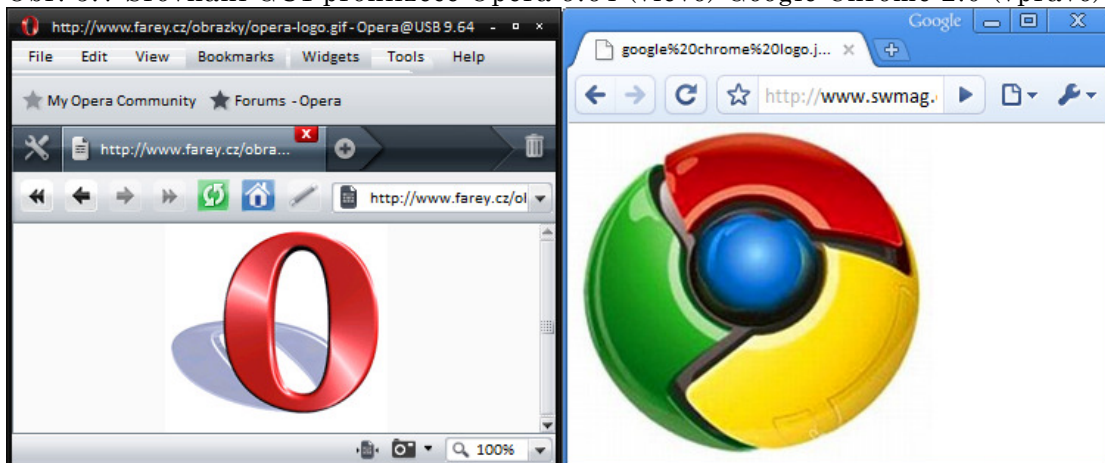
Uživatelská rozhraní komplexních webových aplikací, jako je tato, směřují čím dál více ke konzistenci s konvencemi obdobných aplikací na desktopu a opouštějí přitom zvyklosti obsahově zaměřeného webu. a jak ukázala heuristická analýza aplikace Zoho Writer v kapitole 4.2.3, tento postup není na úkor použitelnosti.

5.2 Posun významu webového prohlížeče

Jak bylo popsáno v kapitole 3, v uživatelských rozhraních webových aplikací dochází ke konfliktu s uživatelským rozhraním prohlížeče, ve kterém je aplikace spuštěna. Na rostoucí množství a význam webových aplikací reaguje i vývoj webových prohlížečů. v roce 2008 bylo nevýraznějším krokem tímto směrem představení prohlížeče *Chrome* společností Google, která k němu uvádí: „*Uvědomili jsme si, že i web se vyvinul z převážně jednoduchých textových stránek směrem k různorodým interaktivním aplikacím a že musíme koncept internetového prohlížeče kompletně přehodnotit ... Většinu lidí samotný prohlížeč nezajímá. Je to pro ně jen nástroj pro spuštění toho důležitého – stránek a aplikací, které tvoří web.*“ [GOOGLE1]

Ale i přes pro Google typický vzletný tón propagačního textu je třeba uznat, že prohlížeč Google Chrome skutečně směřuje k omezení uživatelského rozhraní prohlížeče a dává více prostoru webovým aplikacím. Viditelnou změnou je omezení grafického uživatelského rozhraní na minimum při běžné práci s prohlížečem. Srovnání s běžným bohatým grafickým uživatelským rozhraním prohlížeče Opera je na Obr. 5.7. Rozdíl je v eliminaci množství ovládacích prvků na minimum a díky tomu zvětšené ploše věnované samotnému obsahu.

Obr. 5.7 Srovnání GUI prohlížeče Opera 9.64 (vlevo) Google Chrome 2.0 (vpravo)



Zdroj: Vlastní zpracování autora

Stejným vývojem prochází i populární prohlížeč Windows Internet Explorer. k výraznému zmenšení plochy věnované rozhraní samotného prohlížeče došlo už ve verzi 7 v roce 2006 a k dalšímu snížení počtu viditelných ovládacích prvků došlo ve verzi 8 v roce 2009. Vývojáři tedy zcela evidentně reflektují, že se pozornost uživatele přesouvá od uživatelského rozhraní prohlížeče k samotnému obsahu a uživatelskému rozhraní webových aplikací.

Další významný posun je sice již mimo zaměření této práce na uživatelské rozhraní, ale je natolik závažný, že je třeba ho alespoň okrajově zmínit. Jde o řešení pro offline webové aplikace, což jsou v zásadě klasické webové aplikace schopné pracovat i bez dostupnosti stálého připojení k webovému serveru. Zde se otevírají dvě cesty, buď vznikající specifikace HTML 5, nebo již existující software *Gears* (dříve Google Gears), ale vzhledem k provázanému vývoji obojího je pravděpodobné, že se tyto implementace spojí. v obou případech se ale jedná o systém založený na relačním databázovém systému SQLite na straně klienta a související sadě komponent, které zajišťují zcela transparentní práci s podporovanými webovými aplikacemi offline. [Hassman] Software Gears lze v současné době integrovat s řadou webových prohlížečů

(Firefox 1.5 nebo vyšší, Safari 3.1.1 nebo vyšší, Internet Explorer 6 nebo vyšší a Google Chrome podporuje Gears nativně [GOOGLE2]) a lze jej využít na mnoha webových aplikacích, zejména od společností Google a Zoho, nebo v systému WordPress. Jde tedy o reálnou a do praxe už postupně zaváděnou eliminaci jedné z klíčových omezujících vlastností webových aplikací – potřeby stálého spojení mezi klientským počítačem a serverem.

5.3 Mezistupeň mezi webovou a desktopovou aplikací

5.3.1 Site-specific browser

Významným posunem ve vztahu webových a klasických desktopových aplikací byl vznik konceptu *site-specific browser* (SSB). Site-specific browser, někdy nazývaný také *distraction free browser* (prohlížeč bez vyrušování) je software určený pro vykreslení právě jedné webové stránky či aplikace. Jde vlastně o webový prohlížeč zbavený vlastního uživatelského rozhraní. Výraz *site-specific* nemusí znamenat, že by každá webová stránka (site) vyžadovala proprietární prohlížeč, nýbrž že konkrétní instance prohlížeče se váže pouze na jednu stránku⁶ a není určena pro procházení více stránkami. Proto takový prohlížeč nepotřebuje žádné vlastní navigační prvky. Naopak je zejména z uživatelského hlediska těsněji integrován v operačním systému koncového počítače. [Arrington1]

⁶ První podobná řešení vznikla jako proprietární a byla skutečně vázána na jednu webovou stránku; počínaje rokem 2005 se ale začaly objevovat SSB pro obecné použití, tedy prohlížeče, které mohou být svázány s více libovolnými webovými stránkami nebo aplikacemi. Jejich architektura ale bývá taková, že pro každou takovou stránku mohou mít specifické nastavení, klientské skripty i softwarové doplňky. Proto také „site-specific“

Práce s webovými aplikacemi v prostředí SSB se výrazně blíží běžnému způsobu práce s těmi desktopovými. Uživatel nejprve vytvoří zástupce aplikace, například na ploše (v závislosti na operačním systému. Ve Windows to může být na ploše, v nabídce start nebo v panelu rychlého spuštění. Zástupce je typicky vytvořen funkcí v „běžném“ prohlížeči, ve chvíli kdy je uživatel na dané stránce), potom už může spustit webovou aplikaci jako jakoukoliv desktopovou prostřednictvím zástupce a aplikace se spustí v samostatném okně, které prakticky ničím nepřipomíná běžný webový prohlížeč (Obr. 5.8)

Obr. 5.8 Webová aplikace Pixlr v prostředí Mozilla Prism



Zdroj: pixlr.com, vlastní zpracování autora

Pro úplnost uvedu alespoň nejvýznamnější SSB. k prvenství mezi obecně použitelnými site-specific browsersy se hlásí projekt Bubbles, který byla představen na konci roku 2005, a tehdy byl postaven na vykreslovacím jádře Trident. v roce 2007 Mozilla představila site-specific browser Prism (původně WebRunner), který je rozšiřujícím pluginem prohlížeče Firefox a je shodně s Firefoxem postaven na jádře Gecko. Ve stejném roce začaly vycházet rané verze SSB Fluid, vyvíjeného nativně pro Mac OS X a běžícího na jádře WebKit. v září

2008 k nim přibyl Google Chrome, „běžný“ webový prohlížeč zmíněný v kapitole 5.2, který ale jako první z pěti nejpoužívanějších prohlížečů umí bez jakýchkoliv rozšíření vytvořit pro aktuálně zobrazenou stránku spustitelného zástupce samostatného okna SSB. Proto je také v kapitole 5.2 zdůrazňována jeho orientace na webové aplikace.

Může se zdát, že význam prohlížečů zbavených uživatelského rozhraní není pro konvergenci webových a desktopových aplikací vysoký, ale jak bylo popsáno ve 3. kapitole, hlavním důvodem odlišností uživatelského rozhraní webových aplikací je vazba na běžný webový prohlížeč a na hypertextový formát webových stránek. Odstranění těchto vazeb dává vývojářům možnost upustit od konvencí zavedených na webových stránkách hypertextového typu a přiblížit se více konvencím desktopových aplikací, na které je uživatel při práci se složitějšími aplikacemi navyklý.

5.3.2 Adobe AIR

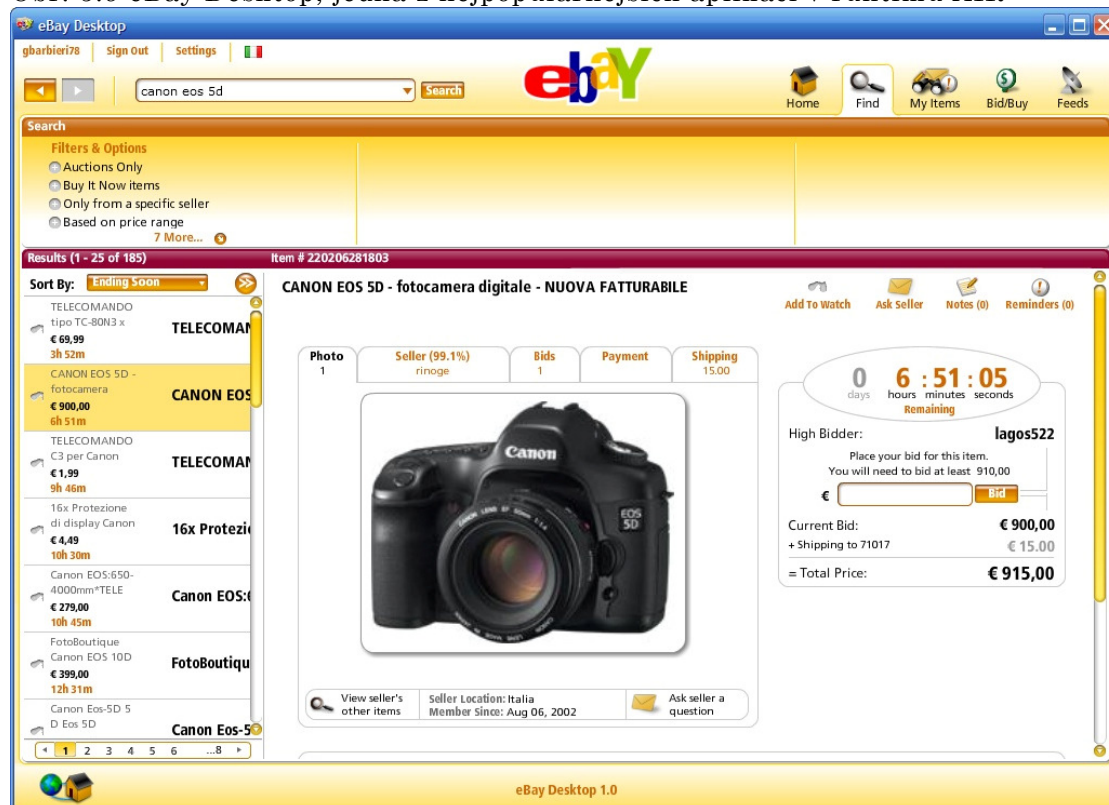
Platforma Adobe AIR (Adobe Integrated Runtime) je v sice některých zdrojích označována za dalšího z představitelů SSB, ovšem AIR jde v přesunu webové aplikace na desktop mnohem dál. Jde v tomto případě o runtime, umožňující běh aplikací vyvinutých typickými webovými technologiemi přímo na koncovém počítači. AIR tedy na rozdíl od SSB, přestože je postaven na vykreslovacím jádře WebKit a pracuje se stejnými technologiemi, není způsobem, jak přistupovat k webovým aplikacím, nýbrž spíše nástrojem po přenesení webových aplikací na desktop.

Konkrétněji umí prostředí Adobe AIR vykonávat kód HTML, JavaScriptu, Flashe a Actionscriptu a je kompatibilní s populárními JavaScript/Ajax frameworky [Chambers], čímž dává na výběr

nejčastěji používané nástroje při tvorbě webových aplikací. Aplikace vytvořené pro AIR ale nemají podobu webové stránky, nejsou to tedy webové aplikace. Naopak jsou ve formě balíků (*.air) instalovány na koncový počítač, kde se chovají zcela jako nativní programy, včetně spouštěcího souboru *.exe na Windows a jeho obdob na Linuxu a Mac OS X. Stejně jako desktopové aplikace mají také přístup do souborového systému počítače a dále platforma nabízí vedle klasického přístupu k databázi na webu i možnosti pro práci s lokálními daty v relačním databázovém systému SQLite. Spolu s dalšími rozšiřujícími knihovnami, například pro práci s okny nebo s příkazovou řádkou, tak AIR umožňuje relativně jednoduchý vznik plnohodnotné desktopové aplikace prostřednictvím škály programovacích jazyků a nástrojů používaných pro vývoj aplikací webových.

Význam této platformy spočívá v tom, že umožňuje vývojářům webových aplikací k fungující webové aplikaci jednoduše a rychle dovyvinout plnohodnotnou desktopovou alternativu, aniž by k tomu bylo potřeba jiných postupů a programovacích jazyků, a tedy jiných programátorů. [Till] Pro otázku konvergence uživatelských rozhraní webových a desktopových aplikací je Adobe AIR velmi podstatný, protože umožňuje snadný vznik dvojic aplikací postavených na stejném základě, ale technicky uzpůsobených alternativně pro běh na webu nebo na desktopu. Kromě některých aspektů vázaných na nadstandardní API runtime AIR, jako je například implementace operací drag & drop, by vzniklé aplikace pro web i desktop a jejich GUI mohly být prakticky identické. Tato technologie tedy má potenciál značně přiblížit uživatelská rozhraní webových a desktopových aplikací.

Obr. 5.9 eBay Desktop, jedna z nejpopulárnějších aplikací v runtime AIR



Zdroj: <http://www.giovy.it/tag/ebay/>

V praxi jsou ale vyvíjeny aplikace pro AIR spíše jako výbavou bohatší alternativy k těm webovým při zachování UI charakteru webové aplikace (např. viz Obr. 5.9), a AIR je využit kvůli výrazně jednodušší implementaci mnoha funkcí. Již existující aplikace pro runtime Adobe AIR zároveň naznačují možný vývoj uživatelského rozhraní desktopových aplikací, protože nemají ve vývojářských nástrojích tak pevně ukotvený vztah k tradičním vzorům používaným při návrhu uživatelského rozhraní desktopové aplikace a přitom umožňují i v nových vzorech zachovat schémata ovládání konzistentní s daným prostředím.

5.4 Predikce budoucího vývoje

Rozbor teoretických východisek, jejich ověření na příkladech aplikací a analýza konvergentních tendencí společně přinášejí použitelné

podklady pro odhad dalšího vývoje vztahu uživatelských rozhraní webových a desktopových aplikací. Podmínky pro budoucí vývoj lze rozdělit do dvou kategorií, a to na předpoklady teoretické a technologické.

5.4.1 Teoretické předpoklady

Teoretickými předpoklady jsou myšleny důvody pro konvergenci obou uživatelských rozhraní. Pokud by ke konvergenci důvod nebyl, pochopitelně by neproběhla. Jak ale ukázaly příklady komplexnějších webových aplikací, složité uživatelské rozhraní je vhodné řešit s ohledy na konvence obdobných desktopových aplikací, na které je uživatel v daném typu aplikace navyklý. Mnohé takové aplikace tedy raději ruší konvence webového prostředí a adaptují standardy desktopové.

Z tohoto všeobecně pozorovatelného trendu jsou výjimkou webové aplikace společnosti Google, které se nesnaží přejímat konvence desktopových aplikací, nýbrž si vytvářejí unikátní styl mezi všemi těmito aplikacemi silně vnitřně konzistentní.

Vývojářům desktopových aplikací zase rozhraní webových aplikací ukázala, že konvence a použitelnost lze udržet i při změně modelu UI nebo třeba s elegantnější grafickou formou. Dá se tak říci, že nová konkurence ve formě efektních a přitom uživatelsky jednoduchých a tedy i efektivních webových aplikací rozhýbala obdobné trendy i v prostředí desktopových aplikací.

Z teoretického hlediska požadavků na uživatelská rozhraní tedy lze očekávat oboustranné sbližování.

5.4.2 Technologické předpoklady

Aby mohly být naplněny teoretické předpoklady, jsou třeba také vhodné technologické nástroje. v tomto bodě jde pouze o webové aplikace, protože u uživatelských rozhraní desktopových aplikací lze v tomto smyslu těžko mluvit o nějakém technologickém omezení. Webová platforma ovšem nebyla vůbec pro tvorbu aplikací koncipována, proto je tvorba aplikací s bohatým uživatelským rozhraním na ní obecně pracnější a implementace výše popsaných problémových ovládacích prvků je stále komplikovaná. v této oblasti v posledních letech pomáhají vývojářům JavaScriptové toolkity (jako je populární *Ext*) a vývojové frameworky, jako *Google Web Toolkit*.

V podkapitole 5.2 byl popsán posun významu webového prohlížeče. Tímto směrem se zřejmě budou webové prohlížeče ubírat i v následujících letech.

Potom tedy už je posledním velkým hendikepem webových aplikací slabá vazba na prostředí operačního systému. v této oblasti dělá krok kupředu zmíněný koncept Site-Specific Browser či implementace klientských relačních databází. Ovšem už koncept SQL databáze na straně klienta vzbuzuje obavy o bezpečnost takového řešení, je tedy velmi nepravděpodobné, že bychom se u webových aplikací v blízké době dočkali ještě hlubší integrace s operačním systémem, jako by byl přístup k souborovému systému klientského počítače, a to právě z důvodu bezpečnosti. Ke vzniku dalších nových konceptů a částečným posunům k integraci zcela jistě dojde, ale někde tady se zdá být v následujících letech technická hranice možností konvergence ze strany webových aplikací.

Jakýkoliv další vývoj lze odhadovat jen do horizontu let 2011-2012, a to především kvůli specifikaci HTML5, od které jsou všeobecně

očekávány zásadní změny v možnostech webových aplikací a v chápání Webu vůbec. Její uvedení bylo plánováno na rok, 2010, nicméně specifikace je zatím ve stádiu pracovního návrhu a vývoj stále nabírá skluz.

6 Závěr

První otázkou stanovenou v úvodu této práce bylo, jaké jsou rozdíly mezi uživatelskými rozhraními webových a desktopových aplikací a čím jsou způsobeny. Analýza v kapitole 3 ukázala, že příčinou je vazba na prostředí, ze kterého dané aplikace vycházejí. U webových aplikací jde o původ ve webovém obsahu pasivního, prezentačního charakteru a dále o omezení uživatelského rozhraní ze strany webového prohlížeče, ve kterém běží. U desktopových aplikací je zase historicky zakotven a uživatelem vyžadován silně konzistentní charakter uživatelských rozhraní na dané platformě. Praktická část práce výstupy této analýzy potvrdila, nicméně ukázala, že u webových aplikací není vazba na prostředí tak pevná, jak bylo předpokládáno. Na Webu je tedy uživatel přizpůsobivější, což je dáno výraznou diverzitou tohoto prostředí. Naopak v prostředí desktopových aplikací se ukázalo dodržování zažitých konvencí platformy jako klíčový požadavek.

Druhým úkolem bylo na případech konkrétních aplikací zjistit, jaký mají popisované rozdíly dopad na použitelnost. Heuristická analýza použitelnosti dvou textových procesorů neprokázala rozdíl v použitelnosti, jiná ovšem byla situace u komunikačních klientů, kde analýza ukázala vyšší použitelnost uživatelského rozhraní webové aplikace oproti desktopové. Poznatkem v obou kategoriích aplikací byl nižší rozsah funkcionality ve webových aplikacích. Výsledkem přímého porovnání a heuristické analýzy použitelnosti konkrétních vybraných aplikací tedy je závěr, že ty webové mají užší nabídku funkcí, možností nastavení i customizace, ale použitelnost je v těchto případech stejná nebo dokonce vyšší, než u obdobného desktopového SW.

Posledním stanoveným úkolem bylo prozkoumat, do jaké míry se uživatelská rozhraní webových a desktopových aplikací vzájemně ovlivňují, a dále se na základě dosavadního vývoje stanovit předpoklad dalšího směřování jejich vztahu.

Na konkrétních příkladech změn uživatelských rozhraní v čase bylo demonstrováno, že nová konkurence ve formě graficky efektních webových aplikací rozhýbala obdobné trendy i v prostředí desktopových aplikací. U těch je dalším sbližujícím trendem i zjednodušování uživatelských rozhraní. Naopak u webových aplikací se ukázalo, že v případě komplexních aplikací je vhodnější následovat zažité konvence uživatelských rozhraní obdobných desktopových aplikací, než dodržovat konvence webové. Ty totiž nejsou kvůli diverzifikovanému prostředí Webu natolik pevně zakotvené, aby jejich porušení snižovalo použitelnost. Uživatelské prostředí webových aplikací, zejména těch složitějších, se tedy bude desktopovým přibližovat.

Přibližování uživatelských rozhraní webových aplikací desktopovým budou pomáhat i technologické inovace, nicméně ty zřejmě již brzy narazí v tomto posouvání možností webových aplikací na omezení především bezpečnostního charakteru. V blízkých letech tedy je možné očekávat sblížení, nikoliv však naprosté prolnutí uživatelských rozhraní webových a desktopových aplikací. Webová aplikace si zachová určitý stupeň izolace od operačního systému koncového počítače a především tím se bude od desktopové odlišovat.

Zdroje

Publikace

[Chambers] CHAMBERS, Mike, et al. *Adobe Integrated Runtime (AIR) for JavaScript Developers*. 1st edition. [s.l.] : Adobe Dev Library, 2007. 174 s. ISBN 0596515197.

[IEEE 90] IEEE. *IEEE Standard Glossary of Software Engineering Terminology/IEEE Std 610.12-1990*. 1st edition. [s.l.] : Institute of Electrical & Electronics Engineer, 1991. 96 s. ISBN 155937067X.

[Nielsen1] NIELSEN, Jakob. *Usability Engineering*. 1st edition. [s.l.] : Morgan Kaufmann, 1993. 362 s. ISBN 0125184069.

[Powell] POWELL, Thomas. *Ajax: The Complete Reference*. 1st edition. [s.l.] : McGraw-Hill Osborne Media, 2008. 654 s. ISBN 007149216X.

[Proctor] PROCTOR, Robert V., VU, Khim-Phuong L. *The Handbook of Human Factors in Web Design*. [s.l.] : CRC, 2004. 208 s. ISBN 0805846123.

[Tidwell] TIDWELL, Jenifer. *Designing Interfaces: Patterns for Effective Interaction Design*. [s.l.] : O'Reilly Media, Inc., 2005. 352 s. ISBN 0596008031.

[Basl] BASL, Josef. *Podnikové informační systémy: podnik v informační společnosti*. Praha : Grada, 2002. 142 s. ISBN 80-247-0214-2.

[Carey] CAREY, Jane M. *Human Factors in Information Systems : The Relationship Between User Interface Design and Human Performance (Human Computer Interaction)*. 4th edition. [s.l.] : Ablex Publishing Corporation, 1997. 254 s. ISBN 1567502857.

[Langer] LANGER, Arthur M. *Analysis and Design of Information Systems*. 3rd edition. [s.l.] : Springer, 2007. 418 s. ISBN 1846286549.

[Nuray1] NURAY, Aykin. *Usability and Internationalization of Information Technology*. 1st edition. [s.l.] : CRC, 2004. 392 s. ISBN 0805844783.

[Nuray2] NURAY, Aykin. *Usability and Internationalization. Global and Local User Interfaces*. 1st edition. [s.l.] : Springer, 2007. 576 s. ISBN 3540732888.

[Smith] SMITH, Michael J., et al. *Usability Evaluation and Interface Design : Cognitive Engineering, Intelligent Agents, and Virtual Reality, Volume 1*. 1st edition. [s.l.] : CRC, 2001. 1592 s. ISBN 0805836071.

Internetové zdroje

- [Arrington1] ARRINGTON, Michael. Bridging Desktop And Web Applications - a Look At Mozilla Prism. *TechCrunch* [online]. 2008 [cit. 2009-06-13]. Dostupný z WWW: <<http://www.techcrunch.com/2008/03/22/bridging-desktop-and-web-applications-a-look-at-mozilla-prism/>>.
- [GOOGLE1] Google : *Google Chrome* [online]. 2009 [cit. 2009-06-12]. Dostupný z WWW: <<http://www.google.com/chrome/intl/cs/why.html?brand=CHMB>>.
- [GOOGLE2] Google Code : *Gears API - Gears FAQ* [online]. 2009 [cit. 2009-06-13]. Dostupný z WWW: <http://code.google.com/intl/cs/apis/gears/gears_faq.html#supportedSystems>.
- [Hassman] HASSMAN, Martn. SQL si razí cestu do HTML5. *Lupa* [online]. 2007 [cit. 2009-03-17]. Dostupný z WWW: <<http://www.lupa.cz/clanky/sql-si-razi-cestu-do-html5/>>.
- [Howe] HOWE, Denis. User interface. *The Free On-line Dictionary of Computing* [online]. 2009 [cit. 2009-05-28]. Dostupný z WWW: <[Dictionary.com http://dictionary.reference.com/browse/user interface](http://dictionary.reference.com/browse/user+interface)>.
- [Nielsen2] NIELSEN, Jakob. How to Conduct a Heuristic Evaluation. *Useit.com* [online]. 1994 [cit. 2009-05-25]. Dostupný z WWW: <http://www.useit.com/papers/heuristic/heuristic_evaluation.html>.
- [Nielsen3] NIELSEN, Jakob. Ten Usability Heuristics. *Useit.com* [online]. 1994 [cit. 2009-05-25]. Dostupný z WWW: <http://www.useit.com/papers/heuristic/heuristic_list.html>.
- [Till] TILL, Michal. Svěží vzduch pro vývoj: Adobe AIR. *Connect* [online]. 2009 [cit. 2009-06-16]. Dostupný z WWW: <<http://connect.zive.cz/jaky-je-adobe-air>>.
- [Antoš] ANTOŠ, David. Webové kancelářské balíky, aneb sbohem desktope? (1/3). *Lupa* [online]. 2006 [cit. 2009-03-15]. Dostupný z WWW: <<http://www.lupa.cz/clanky/webove-kancelarske-baliky-aneb-sbohem-desktope/>>.
- [Arrington2] ARRINGTON, Michael. Bridging Desktop And Web Applications, Part 2. *TechCrunch* [online]. 2008 [cit. 2009-06-12]. Dostupný z WWW: <<http://www.techcrunch.com/2008/04/07/bridging-desktop-and-web-applications-part-2/>>.
- [Nielsen4] NIELSEN, Jakob. Heuristic Evaluation. *Useit.com* [online]. 1994 [cit. 2009-05-26]. Dostupný z WWW: <<http://www.useit.com/papers/heuristic/>>..
- [Tillman] TILLMAN, Jean. User Interface Design for Web Applications. *Digital Web Magazine* [online]. 2003 [cit. 2009-06-13]. Dostupný z WWW: <http://www.digital-web.com/articles/user_interface_design_for_web_applications/>.

Seznam aplikací

AVG

<http://www.grisoft.cz/>

Azureus/Vuze

<http://www.vuze.com/>

eBay Desktop

<http://desktop.ebay.com/>

Fontstructor

<http://fontstruct.fontshop.com/>

Google Chrome

<http://www.google.com/chrome/>

meebo

<http://www.meebo.com/>

Microsoft Office Word

<http://www.microsoft.com/cze/office/programs/word/>

Mozilla Firefox

<http://www.mozilla.com/firefox/>

Mozilla Prism

<https://labs.mozilla.com/projects/prism/>

OpenOffice.org Writer

<http://openoffice.org/>

Opera

<http://www.opera.com/>

Pidgin

<http://www.pidgin.im/>

pixlr

<http://pixlr.com/>

Stanford Email & Calendar Web client

<http://www.stanford.edu/services/emailcalendar/web/>

Zoho Writer

<http://writer.zoho.com/>

Seznam obrázků

Obr. 5.1 Fyzická struktura grafického uživatelského rozhraní.....	20
Obr. 5.2 Fyzická struktura na bázi virtuálních oken v Aplikaci FontStruct	21
Obr. 6.1 Procento problémů odhalených heuristickou analýzou	25
Obr. 6.2 GUI desktopového textového procesoru OpenOffice.org Writer 3.....	30
Obr. 6.3 GUI webového textového procesoru Zoho Writer.....	32
Obr. 6.4 Rozdílný ribbon v aplikaci Microsoft Word a Zoho Writer	34
Obr. 6.5 GUI internetového komunikačního klienta Pidgin.....	38
Obr. 6.6 GUI webového instant messengeru meebo	40
Obr. 6.7 GUI webového instant messengeru meebo se samostatnými okny.....	41
Obr. 7.1 Rozhraní aplikace AVG 7.0, rok 2004	45
Obr. 7.2 Rozhraní aplikace AVG 8.5, rok 2009	45
Obr. 7.3 Rozhraní aplikace Azureus 2.2.0.0, rok 2004.....	46
Obr. 7.4 Rozhraní aplikace Azureus Vuze 4.2.0.3 Beta 17, rok 2009.....	47
Obr. 7.5 Staré rozhraní webmailu Standfordské univerzity – do roku 2009	48
Obr. 7.6 Nové rozhraní webmailu Standfordské univerzity – rok 2009.....	48
Obr. 7.7 Srovnání GUI prohlížeče Opera 9.64 a Google Chrome 2.0.....	50
Obr. 7.8 Webová aplikace Pixlr v prostředí Mozilla Prism.....	52
Obr. 7.9 eBay Desktop, jedna z nejpopulárnějších aplikací v runtimu AIR.....	55

Seznam příloh

Příloha 1

Zápis z heuristické analýzy použitelnosti aplikace OpenOffice.org Writer 66

Příloha 2

Zápis z heuristické analýzy použitelnosti aplikace Zoho Writer 66

Příloha 3

Zápis z heuristické analýzy použitelnosti aplikace Pidgin 67

Příloha 4

Zápis z heuristické analýzy použitelnosti aplikace meebo 67

Přílohy

Příloha 1

Zápis z heuristické analýzy použitelnosti aplikace OpenOffice.org Writer

OpenOffice.org Writer		
Problém v použitelnosti	Heuristika	Priorita
Automatické opravy upravují slova na nezamýšlené výrazy a není nabídnuta cesta zpět	3	2
Na první pohled nepřehledný panel nástrojů	8	2
Tlačítko Help umístěné v některých dialogích v rozporu se zvyklostmi	4	3
Nezvykle umístěná tlačítka Zpět a Opakovat	4	3
Tlačítko Back v dialogu Tools -> Options vyvolává neočekávanou akci	4	3
Při startu aplikace velmi dlouhé čekání na splash screen bez indikace stavu	1	3
Nedostatečně intuitivní ikony v panelu nástrojů, navíc bez členění	6	3
Příliš rychlé scrollování při výběru textu	5	3
Nezobrazená mezera na konci řádku	1	3

Zdroj: Vlastní heuristická analýza použitelnosti

Příloha 2 Zápis z heuristické analýzy použitelnosti aplikace Zoho Writer

Zoho Writer		
Problém v použitelnosti	Heuristika	Priorita
V některých situacích se chová tlačítko Undo neočekávaně. Někdy provede zpět více kroků.	3	2
Chybí indikace zapnutí či vypnutí spellcheckeru a indikace průběhu kontroly.	1	2
Obecně podivné chování spellcheckeru v průběhu kontroly	3	2
Mnoho funkcí nefunguje při vybrání více prvků, vadí zejména v tabulkách	4	2
Nefunguje Copy & paste obrázků z jiných aplikací	4	3

Chování tlačítka volby jazyka. Není konzistentní se zbytkem aplikace	4	3
Jen základní klávesové zkratky	7	3

Zdroj: Vlastní heuristická analýza použitelnosti

Příloha 3 Zápis z heuristické analýzy použitelnosti aplikace Pidgin

Pidgin		
Problém v použitelnosti	Heuristika	Priorita
Tlačítka v dialogových oknech prohozená oproti zavedenému zvyklostem	4	2
Chybí zobrazení stavu jednotlivých účtů při využití více protokolů	1	2
Nekonzistence klávesových zkratk s běžnými IM klienty	4	2
Chybí customizace klávesových zkratk	7	3
Nelze nastavit výchozí jazyk pro spellchecking	5	3
Souborové dialogy z toolkitu GDK nejsou konzistentní s nativními aplikacemi	4	3
V menu jsou ikony zhruba u poloviny položek; menu nepřehledné	8	3

Zdroj: Vlastní heuristická analýza použitelnosti

Příloha 4 Zápis z heuristické analýzy použitelnosti aplikace meebo

meebo		
Problém v použitelnosti	Heuristika	Priorita
Nekonzistentní chování pravého tlačítka myši, v contact listu patří webové aplikaci, jinde prohlížeči	4	2
Nefunguje Copy & paste ani Drag & drop souborů	4	3
Vnucující se prvky (reklama, Meebo Blog atd.)	8	3

Zdroj: Vlastní heuristická analýza použitelnosti

