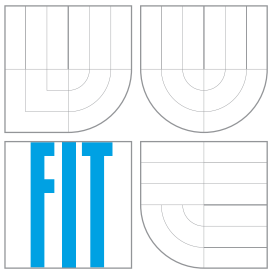
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

NÁSTROJ PRO ANALÝZU PSANÍ UŽIVATELE NA KLÁVESNICI JAKO ADD-IN DO MS OFFICE

A TOOL FOR ANALYSIS OF USER'S TYPING SKILLS AS AN ADD-IN TO MS OFFICE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JAKUB DANĚK
AUTHOR

VEDOUCÍ PRÁCE — Doc. Ing. ADAM HEROUT, Ph.D.
SUPERVISOR

BRNO 2012

Abstrakt

Tato práce popisuje návrh a implementaci add-inu do MS Word a MS Outlook, jehož účelem je analyzovat psaní uživatele. Zabývá se metodami analýzy rychlosti a chybovosti psaní a jejich použitím pro posouzení schopnosti psaní uživatele. Součástí implementace je i grafické rozhraní doplňku, zobrazující analyzované údaje v podobě grafu.

Abstract

The thesis describes a project and implementation of MS Word and MS Outlook add-in purposed on analysis of user's typing. It focuses on methods of analysis of typing frequency and typing errors and their usage for consideration of typing abilities of users. User's interface of the add-in is a component of the implementation, showing analysed data in a graph.

Klíčová slova

analýza editace textu, analýza chybovosti psaní, MS Office, MS Word, MS Outlook, doplněk, add-in, klávesnice, detekce stisků kláves, rychlost psaní

Keywords

text editing analysis, typing error analysis, MS Office, MS Word, MS Outlook, Add-in, keyboard, keyboard keypress detection, typing frequency

Citace

Jakub Daněk: Nástroj pro analýzu psaní uživatele na klávesnici jako add-in do MS Office, bakalářská práce, Brno, FIT VUT v Brně, 2012

Nástroj pro analýzu psaní uživatele na klávesnici jako add-in do MS Office

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Doc. Ing. Adama Herouta, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Jakub Daněk
8. května 2012

Poděkování

Chtěl bych poděkovat panu Doc. Ing. Adamovi Heroutovi, Ph.D za vedení této práce a za jeho cenné rady a návrhy.

Obsah

Kapitola 1

Úvod

Rychlost a způsob psaní jsou podstatné vlastnosti popisující činnost často vykonávanou různými uživateli počítačů. Při psaní textu na počítači je možno získat velké množství údajů o psaní a analýzou těchto údajů pak umožnit uživateli své psaní zlepšit. Pokud tyto údaje budeme získávat bez nutnosti uživatele spouštět speciální aplikace pro trénování psaní, uživatel bude moci pohodlně analyzovat a zlepšovat své psaní. V této práci se pokusíme navrhnout a realizovat add-in (doplněk) do aplikací MS Office a MS Word, verzí 2007 a 2010, jehož účelem bude analyzovat psaní uživatele a následné analyzované údaje přehledně uživateli zobrazit. Zobrazení těchto údajů bude provedeno tak, aby bylo možné porovnávat tyto údaje v průběhu času a bylo možné zároveň pozorovat chyby v psaní. Z těchto údajů pak uživatel bude schopen posoudit reálně vlastní psaní a případně ho i vylepšit.

Tato práce je rozdělena do tří částí. V první části jsou uvedeny metody v současné době používané pro různé části analýzy psaní a jejich možné využití v naší práci. V druhé části jsou podrobně popsány metody použité v této práci a zdůvodnění jejich použití. Ve třetí a poslední části je popis implementace a testování vytvořených doplňků, včetně popisu vytvořeného uživatelského rozhraní a jeho testování. Součástí této části je i obecný popis vytváření nových doplňků a testování správného měření údajů.

Kapitola 2

Současné metody pro detekci stisků kláves a analýzu psaní

Tato kapitola se bude zabývat metodami pro detekci a analýzu psaní uživatele na klávesnici. Popíšeme si zde jednotlivé současné metody a jejich možné uplatnění pro naše potřeby.

2.1 Detekce stisků kláves

Tento oddíl popíše jednotlivé možnosti detekce stisků kláves a to jak metody závislé na operačním systému, tak metody nezávislé.

2.1.1 Obecné metody detekce stisků kláves

Zde si uvedeme používané metody pro detekci stisků kláves, které nejsou závislé na použitém operačním systému. Tyto metody se dělí na dva hlavní typy - hardwarové a softwarové metody detekce stisků kláves.

Hardwarové metody

Hardwarové metody pro detekci stisků kláves [**?**] pracují bez závislosti na softwaru počítače a jsou počítačem nezjistitelné. Dělí se na dva základní typy:

1. Zařízení, která se napojí na přenos dat mezi klávesnicí a počítačem.
2. Zařízení pro zachytávání záření/vlnění produkovaného klávesnicí.

Metoda č. 1 má výhodu přesnosti odchylky menší než 1 *ms*, což je dostatečná přesnost pro jakoukoliv analýzu rychlosti psaní. Tato metoda má také absolutní přesnost detekce a rozpoznání stisknutých kláves. Metoda č. 2 je využívána pro dálkovou detekci stisknutých kláves a na rozdíl od metody č. 1 nemá absolutní přesnost detekce a rozpoznání stisknutých kláves.

Všechny hardwarové metody mají nevýhodu obtížnosti napojení na software počítače, kde by probíhala analýza rozpoznaných kláves, a nutnosti zapojení samostatného zařízení.

Softwarové metody

Softwarové metody pro detekci stisků kláves [**?**] jsou součástí softwaru počítače, ve kterém detekují stisky kláves. Dělí se na tyto základní typy:

1. Software pracující na úrovni jádra, často jako ovladač zařízení.
2. Software závislý na funkcích operačního systému. Bude rozepsán v kapitole 2.1.2.
3. Software pracující jako virtuální stroj [**?**].

Výhodou softwarových metod detekce stisku kláves je jejich jednoduché napojení pro další zpracování analýzou v počítači a snadná instalace, nevýhodou však je potřeba spolupráce antivirových programů pro umožnění detekce stisků kláves. Díky těmto vlastnostem je výhodnější pro potřeby tohoto projektu využít některou ze softwarových metod.

2.1.2 Metody specifické pro MS Windows

Všechny operační systémy Microsoft Windows mají podporu funkcí *Windows API* [**?**]. Windows API obsahuje různé funkce pro práci programů v operačním systému a mezi jinými obsahuje i několik funkcí, které jsou využitelné pro detekci stisknutých kláves.

Pasivní zjištění stiknutých kláves pomocí funkcí z Windows API

Windows API obsahuje funkci *SetWindowsHookEx* [**?**], která umožňuje vložit funkci pro zachytávání jednotlivých zpráv zasílaných konkrétní aplikací, nebo celému systému. Obsahem těchto zpráv jsou informace o událostech, které se odehrály, mimo jiné i o stisku kláves. SetWindowsHookEx umožňuje vložit dva různé typy funkcí pro předání zpráv o stisku kláves:

1. Funkce pro předání zpráv o stisku kláves, které jsou určeny pro konkrétní aplikaci.
2. Funkce pro předání zpráv o stisku kláves, které jsou doručeny operačnímu systému a jsou určeny pro libovolnou aplikaci.

Výhodou funkce typu 1 je, že zprávu o stisku kláves dostane ještě předtím, než je jakkoliv zpracována, což zajišťuje lepší přesnost měření času stisku kláves. Tímto typem se však na rozdíl od druhého typu zachytí všechny stisky kláves, ne pouze ty, které jsou určeny aplikaci, kterou sledujeme. Pokud je tedy tento typ využit, je nutné zajistit filtrování zpráv o stisku kláves, které nejsou určeny k analýze. Pokud je využita funkce typu 2, je nutno zajistit její dostatečnou přesnost.

Aktivní zjištění stiknutých kláves pomocí funkcí z Windows API

Windows API obsahuje několik funkcí pro zjištění stavu jak jednotlivých virtuálních[1] kláves, tak i celé klávesnice [**?**]. Aktivita těchto funkcí spočívá v tom, že zjišťují jednu z uvedených informací:

- Stav kláves(y) v době dotazu funkce. Například funkce *GetKeyState*.
- Stav klávesy od posledního dotazu na její stav. Například funkce *GetAsyncKeyState*.

Obě tyto možnosti mají nevýhodu v tom, že zjišťují stav kláves až v okamžiku volání funkcí. To zapříčiní, že je nutno volat uvedené funkce v nekonečné smyčce, aby byla zajištěna co největší přesnost měření času stisku kláves. Tento postup však generuje velké množství aktivity procesoru a může způsobit zpomalení práce počítače.

Zjištění stiknutých kláves pomocí úpravy jádra OS

Úprava jádra OS je většinou uskutečněna ve formě ovladače klávesnice, který tak získává informace o stisku kláves v nejkratším možném čase od jejich fyzického stisknutí. Nevýhodou této metody je však její náročnost na realizaci a riziko takového zásahu do jádra OS, který způsobí nefunkčnost celého systému.

[1] Kláves přeložených operačním systémem z hardwarového kódu předaného klávesnicí na obecný kód kláves nezávislý na hardwaru (číslo v rozsahu 0 – 255).

2.2 Analýza rychlosti psaní

V současné době existují dvě používané metody analýzy rychlosti psaní – počítání počtu slov nebo úhozů za minutu a analýza psaní jednotlivými prsty. Popíšeme si zde obě tyto metody a jejich možný přínos pro naši práci.

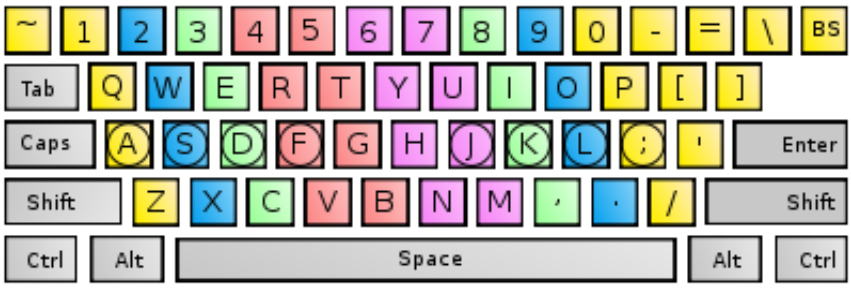
2.2.1 Počet slov za minutu a počet úhozů za minutu

Počet slov za minutu – *wpm* [**?**] je obecně užívaná jednotka pro rychlost psaní uživatele na klávesnici. Jedním slovem zde rozumíme pět znaků. V současné době je toto hlavní způsob určení rychlosti psaní v některých zemích.

Další možností je zjištění počtu úhozů za minutu. *Úhozem* budeme rozumět jeden stisk klávesy. Tato jednotka je používaná a uznávaná v České republice [**?**] a na rozdíl od počtu slov za minutu udává rychlost prstů uživatele i pro vkládání textu pomocí například klávesových zkratek.

2.2.2 Analýza psaní jednotlivými prsty

Při využití hmatové metody ovládání klávesnice všemi deseti prsty [**?**] je možno odhadnout, kterou klávesu tiskne který prst. Následující obrázek udává ideální rozložení jednotlivých prstů pro jednotlivé klávesy v rozložení *QWERTY*. Každý prst má přiřazenou barvu a výchozí pozice prstů pro psaní je zaznačena kolečky.



Obrázek 2.1: Rozložení prstů na klávesnici pro psaní.

Zdroj: http://en.wikipedia.org/wiki/File:Touch_typing.svg

Palce jsou v tomto rozložení umístěné na klávese mezerník. Rozložení však udává pouze ideální použití kláves jednotlivými prsty a to pouze pro využití uvedené metody psaní se standardní klávesnicí. Pokud použijeme jinou metodu či klávesnici, data získaná z této analýzy budou chybná. Analýza jednotlivých prstů či kláves nám však umožňuje získat podrobnější informace o tom, jak se liší rychlost jednotlivých prstů či kláves u uživatele.

2.3 Metrika pro analýzu editace psaní předem známého textu

V tomto oddíle si popíšeme metriku pro detekci a analýzu editace textu [**?**], který je předem znám a úkolem uživatele je tento text přepsat. Tato metrika je schopna detekovat různé typy chyb pro psaní a zároveň zjistit závažnost těchto chyb. Na konci oddílu si na příkladu ukážeme použití této metriky a její výhody.

2.3.1 Popis metriky

K přesnému vyjádření míry editace textu využijeme tzv. *New Unified Error Metric* [**?**]. Je to metrika, která se pokouší obecně popsat používané metody analýzy editace textu a chybovosti psaní. Tato metrika vychází z předpokladu, že uživatel má vstupní text k přepsání, který je metrice znám a tudíž je možno zjistit odchylky přepsaného textu od toho k přepsání. Metrika pracuje pouze se stisky kláves a jejich vzájemným pořadím, přičemž ignoruje klávesy, které mění význam následných stisků kláves, například klávesa *Shift*. Výpočet metriky si ukážeme na příkladu.

Text k přepsání:	*Vzorový text k přepsání.*
Vstup uživatele na klávesnici:	*Vzorová*$\Leftarrow\leftarrow\leftarrow\Leftarrow\rightarrow\rightarrow$ *texta* $k\Leftarrow\leftarrow\leftarrow\Leftarrow\rightarrow\rightarrow$ *přepsání.*
Přepsaný text:	*Vzorová text k přepsání.*

Vysvětlivky:
- $\leftarrow$ Klávesa šipka doleva.
- $\rightarrow$ Klávesa šipka doprava.
- $\Leftarrow$ Klávesa backspace.

Pokud porovnáme text k přepsání s přepsaným textem, všimneme si tučně uvedené chyby v prvním přepsaném slově. Toto chybně přepsané písmeno označíme jako ***Incorrect Not Fixed***, dále jen ***INF***. Pokud porovnáme i vstup uživatele na klávesnici s přepsaným textem, všimneme si, že uživatel provedl při přepisu textu několik úkonů, které nebyly potřeba - napsal chybný znak a později ho opravil. Tato chyba v přepisu se nepromítla do výsledného textu, ale pro analýzu je přesto důležitá. Chybně napsaný znak, který byl později opraven, označíme jako ***Incorrect Fixed***, dále jen ***IF***. Vstup uživatele, který vedl k opravě jeho chyby v přepisu – znaky šipky doleva a znak klávesy backspace, označíme jako ***Fixes***, dále jen ***F***. Veškeré znaky, které uživatel napsal správně a jejichž úprava nebyla nutná pro shodu přepsaného textu s textem k přepsání, označíme jako ***Correct***, dále jen ***C***.

Obecně můžeme definovat jednotlivá označení takto:

- $IF =$ Znaky, které byly přepsány nesprávně a byly později opraveny.
- $INF =$ Znaky, které byly přepsány nesprávně a nebyly později opraveny, nebo znaky, které nebyly přepsány a měly být.
- $F =$ Stisky kláves, které vedly k opravě špatně napsaných znaků, nebo které pohybovaly textovým kurzorem.
- $C =$ Znaky, které byly přepsány správně.

Dle použité metriky nám tedy v uvedeném příkladu vyšly níže uvedené počty pro jednotlivá označení.

$$\begin{aligned} IF &= 1 \\ INF &= 1 \\ F &= 5 \\ C &= 23 \end{aligned}$$

Dále můžeme blíže popsat některé vlastnosti použité metriky:

1. Součet hodnot s označením C a INF udává počet veškerých znaků v přepsaném textu. Pro jejich odlišení je potřeba znát vstupní bezchybný text.
2. Stisky kláves s označením IF lze rozpoznat jednoduše tak, že to jsou takové stisky kláves náležící písmenům, které jsou ve vstupu uživatele, ale ne v přepsaném textu.

2.3.2 Vzorce pro analýzu textu popsané metrikou

Výsledky získané výše popsanou metrikou můžeme analyzovat pomocí několika vzorců. Tyto vzorce si teď uvedeme spolu s vysvětlením jejich využití. Od této části dále budeme pro jednoduchost označovat C, INF, IF a F jako počty prvků jednotlivých typů.

MSD chybovost

Následující vzorec využívá algoritmu pro výpočet hodnoty *MSD* [**?**] (*Minimum String Distance*). *MSD* je hodnota, která udává minimální počet operací pro převod jednoho řetězce na jiný. K tomuto převodu je možné využít tří druhů operací.

- Vložení.
- Smazání.
- Nahrazení.

Každá z těchto operací pracuje pouze s jedním znakem a operace lze libovolně řetězit. Uvažujme následující řetězce.

Řetězec č. 1: *abcd.*
Řetězec č. 2: *acbd.*

Pro převod z řetězce č.1 je hodnota $MSD = 2$. Akce nutné k převodu řetězce s $MSD = 2$ jsou tyto.

- Odstraň znak c a vlož znak c za znak b.
- Vlož znak b za znak a a odstraň druhý znak b.
- Zaměň znak b za znak c a zaměň znak c za druhý znak b.

Operací převodu může být tedy několik se stejným minimálním počtem akcí, ale hodnota *MSD* se nemění.

Vzorec pro výpočet *MSD chybovosti* je uveden zde.

$$MSD\ chybovost = \frac{MSD(P,T)}{C + INF} \times 100\% \tag{2.1}$$

Hodnoty P a T označují řetězce pro porovnání, v našem případě řetězec bezchybný a řetězec přepsaný. Hodnota $C + INF$ označuje veškerý přepsaný text.

Tento vzorec nám v procentech udává míru chybovosti přepsaného textu, ale dle zadaného vzorce je možno tuto hodnotu měřit až nad celými řetězci, ne dle jednotlivých znaků. Využijeme tedy metriky popsané v kapitole 2.3.1 a hodnotu *MSD chybovosti* budeme počítat pomocí ekvivalentního vzorce.

$$MSD\ chybovost = \frac{INF}{C + INF} \times 100\% \tag{2.2}$$

Key Strokes Per Character – KSPC

Hodnota *KSPC* [**?**] nám udává poměr počtu stisků kláves k přepsaným písmenům. Vyjadřuje nám poměr opravených chyb k přepsanému textu a tím nám doplňuje výsledky *MSD chybovosti*, které nám uvádí údaj o neopravených chybách. Jednoduchý vzorec pro výpočet této hodnoty:

$$KSPC = \frac{|Vstup\ uživatele|}{|Přepsaný\ text|}. \tag{2.3}$$

Toto vyjádření je velice triviální, ale pro naše potřeby si přesto vyjádříme tuto hodnotu i v metrice popsané v kapitole 2.3.1.

$$KSPC = \frac{C + INF + F}{INF + C} \tag{2.4}$$

Účinnost oprav chyb

Účinností oprav chyb [**?**] budeme rozumět počet akcí, které v průměru potřebujeme na opravu jedné chyby. Tento ukazatel nám umožňuje podrobněji analyzovat dopad chyb na rychlost psaní textu.

$$Účinnost\ oprav\ chyb = \frac{IF}{F} \tag{2.5}$$

Svědomitost uživatele

Svědomitostí uživatele [**?**] budeme rozumět poměr opravených chyb k celkovému množství chyb.

$$Svědomitost\ uživatele = \frac{IF}{IF + INF} \tag{2.6}$$

Nevyužité pásmo

Nevyužitým pásmem [**?**] budeme rozumět poměr množství stisků kláves, které neměly za důsledek napsání správného písmene a všech kláves. Tato hodnota nám udává celkovou účinnost psaní.

$$Nevyužité\ pásmo = \frac{INF + IF + F}{C + INF + IF + F} \tag{2.7}$$

Využité pásmo

Využitým pásmem [**?**] budeme rozumět poměr množství stisků kláves, které měly za důsledek napsání správného písmene a všech kláves. Tato hodnota nám udává celkovou účinnost psaní.

$$Využité\ pásmo = \frac{C}{C + INF + IF + F} \tag{2.8}$$

Chybovost neopravených znaků

Chybovostí neopravených znaků [**?**] budeme rozumět poměr počtu chybně napsaných neopravených znaků ke všem napsaným znakům, i chybných a následně opravených.

$$\textit{Chybovost neopravených znaků} = \frac{INF}{C + INF + IF} \times 100\% \tag{2.9}$$

Chybovost opravených znaků

Chybovostí opravených znaků [**?**] budeme rozumět poměr počtu chybně napsaných a opravených znaků ke všem napsaným znakům.

$$\textit{Chybovost opravených znaků} = \frac{IF}{C + INF + IF} \times 100\% \tag{2.10}$$

Celková chybovost

Celkovou chybovostí [**?**] budeme rozumět poměr počtu chybně napsaných znaků ke všem napsaným znakům. Tato hodnota je součtem *chybovostí opravených znaků* a *chybovostí neopravených znaků*.

$$\textit{Celková chybovost} = \frac{INF + IF}{C + INF + IF} \times 100\% \tag{2.11}$$

2.3.3 Ukázka výpočtu hodnot dle uvedených vzorců

Na dvou příkladech si zde ukážeme hodnoty popsané metriky a jak se liší v závislosti na vstupu uživatele. V obou příkladech použijeme řetězec k přepsání a řetězec přepsaný, které jsme použili už v kapitole 2.3. Příklady se budou lišit pouze ve vstupech uživatele na klávesnici.

Text k přepsání:	*Vzorový text k přepsání.*
Vstup uživatele č. 1 na klávesnici:	*Vzorová* ***texta*** *k←←⇐→→ přepsání.*
Vstup uživatele č. 2 na klávesnici:	*Vzorová* ***texta⇐*** *k přepsání.*
Přepsaný text:	*Vzorová text k přepsání.*

Vysvětlivky:
- ← Klávesa šipka doleva.
- → Klávesa šipka doprava.
- ⇐ Klávesa backspace.

V tomto příkladu jsme udělali řadu chyb a všechny kromě jedné jsme i opravili. Jednotlivé základní hodnoty a výsledné hodnoty pro všechny vzorce metriky jsou uvedeny níže. Zvýrazněné hodnoty metriky nám ukazují rozdíly ve výsledcích po pouhém efektivnějším způsobu opravování chyb.

Název hodnoty	Hodnota pro vstup č.1	Hodnota pro vstup č.2
IF	1	1
INF	1	1
F	5	1
C	23	23
MSD chybovost	$4,2\%$	$4,2\%$
KSPC	$1,125$	$1,041$
Účinnost oprav chyb	$\mathbf{0,2}$	$\mathbf{1,000}$
Svědomitost uživatele	$0,5$	$0,5$
Nevyužité pásmo	$\mathbf{0,233}$	$\mathbf{0,115}$
Využité pásmo	$\mathbf{0,767}$	$\mathbf{0,885}$
Chybovost neopravených znaků	4%	4%
Chybovost opravených znaků	4%	4%
Celková chybovost	8%	8%

Tabulka 2.1: Hodnoty metriky pro dva různé vstupy.

Kapitola 3

Detekce a analýza psaní a editace v prostředí MS Office

V této kapitole si popíšeme metody použité v této práci pro detekci a analýzu psaní a editace. Navrhneme nové metody pro detekci a analýzu editace a pro analýzu psaní. Všechny tyto metody budou navrženy pro aplikace MS Word a Outlook, pro verze 2007 a 2010. Důvod omezení na tyto dvě verze (2007 a 2010) bude popsán v kapitole 4.1. Omezení na aplikace MS Word a Outlook je z toho důvodu, že v těchto dvou aplikacích z MS Office je zapisování většího množství textu následováno odmlkami v psaní, je tedy možno přizpůsobit metody analýzy a detekce tomuto stylu psaní a prostředí MS Office. V této kapitole budou použity pojmy jako například *délka textu*, nebo *znak*. **Za znaky budeme od teď považovat nejenom textové znaky, ale i například obrázky, grafy a jiné prvky textu. Každý takový prvek bude mít délku jedna, délka textu, například znaku a obrázku, bude tedy dva. Tabulky budou mít délku odpovídající délce jejich obsahu.**

3.1 Zachycení událostí potřebných pro detekci psaní a editace

Pro detekci psaní a detekci a analýzu editace budeme potřebovat zachytit několik různých událostí, jejichž využití si vysvětlíme později v kapitole 3.2.2. Způsob zachycení těchto akcí je závislý na prostředí MS Office a bude vysvětlen v kapitole 3.1.1. Události, které budeme potřebovat zachytávat jsou následující.

- Událost stisku klávesy.
- Událost stisku pravého tlačítka myši[1].
- Událost změny aktivního dokumentu nebo zprávy.
- Události *zpět* a *opakovat* a další *zvláštní události*.

Událostmi zpět a opakovat budeme rozumět události, které nastanou při automatizovaném vrácení nebo znovuprovedení posledních úprav. Pod pojmem zvláštní události budeme

[1] Pravého virtuálního tlačítka, ne nutně pravého fyzického tlačítka myši.

rozumět všechny jinak neuvažované události, které nemůžeme přímo detekovat. Dalšími takovými událostmi, kromě zpět a opakovat, budou například vkládání obrázku nebo tabulky pomocí stisku tlačítka myši či kláves.

Pomocí detekce těchto akcí budeme schopni určit nastalé možné okamžiky psaní či editace. Abychom však byli schopni přesně určit tyto okamžiky, budeme potřebovat zjistit některé údaje u každé z těchto akcí. Způsob zjištění těchto údajů je závislý na použité aplikaci z MS Office a upřesníme si ho později v kapitole 4.1.1. Tyto údaje jsou zde rozepsané.

- Délka textu aktivního dokumentu nebo zprávy.
- Délka právě vybraného textu.
- Pozice textového kurzoru v aktivním dokumentu nebo zprávě.

Při akci stisku klávesy navíc budeme potřebovat zjistit některé další údaje z důvodů přesnějšího určení nastalé akce. Tato akce je jediná, pro které bude potřeba zjišťovat některé údaje navíc oproti jiným akcím, zároveň to bude jediná akce detekovaná pro potřeby analýzy rychlosti psaní.

- Virtuální kód stisknuté klávesy – identifikátor psaného znaku či klávesy.
- Počet opakovaných stisků kláves – Tato hodnota je > 1 pokud klávesu držíme stisknutou bez následného uvolnění v krátkém čase [**?**]. Při této situaci jsou zasílány opakovaně nové zprávy s typem událostí *stisknutí klávesy*.
- Příznak současného stisknutí kláves *Ctrl* nebo *Alt* při stisku klávesy.
- Čas stisku klávesy.

3.1.1 Způsob zachycení jednotlivých událostí potřebných pro detekci psaní a editace

S ohledem na použité prostředky si zde rozepíšeme způsoby zachycení jednotlivých akcí, které potřebujeme pro detekci psaní a editace.

Pasivní detekce akce stisku klávesy a myši pomocí funkce z Windows API

Pro detekci akce stisku kláves na klávesnici využijeme funkci pro pasivní detekci stisků kláves z kapitoly 2.1.2. Tato funkce je volána pokaždé, když nastane akce stisku klávesy a předává nám množství zpráv, z nichž některé využijeme pro detekci psaní na klávesnici. Zpráv o stisku kláves jsou dva různé typy [**?**]: zprávy typu *stisknutí* a zprávy typu *uvolnění* klávesy. Zprávy typu stisknutí jsou zasílány v okamžiku fyzického stisknutí klávesy a opakovaně při následném držení klávesy. Zprávy typu uvolnění jsou zasílány při ukončení fyzického stisknutí klávesy. My budeme pro detekci psaní na klávesnici zachytávat pouze zprávy typu stisknutí.

Pro detekci akce stisku pravého tlačítka myši použijeme analogicky funkci z kapitoly 2.1.2, avšak nastavenou na zachytávání zpráv o akcích myši [**?**].

Detekce změny aktivního dokumentu nebo zprávy pomocí událostí MS Office

Detekci změny aktivního dokumentu nebo zprávy budeme provádět pomocí detekce vyvolaných událostí [**?**]. Tyto změny potřebujeme detekovat, abychom byli schopni správně detekovat editaci textu ve vícero oknech. **Každá událost změny aktivního dokumentu nebo zprávy je detekována dvakrát**, jednou pro okno, které zavíráme a jednou pro okno které otevíráme. Podrobněji bude tato potřeba vysvětlena v kapitole 3.2.2.

Detekce událostí zpět, opakovat a ostatních zvláštních událostí analýzou ostatních událostí

Pro detekci těchto událostí nebude možno využít událostí vyvolávaných produkty MS Office, neboť takové události pro naše potřeby neexistují. Místo toho budeme tyto události detekovat analýzou událostí akcí stisku kláves a stisku pravého tlačítka myši. Algoritmus detekce těchto událostí si vysvětlíme v kapitole 3.2.2, kde budou vysvětleny i algoritmy použité pro detekci editace textu.

3.2 Detekce a analýza editace předem neznámého textu v prostředí MS Office

Metrika popsaná v kapitole 2.3.1 byla vytvořena tak, aby umožňovala analyzovat psaní uživatele, který přepisuje předem známý text za pomoci klávesnice, bez klávesových zkratek nebo myši. Situace, kterou potřebujeme analyzovat v prostředí MS Office je však jiná. Rozdílem je jak možnost použití myši nebo klávesových zkratek, tak i například vkládání obrázků a jiné akce, které znesnadňují detekci editace psaní. Rozepíšeme si zde hlavní rozdíly.

- Máme předem neznámý text, nejsme tedy schopni s naprostou přesností určit všechny chyby při uživatelově psaní.
- Text, který uživatel píše, je podroben kontrole správnosti zabudované v prostředí MS Office. Díky tomu je množství neopravených chyb značně zmenšeno.
- Kromě základních operací psaní na klávesnici je možno používat různé zvláštní akce jako je vložení a vyjmutí textu, vrácení úprav zpět atd.
- Existuje operace nahrazení textu jiným textem.
- Dokument či zpráva mohou sestávat i z tabulek, obrázků atd.
- Je možno otevřít uložený text a následně ho editovat.

Pokusíme se tedy teď navrhnout metriku pro detekci a analýzu editace textu takovou, která by byla schopna pracovat s popsanými rozdíly a analyzovat editaci textu v prostředí MS Office.

3.2.1 Návrh metriky pro analýzu editace pro použití v produktech MS Office

Při návrhu této metriky se zaměříme na analýzu vstupu uživatele a pokusíme se z něj získat veškeré potřebné informace. Ze vstupu uživatele můžeme vyvodit výsledný text, avšak ne vstupní text. Díky této vlastnosti však nemůžeme, tak jako v metrice popsané v kapitole

2.3.1, detekovat neopravené chyby uživatele. Každou část uživatelova vstupu však můžeme rozdělit na *pět typů*.

Typ akce	**Popis akce**
Přidání písmene do textu.	Touto akcí budeme rozumět přidání jednoho písmene do textu, i kdyby bylo do textu vloženo několik písmen jednou akcí psaní uživatele.
Odebrání písmene z textu.	Touto akcí budeme rozumět odebrání jednoho písmene z textu, i kdyby bylo z textu vyjmuto několik písmen jednou akcí uživatelova vstupu.
Nahrazení textu jiným textem.	Touto akcí budeme rozumět operaci nahrazení úseku textu jiným úsekem textu. Například akce nahrazení pěti písmen třemi jinými je jedna akce, pokud bude provedena jednou akcí uživatelova vstupu.
Pohyb textového kurzoru, pokud nenastane při přidání textu.	Touto akcí rozumíme změnu polohy textového kurzoru pomocí jedné akce uživatelova vstupu, pokud touto stejnou akcí uživatelova vstupu nenastane akce přidání písmene.
Pohyb textového kurzoru, pokud nastane při přidání textu.	Touto akcí rozumíme změnu polohy textového kurzoru pomocí jedné akce uživatelova vstupu, pokud touto stejnou akcí uživatelova vstupu nastane akce přidání písmene.

Tabulka 3.1: Tabulka typů uživatelova vstupu

Poslední ze seznamu akcí budeme pro účely této metriky ignorovat, neboť množství akcí, které uživatel potřebuje pro napsání textu, nás nebude v této metrice zajímat. Dále můžeme nahradit každou akci nahrazení textu množinou akcí přidání písmene do textu a odebrání písmene z textu. Celkem tedy máme nakonec tři různé *základní typy akcí*, které budeme detekovat.

Tyto základní typy akcí si pro naše potřeby označíme následovně.

- Akce přidání písmene jako $\boldsymbol{P}$.
- Akce odebrání písmene jako $\boldsymbol{IFW}$.
- Akce pohybu textového kurzoru jako $\boldsymbol{FW}$.

Pokud se podíváme na metriku popsanou v kapitole 2.3.1, můžeme si všimnout určitých podobností mezi typy akcí v naší metrice a typy akcí popsanými v metrice v kapitole 2.3.1. Tyto podobnosti jsou dostatečné na to, abychom mohli popsat jednotlivé naše základní akce ekvivalentním popisem z metriky, popsané v kapitole 2.3.1. Výjimku tvoří typ akce FW, který je rozšířením akcí typu F, popsaných v kapitole 2.3.1, takovým, že v sobě zahrnuje určité pohyby textového kurzoru jako akci směřující k opravě chyb, nebo obecně jako akci, kterou v ideálním psaní textu není třeba provádět. Protože se však jedná pouze o logické rozšíření oproti druhé metrice, budeme **považovat $\boldsymbol{FW}$ za nástupce $\boldsymbol{F}$ pro námi navrženou metriku**.

Pro všechny typy základních akcí platí následující vzorce, popisující jejich vztah vzhledem k typům akcí popsaných v metrice z kapitoly 2.3.1. Od teď dále budeme pod označením P, IFW a FW rozumět počty prvků, náležící jednotlivým typům akcí.

$$P = IF + C + INF \tag{3.1}$$
$$IFW = IF \tag{3.2}$$
$$FW = F \tag{3.3}$$

U těchto hodnot budeme navíc uvažovat situaci, kdy uživatel otevřel dokument se zapsaným textem a tento text následně smazal. **Pokud taková situace nastane, budeme pouze zvyšovat hodnotu FW. Hodnotu P nebudeme zvyšovat, pokud by měla převýšit hodnotu IFW**, neboť by mohla nastat situace, kdy jsme smazali více znaků než jsme napsali, což by mohlo vést k nesmyslným výsledkům pro jednotlivé vzorce metriky.

Když teď máme nalezený vztah mezi námi navrhovanou metrikou a metrikou popsanou v kapitole 2.3.1, pokusíme se využít vzorce pro popis uživatelova psaní a editace, které jsou definovány v kapitole 2.3.2.

U některých vzorců nastává problém, neboť jak již bylo řečeno, nejsme schopni detekovat chybně napsané a neopravené znaky. Tuto naši neschopnost však v podstatné míře nahrazuje schopnost automatické opravy chyb, popsaná v kapitole 3.2. Proto tedy budeme uvažovat pro zjednodušení, že $INF \approx 0$. Toto zjednodušení nám umožní využít všechny vzorce, které zde použijeme. Vzorce, které náleží metrice popsané v kapitole 2.3.1, označíme jako *původní*.

$$\begin{aligned}
KSPC\ (původní) &= KSPC \\
Účinnost\ oprav\ chyb\ (původní) &= Účinnost\ oprav\ chyb \\
Nevyužité\ pásmo\ (původní) &\approx Nevyužité\ pásmo \\
Využité\ pásmo\ (původní) &\approx Využité\ pásmo \\
Chybovost\ opravených\ znaků\ (původní) &= Chybovost\ opravených\ znaků
\end{aligned}$$

Pomocí nich odvodíme nové vzorce pro použití v naší metrice. S těmito vzorci už získáváme užitečné statistické údaje, které později použijeme v implementaci.

$$KSPC = \frac{P - IFW + FW}{P - IFW} \tag{3.4}$$
$$Účinnost\ oprav\ chyb = \frac{IFW}{FW} \tag{3.5}$$
$$Nevyužité\ pásmo = \frac{IFW + FW}{P + FW} \tag{3.6}$$
$$Využité\ pásmo = \frac{P - IFW}{P + FW} \tag{3.7}$$
$$Chybovost\ opravených\ znaků = \frac{IFW}{P} \tag{3.8}$$

3.2.2 Zjištění uživatelských akcí potřebných v metrice ze zachycených událostí

Pro zjištění jednotlivých základních akcí z uživatelského vstupu potřebujeme převést jednotlivé události tvořící vstup, popsané v kapitole 3.1, na základní akce popsané v kapitole 3.2.1. Všechny algoritmy pro tento převod budou používat dvě množiny parametrů – **parametry současné události a parametry budoucí události**. Takto to bude z toho důvodu, že budeme potřebovat porovnávat stav před událostí a po ní, abychom zaznamenali změny provedené událostí. Porovnávání současné a budoucí události bude potřeba provádět až budoucí událost nastane, proto bude algoritmus převodu události na základní akci pracovat se zpožděním vždy jedné události. Tento postup je nutný z toho důvodu, **že jednotlivé události jsou nám oznámeny před tím, než jsou dále zpracovávány a provedou změny v dokumentu**. Zajištění převodu všech událostí na množiny základních akcí bude provedeno díky zachytávání událostí změn aktivního dokumentu nebo zprávy. **Ve výsledku tedy budeme reagovat na současnou událost až v okamžiku, kdy nastane událost budoucí.**

Všechny algoritmy pro převod událostí budou pracovat se společnou frontou událostí, která obsahuje jednotlivé události v pořadí, ve kterém byly vyvolány. Také budou mít společnou množinu proměnných pro uložení informací o současné události, aby bylo možné ji porovnat s budoucí událostí, až tato nastane. Algoritmy porovnají parametry současné události a budoucí události a následně rozhodnou, zda a jak současnou událost převedou na množinu základních akcí.

Uvedeme si zde teď algoritmy převodu jednotlivých událostí. **Všechny tyto algoritmy jsou volány v závislosti na typu současné události, ne budoucí**, takže pro posloupnosti

$$\text{stisk klávesy enter} \rightarrow \text{stisk tlačítka myši}$$

budeme pro detekci události stisku tlačítka myši volat algoritmus pro zpracování stisku klávesy enter.

Veškerá detekce editace bude prováděna na právě otevřeném dokumentu či zprávě, nikoliv na jiných otevřených a aktivních oknech, tudíž například úpravy nastavení nebudou brány pro detekci editace v potaz.

Parametry údajů popisující události

Pro údaje blíže popisující události si napřed zavedeme zkrácený tvar, abychom mohli přehledněji zapsat algoritmy popisující převod událostí na základní akce.

Délka textu aktivního dokumentu nebo zprávy	$= \boldsymbol{DDokument}$
Délka právě vybraného textu	$= \boldsymbol{DText}$
Pozice textového kurzoru	$= \boldsymbol{CPozice}$
Virtuální kód stisknuté klávesy	$= \boldsymbol{KVirtual}$
Počet opakovaných stisků kláves	$= \boldsymbol{KOpakovani}$
Příznak kláves Ctrl *nebo* Alt	$= \boldsymbol{KCtrlAlt}$
Čas stisku klávesy	$= \boldsymbol{KCas}$

Pokud bude údaj popisovat současnou událost, přidáme na konec jeho názvu $\_S$, pokud bude popisovat budoucí událost, přidáme $\_B$. Ve výsledku budou mít tvar například $DDokument\_S$, $DDokument\_B$.

Poslední čtyři uvedené údaje popisují pouze událost stisku klávesy, u ostatních událostí je tedy nebudeme uvažovat. U virtuálních kódů kláves si jednotlivé klávesy zařadíme do dvou skupin, v závislosti na jejich významu. Budou to skupiny *klávesy písmen* a *klávesy ostatní*. Klávesami písmen budeme rozumět klávesy reprezentující jednotlivé znaky, klávesami ostatními budou klávesy nezařazené do kláves písmen.

Událost stisku klávesy

Následující algoritmus popisuje převod události stisku klávesy na množinu základních akcí, popsaných v kapitole 3.2.1. Tento algoritmus obsahuje detekci několika různých akcí, mezi jinými i detekci událostí zpět a opakovat. Algoritmus zároveň uvažuje akci nahrazení textu jiným textem a pokud tato akce nastane, bude převedena na základní akce odebrání písmen a přidání písmen. Také uvažuje operaci vložení jednoho či více znaků pomocí klávesových zkratek. Algoritmus obsahuje volání dvou funkcí. Funkce *Nastav_Budouci_na_Soucasne()* nastaví všechny (i parametry nepoužívané algoritmem) parametry budoucí akce jako současné a tím připraví algoritmus na další volání pro příští událost. Funkce *Zpet_Opakovat()* se volá pro detekci možných událostí zpět nebo opakovat a při odstranění znaků.

Algoritmus 1: Klávesy

Input: $DDokument\_B, DText\_B, CPozice\_B, KVirtual\_B, KCtrlAlt\_B$
Input: $DDokument\_S, DText\_S, CPozice\_S, KVirtual\_S, KCtrlAlt\_S$

```
    // Detekce stisku klávesy písmene nebo vložení textu
 1: if (NOT KCtrlAlt AND KVirtual_S == Klávesa_písmene)
 2: OR (KCtrlAlt AND (KVirtual_S == 'V' OR KVirtual_S == 'Insert'))
 3: then
 4:     if DText_S > 0 then
 5:         FW += 1
 6:         IFW += DText_S
 7:     end if
 8:     P += (DDokument_B - DDokument_S + DText_S)
        // Zajištění omezení hodnoty IFW
 9:     if P < IFW then
10:         IFW = P
11:     end if
12: end if
    // Detekce pohybu textového kurzoru bez úprav textu
13: else if DDokument_S == DDokument_B AND CPozice_S != CPozice_B then
14:     FW += 1
15: end if
    // Detekce událostí zpět, opakovat a odstranění znaků.
    // Také detekuje jinak nepopsané případy.
16: else if DDokument_S != DDokument_B then
17:     Zpet_Opakovat()
18: end if
19: Nastav_Budouci_na_Soucasne()
```

Událost stisku pravého tlačítka myši

Zde je uveden algoritmus pro převod události stisku klávesy na základní akce. Tento algoritmus detekuje události zpět a dopředu a také událost pohybu textového kurzoru. Jiné akce a události není potřeba detekovat, neboť je popisují ostatní algoritmy. V těle algoritmu jsou stejně jako v předchozím algoritmu, volány dvě funkce.

Algoritmus 2: Myš

Input: $DDokument\_B, CPozice\_B$
Input: $DDokument\_S, CPozice\_S$

```
    // Detekce pohybu textového kurzoru
1:  if CPozice_S != CPozice_B then
2:      FW += 1
3:  end if
    // Detekce událostí zpět, opakovat a ostatních zvláštních událostí
    // Obsahuje i detekci vložení a vyjmutí znaků pomocí myši
4:  else if DDokument_S != DDokument_B then
5:      Zpet_Opakovat()
6:  end if
7:  Nastav_Budouci_na_Soucasne()
```

Událost změny aktivního dokumentu nebo zprávy

Algoritmus změny aktivního dokumentu nebo zprávy je velice jednoduchý, jeho jediným účelem je vložit prázdnou akci mezi ostatní akce a zajistit, že parametry současné události se nastaví tak, aby pro příští událost korektně vyjadřovaly stav dokumentu nebo zprávy. Jak již bylo řečeno v kapitole 3.1.1, každá událost změny této změny je detekována dvakrát. Jednou před změnou a jednou po změně. Díky tomuto získáme vždy stav aktivního prvku v okamžiku ukončení/počátku jeho aktivity.

Události zpět, opakovat a ostatní zvláštní události

Události zpět a opakovat je potřeba detekovat během jiných událostí, neboť MS Office nám neumožňuje je detekovat přímo. Z tohoto důvodu je algoritmus zpracování těchto událostí volán jako funkce v jiných algoritmech a není třeba v něm tudíž volat funkce *Nastav_Budouci_na_Soucasne()*. Níže uvedený algoritmus má jednu situaci, kdy nebude správně zobrazovat realitu. Touto situací je, když události zpět nebo opakovat odstraní událost nahrazení textu. Pokud toto nastane, nejsme schopni z námi známých parametrů zjistit, kolik bylo znaků nahrazeno. Vše co jsme schopni zjistit je, kolik znaků bylo přidáno nebo odebráno. Toto je také hlavním důvodem, proč jsou jednotlivé události zpracovávány vlastními algoritmy a až pokud nelze nalézt jiný algoritmus pro zpracování nastálé události, využije se tohoto.

Algoritmus 3: Zpet_Opakovat

Input: $DDokument\_B$
Input: $DDokument\_S, DText\_S$

```
    // Operace přidala znaky do textu
 1: if DDokument_B > DDokument_S then
 2:     P+= (DDokument_B - DDokument_S + DText_S)
 3:     IFW += DText_S
 4:     if DText_S != 0 then
 5:         FW+=1
 6:     end if
        // Zajištění omezení hodnoty IFW
 7:     if P < IFW then
 8:         IFW = P
 9:     end if
10: end if
    // Operace odebrala znaky z textu
11: else
12:     IFW += - (DDokument_S - DDokument_B)
13:     FW += 1
        // Zajištění omezení hodnoty IFW
14:     if P < IFW then
15:         IFW = P
16:     end if
17: end if
```

Uvedený algoritmus byl posledním algoritmem, který jsme potřebovali pro převod jednotlivých událostí. Můžeme si všimnout, že algoritmy mají některé části společné, proto bude v implementaci možné je zkombinovat do jednoho algoritmu. Pro zjednodušení jsme však uvedli všechny algoritmy samostatně, aby byla zřejmá jejich funkčnost a jejich omezení.

Tímto jsme dokončili popis naší metriky pro detekci a analýzu editace textu, včetně způsobu získávání všech potřebných údajů.

3.3 Detekce psaní a analýza rychlosti psaní uživatele

Zde si popíšeme metodu, kterou použijeme pro detekci a následnou analýzu psaní uživatele na klávesnici. Tato metoda detekuje a analyzuje psaní tak, aby popsala reálnou rychlost uživatele. Tato metoda je navržena tak, aby popisovala reálnou rychlost psaní uživatele, který se zabývá *nestálým psaním textu*. Tímto psaním budeme rozumět styl psaní, kdy uživatel aktivně nemusí psát od spuštění programu do jeho ukončení, ale může docházet k různě dlouhým časovým úsekům, při kterých nepíše.

3.3.1 Detekce psaní uživatele

Pro detekci psaní uživatele na klávesnici využijeme funkci pro pasivní detekci stisků kláves z kapitoly 2.1.2 tak, jak jsme ji popsali v kapitole 3.1.1. Detekce psaní bude pracovat pouze se stisky kláves na klávesnici, ale na rozdíl od analýzy editace 3.2.1, bude tato detekce detekovat veškeré stisky kláves, ne pouze stisky klávesy upravující písmena v dokumentu či zprávě. Pro detekci psaní budeme uvažovat dva zvláštní případy, které mohou nastat. Těmito případy jsou opakovaný stisk klávesy bez jejího uvolnění a dlouhý časový úsek, při kterém uživatel aktivně nepíše text, ale například přemýšlí nad jeho zněním. Oba dva případy si zde rozepíšeme.

3.3.2 Opakovaný stisk klávesy

Opakovaným stiskem klávesy budeme rozumět takový stisk klávesy, který nastává při dlouhodobém stisknutí klávesy bez jejího uvolnění. Informaci o tomto případu dostáváme pomocí jednoho ze zaznamenávaných údajů pro detekci stisku klávesy (viz kapitola 3.1). Pokud tento opakovaný stisk klávesy nastane, musíme ho nějakým způsobem řešit. Pokud bychom to neudělali, tak například pro situaci, kdy uživatel napíše větu a tuto větu následně smaže jedním dlouhým stiskem klávesy *Backspace*, bychom detekovali množství kláves velice rychle za sebou. Pokud tedy zjistíme v průběhu analýzy takovýto opakovaný stisk klávesy, musíme ho brát jako jednu jedinou akci stisku klávesy, která má však na rozdíl od ostatních akcí stisků kláves nějakou časovou délku. Tím, že budeme u takovéto akce uvažovat její délku zajistíme, že čas mezi opakovaným stiskem a příštím neopakujícím stiskem klávesy bude stejný, jako čas, který by byl mezi dvěma stisky kláves, jako kdyby opakující akce mezi nimi neexistovaly. Na níže uvedeném příkladu je zobrazen vstup uživatele, kdy pro detekci uvažujeme opakované stisky kláves a vstup, kdy je ignorujeme.

Vstup uživatele:	$a\langle 33\rangle, a\langle 33\rangle, a\langle 33\rangle, a\langle 33\rangle, a\langle 150\rangle, b\langle 200\rangle, c$
Vstup uživatele bez opakovaných stisků:	$a\langle 150\rangle, b\langle 200\rangle, c$
Výsledný text:	aaaaabc

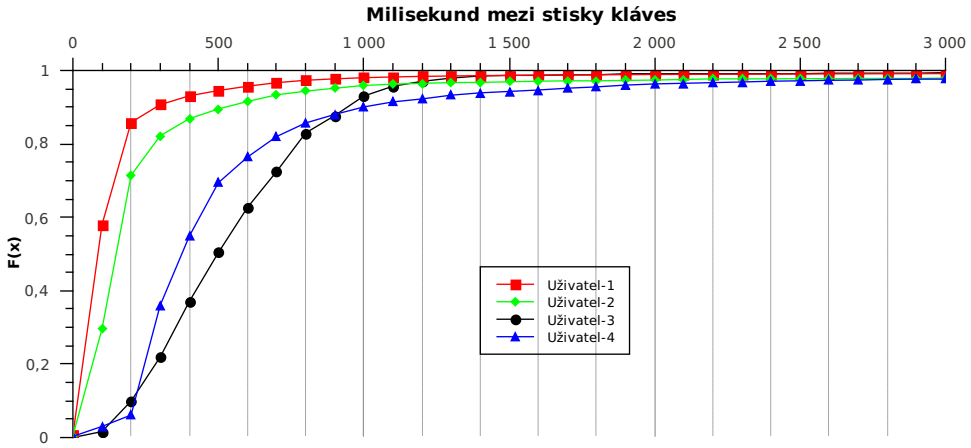
Vysvětlivky:

a,b,c	Jednotlivé klávesy.
$\langle X\rangle$	Počet milisekund mezi stiskem klávesy a následujícím stiskem klávesy.
$a\langle 33\rangle$	Opakovaný stisk klávesy a.

3.3.3 Úsek nepsaní uživatele na klávesnici

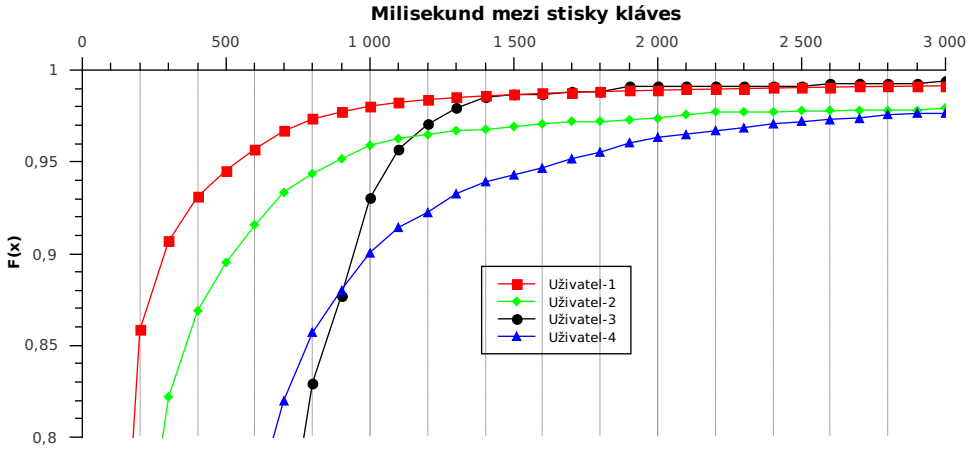
Při dlouhodobém psaní uživatele na klávesnici nastávají situace, kdy po delší časový úsek uživatel nepíše, ale provádí nějakou jinou činnost, která přímo nesouvisí s psaním textu na klávesnici. Takovému časovému úseku budeme říkat úsek *nepsaní*.

Abychom mohli tyto úseky detekovat, potřebujeme určit časovou délku, po jejímž překročení už budeme považovat tento časový úsek za úsek nepsaní. Abychom toto zjistili, vybrali jsme čtveřici uživatelů se značně rozdílnou rychlostí psaní a analyzovali jsme jejich psaní. Následující graf zobrazuje distribuční funkci popisující čas mezi stisky kláves jednotlivých uživatelů. Údaje do grafu byly vzorkovány po 100 *ms*, u každého dobrovolníka bylo potřeba aspoň 1000 naměřených stisků kláves během alespoň 3 sezení.



Obrázek 3.1: Graf distribuční funkce popisující čas mezi stisky kláves čtyř různých uživatelů. Čím rychlejší růst křivky v blízkosti osy Y, tím rychlejší a lepší psaní uživatele.

Na tomto grafu si můžeme všimnout v okolí x-ové souřadnice $x = 1000$, že naměřených hodnot větších než 1000 je velmi málo. Konkrétní množství záleží na stylu psaní uživatele, avšak u všech uživatelů šlo pozorovat změnu od určité hodnoty. Abychom tuto změnu zjistili podrobněji, ukážeme si následující graf, který zobrazuje podrobnější pohled na předchozí graf.



Obrázek 3.2: Přiblížení grafu.

Zde vidíme, že mezi hodnotami 1000 a 1500 růst grafu klesá na své minimum. Z tohoto vyvodíme, že hodnoty větší než 1500 nevznikly v důsledku aktivního psaní na klávesnici, ale z jiných důvodů (například přemýšlení nad textem k psaní). Určíme si tedy jako hraniční hodnotu detekce nepsaní 2000 *ms*. Tuto hodnotu určíme raději vyšší, než je zřejmé z grafu, že je potřeba, z důvodů zajištění odchycení všech stisků kláves zapsaných aktivním psaním na klávesnici.

Při pohledu na grafy lze odhadnout, že mediány naměřených hodnot u některých uživatelů by se značně lišily od průměrných hodnot, bez ohledu na to, jakou hranici mezi 500 a 2000 bychom vybrali. Toto je z důvodu nerovnoměrnosti rozložení hodnot. To je hlavním důvodem proč omezit hodnoty pomocí detekce nepsaní a zachovat tak pro analýzu rychlosti psaní možnost výpočtu průměru.

Na závěr si zde uvedeme tabulku ukazující hodnoty průměrného času mezi stisky kláves jednotlivých vybraných uživatelů s detekcí a bez detekce nepsaní. Z této tabulky jdou také vyčíst schopnosti jednotlivých uživatelů, všimněme si zvláště „Uživatele-3“, který udává příklad uživatele se značně nízkou rychlostí psaní a ukazuje, že i pro takového uživatele lze použít detekci nepsaní. Tato metoda by nešla použít pouze u uživatele, který by měl zvláště malou rychlost psaní, nepředpokládá se však, že by takovýto uživatel využíval nástroje pro analýzu rychlosti psaní.

Uživatel	**Průměrný čas mezi stisky kláves**	**Průměrný čas mezi stisky kláves s detekcí nepsaní**
Uživatel-1	437 *ms*	130*ms*
Uživatel-2	640 *ms*	202*ms*
Uživatel-3	1403 *ms*	534*ms*
Uživatel-4	613 *ms*	457*ms*

Tabulka 3.2: Porovnání průměrných časů mezi stisky kláves uživatelů.

3.3.4 Analýza rychlosti psaní

Při analýze rychlosti psaní se zaměříme na časové rozdíly mezi dvěmi stisky kláves. Tyto časové rozdíly budeme označovat jako *prodlevy* a budeme je uvádět v řádech milisekund. Budeme zjišťovat jak obecně užívaný ukazatel počtu úhozů za minutu, tak se i pokusíme využít ukazatele, které nám zjistí podrobnější informace o uživatelově psaní.

Počet úhozů za minutu

Počet úhozů za minutu je používaný ukazatel pro vyjádření rychlosti psaní uživatele. Tento ukazatel budeme počítat z toho důvodu, že chceme zjistit rychlost psaní prsty uživatele, bez ohledu na jeho chyby. Pro výpočet budeme používat počet stisků kláves za sekundu, který budeme počítat z aritmetického průměru prodlev mezi stisky jednotlivých kláves. Tento průměr označíme jako $\bar{x}$, výsledný počet úhozů za minutu označíme jako U. Výsledný vzorec bude mít tedy následující tvar.

$$U = \frac{1000}{\bar{x}} \times 60$$

3.3.5 Průměrná prodleva a medián prodlevy mezi stisky kláves, jejich odchylky

V analýze rychlosti psaní budeme kromě počtu úhozů za minutu zjišťovat ještě dva ukazatele – *průměrnou prodlevu* a *medián prodlevy*. Průměrnou prodlevou budeme rozumět aritmetický průměr zaznamenaných prodlev. Mediánem prodlevy budeme rozumět medián zaznamenaných prodlev. Jak již bylo řečeno v kapitole 3.3.4, průměrnou prodlevu využijeme pro výpočet počtu úhozů za minutu. Medián prodlevy budeme počítat z důvodu větší přesnosti oproti aritmetickému průměru při menším množstvím hodnot.

Odchylkami od mediánu a od průměru budeme rozumět průměrnou odchylku od těchto dvou hodnot. Tyto odchylky, zvláště od mediánu, budeme zjišťovat proto, abychom přesněji popsali styl uživatelova psaní. Na příkladu si teď uvedeme rozdíly mezi dvěma uživateli, kteří mají podobný medián a průměr prodlevy psaní.

Vstup uživatele č.1:	$i\langle 256\rangle$, $n\langle 223\rangle$, $t\langle 289\rangle$, $e\langle 230\rangle$, $r\langle 172\rangle$, $p\langle 291\rangle$, $u\langle 212\rangle$, $n\langle 243\rangle$, $k\langle 267\rangle$, $c\langle 310\rangle$, $e\langle 595\rangle$, $\_\langle 190\rangle$, $j\langle 320\rangle$, $e\langle 251\rangle$, $\_\langle 181\rangle$, $d\langle 310\rangle$, $u\langle 258\rangle$, $l\langle 255\rangle$, $e\langle 240\rangle$, $z\langle 294\rangle$, $i\langle 220\rangle$, $t\langle 212\rangle$, $a\langle 290\rangle$
Vstup uživatele č.2:	$i\langle 356\rangle$, $n\langle 323\rangle$, $t\langle 189\rangle$, $e\langle 130\rangle$, $r\langle 272\rangle$, $p\langle 391\rangle$, $u\langle 112\rangle$, $n\langle 143\rangle$, $k\langle 167\rangle$, $c\langle 315\rangle$, $e\langle 795\rangle$, $\_\langle 190\rangle$, $j\langle 220\rangle$, $e\langle 351\rangle$, $\_\langle 281\rangle$, $d\langle 410\rangle$, $u\langle 958\rangle$, $l\langle 155\rangle$, $e\langle 240\rangle$, $z\langle 254\rangle$, $i\langle 260\rangle$, $t\langle 232\rangle$, $a\langle 270\rangle$
Výsledný text:	interpunkce je dulezita

Vysvětlivky:

a,b,c	Jednotlivé klávesy.
$\langle X\rangle$	Počet milisekund mezi stiskem klávesy a následujícím stiskem klávesy.
$\_$	Klávesa mezerník.

V tabulce máme uvedeny hodnoty pro oba dva vstupy, všimněme si zvláště jednotlivých odchylek.

Název hodnoty	**Hodnota pro vstup č.1**	**Hodnota pro vstup č.2**
Průměrná prodleva	265,6	274,4
Odchylka od průměrné prodlevy	**50**	**105,2**
Medián prodlevy	255	254
Odchylky od mediánu prodlevy	**48,1**	**102,9**

Tabulka 3.3: Hodnoty analýzy psaní pro dva různé vstupy

Z uvedených odchylek lze vyčíst, že i když se průměrné hodnoty vzájemně skoro shodovaly, tak rozdíl mezi stylem psaní uživatelů byl obrovský. Uživatel č. 1 psal většinou stálým tempem, avšak uživatel č. 2 měl značně rozdílnou rychlost psaní jednotlivými klávesami. Z tohoto můžeme vyvodit, že pokud by se uživatel č. 1 chtěl zaměřit na zvýšení rychlosti svého psaní, bude potřebovat zvýšit hlavně celkovou rychlost psaní, avšak pro uživatele č. 2 bude důležitější zmenšit rozdíly mezi rychlostí psaní různých znaků.

Pokud chceme uživateli prezentovat rozdíly v jeho rychlosti psaní jednotlivými klávesami, máme dvě možnosti. První možností je prezentovat mu uvedené odchylky, kdy ignorujeme prodlevu mezi jednotlivými klávesami, ale počítáme odchylky od všech kláves naráz. Druhou možností by byla metoda analýzy hmatové techniky popsaná v kapitole 2.2.2. Tuto metodu však nevyužijeme, neboť není funkční při jiném rozložení prstů na klávesnici.

3.4 Využití analýzy editace společně s analýzou rychlosti psaní

Při dosavadní analýze rychlosti psaní jsme ignorovali veškerou analýzu editace psaní. Pokud však zkombinujeme analýzu rychlosti psaní s analýzou editace, budeme schopni určit reálnou rychlost psaní uživatele. Pokud například uživatel píše rychlostí 80 úhozů minutu, avšak polovinu napsaného textu psal zbytečně a následně ji smaže, tak jeho reálná rychlost psaní je nižší než u uživatele, který píše rychlostí 60 úhozů za minutu, avšak veškerý jeho text zůstane. Abychom mohli přesněji zobrazit reálnou rychlost uživatelova psaní, zavedeme si nový ukazatel, který bude kombinovat analýzu rychlosti a analýzu editace psaní. Ukazatel nazveme *Počet úhozů se započtením účinnosti* a budeme ho značit jako U_u. Budeme ho počítat z hodnoty *nevyužité pásmo*, kterou označíme P_n, a z hodnoty *počet úhozů za minutu*, označenou jako U. Vzorec pro výpočet je zde uveden.

$$U_u = P_n \times U \tag{3.9}$$

Tento ukazatel nám udává, jakou rychlostí by uživatel skutečně psal, pokud by za stejný časový úsek napsal stejně dlouhý text s nejvyšší možnou účinností psaní.

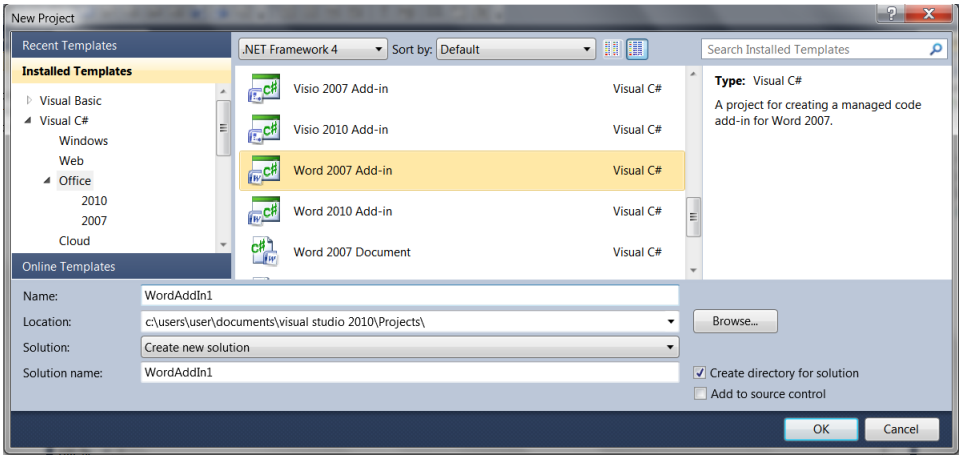
Kapitola 4

Implementace a testy

Za implementační nástroj doplňku (add-inu) byla vybrána aplikace Microsoft Visual Studio 2010. Tato aplikace byla vybrána z toho důvodu, že poskytuje plnou podporu pro programování doplňků pro MS Office a s určitými licenčními omezeními je bezplatně k dostání pro studenty [**?**]. Jako programovací jazyk bylo zvoleno *C#* s platformou *.NET* [**?**]. Vytvořené doplňky pro MS Word a Outlook jsou dva samostatné programy, které však mají společné množství tříd, díky čemuž se urychlil vývoj. Doplňky byly vytvořeny pomocí *VSTO* [**?**], což je sada nástrojů pro tvorbu doplňků do MS Office.

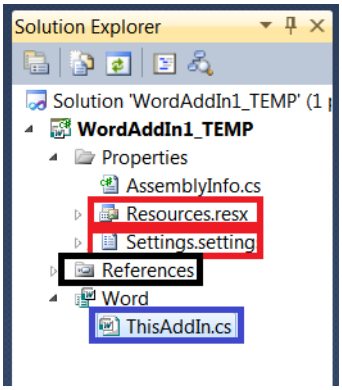
4.1 Tvorba doplňku ve Visual Studio 2010

Při tvorbě doplňku v programu Visual Studio 2010 se postupuje následovně. Napřed se vytvoří projekt doplňku, který chceme tvořit. Doplňky je možno tvořit pro každou aplikaci z balíku MS Office. Programátor si při tvorbě doplňku vybere jazyk, ve kterém bude doplněk implementovat (C#, nebo Visual Basic), verzi platformy .NET a také pro jakou aplikaci bude doplněk tvořit. Je možno tvořit doplňky pro aplikace z balíku MS Office od verzí 2007. Pro aplikace z předchozích balíků MS Office nelze tvořit doplňky v aplikaci Visual Studio 2010. Doplňky nebudeme tvořit pro aplikace z balíků MS Office verzí 2003 a nižších z důvodu neexistence některých událostí a atributů v těchto aplikacích, které potřebujeme pro detekci editace.



Obrázek 4.1: Vytvoření nového doplňku.

Na následujícím obrázku je zobrazena struktura nově vytvořeného doplňku. **Modře** je orámována třída *ThisAddIn*, která se vytváří s každým doplňkem a která je zároveň vstupním bodem, kterým je každý doplněk spouštěn. **Červeně** jsou orámovány soubory, které umožňují globální přístup k některým proměnným. Soubor *Resources.resx* umožňuje přístup k jednotlivým zdrojům, využívaných doplňkem, například obrázkům atd. Soubor *Settings.setting* umožňuje přístup k nastavením aplikace. Těmito nastaveními jsou například řetězce či jiné základní datové typy. Tato nastavení se dělí na dva druhy – *globální* a *uživatelské*. Globální na rozdíl od uživatelských nelze za běhu doplňku měnit. Poslední zvýrazněnou částí obrázku a to **černě** je skupina *References*, která obsahuje seznam knihoven. Tento seznam je při vytvoření nového doplňku nastaven na minimální množinu potřebných knihoven, které jsou třeba pro základní funkce doplňku.



Obrázek 4.2: Výchozí stav nového doplňku.

Do doplňků je možno přidat i třídu pro jeho grafické rozhraní v rámci systému Ribbon, které umožňuje doplňku dosadit část uživatelského rozhraní (tlačítko, obrázek atd.) do menu aplikace, pro kterou doplněk tvoříme.

Tímto jsme si popsali základ tvorby doplňků ve Visual Studio 2010 a vysvětlíme si teď postup pro získání údajů popsaných v kapitole 3.1 z aplikací MS Word a Outlook, které potřebujeme pro implementaci našich doplňků.

4.1.1 Zjištění údajů o dokumentu či zprávě z MS Office

U každé události, kterou zachytíme (viz kapitola 3.1), potřebujeme zjistit několik údajů. Všechny tyto údaje jsou nám k dispozici v rámci programování doplňku jako hodnoty přístupné přes statickou třídu *Globals*. Tato třída nám umožňuje přistupovat k hodnotám a metodám aplikace, pro kterou tvoříme doplněk, odkudkoliv ze zdrojového kódu. V následující tabulce si rozepíšeme, kde se tyto hodnoty nalézají v doplňcích pro aplikace MS Word a Outlook 2007 a 2010.

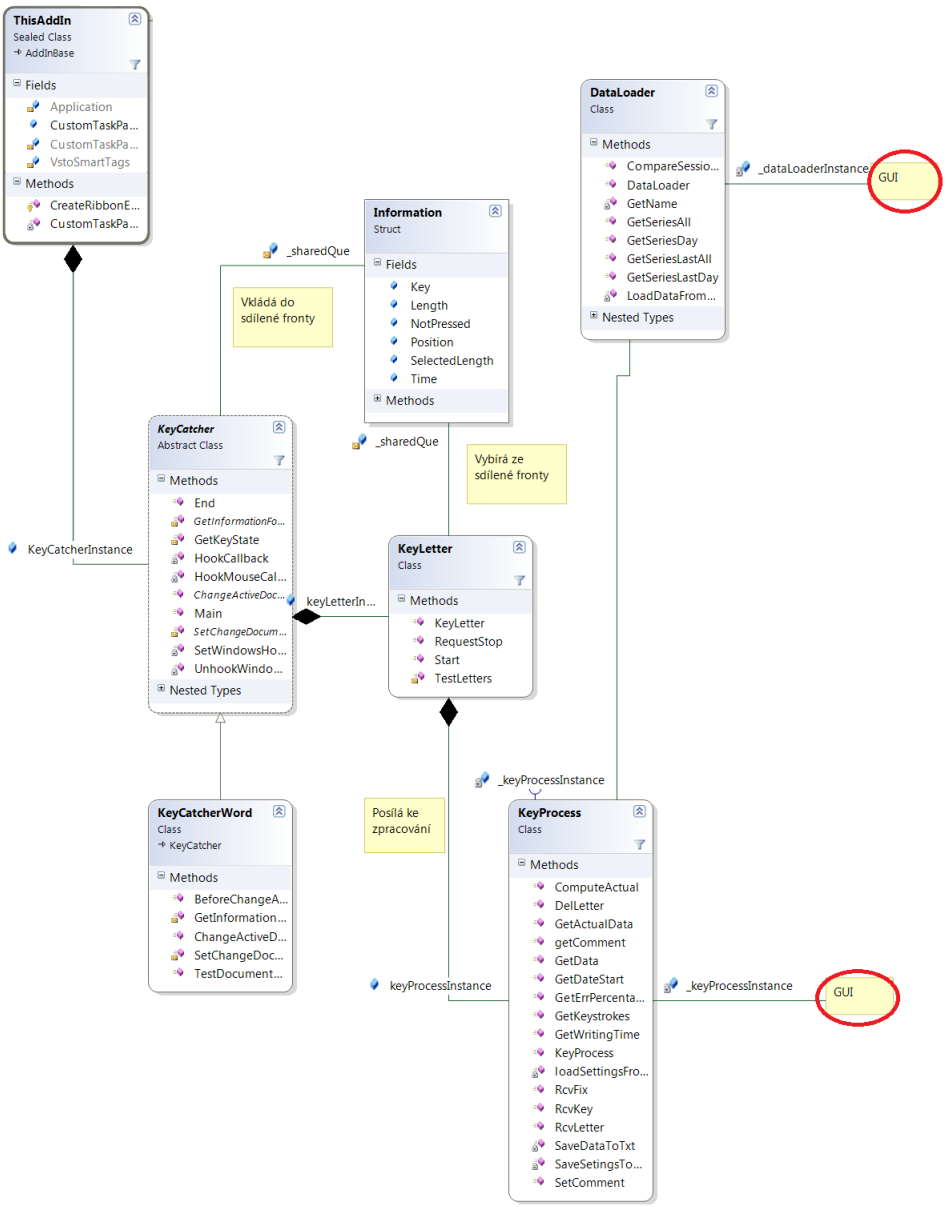
Délka textu aktivního dokumentu nebo zprávy.	
MS Word Outlook	Globals.ThisAddIn.Application.ActiveDocument.Content.Text.Length Globals.ThisAddIn.Application.ActiveInspector().WordEditor .Content.Text.Length
Délka právě vybraného textu.	
MS Word Outlook	Globals.ThisAddIn.Application.ActiveWindow.Selection.Text.Length Globals.ThisAddIn.Application.ActiveInspector().WordEditor .Windows[1].Selection.Text.Length
Pozice textového kurzoru.	
MS Word Outlook	Globals.ThisAddIn.Application.Selection.Start Globals.ThisAddIn.Application.ActiveInspector().WordEditor .Application.Selection.Start

Tabulka 4.1: Zjištění údajů z MS Office.

Při zjišťování délky vybraného textu bude potřeba provést kontrolu, zda výběr obsahuje text, nebo nějaký jiný prvek. Pokud objekt *Selection* obsahuje jeden jiný objekt než je text, hodnota *Selection.Text.Length* je nulová a hodnota *Selection.Type* udává druh objektu. Pokud objekt *Selection* obsahuje více objektů, délka *Selection.Text.Length* udává celkovou délku bez ohledu na typ objektů a hodnota *Selection.Type* udává typ objektu jako text. Pro zjišťování aktuální vybrané délky tedy musíme vždy kontrolovat hodnotu *Selection.Type* a pokud neudává typ objektu jako text, tak je délka právě vybraného textu = 1.

4.2 Implementace datové vrstvy doplňků

Datová struktura obou doplňků je skoro shodná. Doplňky se skládají ze dvou hlavních vrstev - datové vrstvy a uživatelského rozhraní. Uvedeme si zde diagram tříd datové vrstvy doplňků pro MS Word a následně si popíšeme význam jednotlivých tříd. V kapitole 4.3 si později popíšeme i uživatelské rozhraní a formu jeho komunikace s datovou vrstvou.



Obrázek 4.3: Diagram tříd datové vrstvy a jeho napojení na uživatelské rozhraní.

Na obrázku 4.3 vidíme diagram datové vrstvy doplňků a napojení této vrstvy na uživatelské rozhraní. Popíšeme si teď jednotlivé třídy a struktury datové vrstvy a jejich význam. Podrobnější popis jednotlivých tříd a struktur lze nalézt v komentářích ve zdrojových kódech.

4.2.1 Struktury *Information* a *SessionData*

Popíšeme si napřed datové struktury, které jsou využity v implementaci pro předávání a uchovávání hodnot. Struktura *Information* slouží k předávání údajů o jednotlivých událostech, popsaných v kapitole 3.1. Obsahuje podrobné informace o zachycené události. Struktura *SessionData* obsahuje informace o analýze psaní a editace jednoho sezení[1]. Dále také obsahuje metody pro získání těchto informací v různých formách, které jsou vyžadovány pro zobrazení analyzovaných informací v grafické vrstvě.

4.2.2 Třída *ThisAddIn*

Jak už bylo napsáno v kapitole 4.1, metoda této třídy *ThisAddIn_Startup* je volána pro spuštění každého doplňku. Třída vytváří instanci třídy *KeyCatcher* a instance několika tříd uživatelského rozhraní a spouští a nakonec i ukončuje celkovou činnost doplňků. Pro vytváření instance třídy KeyCatcher se využívá kontruktor potomka této abstraktní třídy *KeyCatcherWord()*, nebo *KeyCatcherOutlook()*, který implementuje metody specifické pro jednotlivé doplňky. Součástí této třídy je i metoda *UpdateTaskPanes(int)*, která slouží k předávání informací o aktuální rychlosti psaní části uživatelského rozhraní,která bude popsaná v kapitole 4.3.5.

4.2.3 Třídy *KeyCatcher*, *KeyCatcherWord* a *KeyCatcherOutlook*

Abstraktní třída *KeyCatcher* a její potomci *KeyCatcherWord* a *KeyCatcherOutlook* jsou třídy určené k zachytávání jednotlivých událostí popsaných v kapitole 3.1. Tyto dvě třídy jsou potomky třídy *KeyCatcher*. Doplněk určený pro aplikaci Outlook obsahuje třídu *KeyCatcherOutlook* a doplněk určený pro aplikaci MS Word obsahuje třídu *KeyCatcherWord*. Třída *KeyCatcher* implementuje metody pro nastavení zachytávání stisků kláves a myši a třídy *KeyCatcherWord* a *KeyCatcherOutlook* obsahují metody pro detekci ostatních událostí. Instance třídy při svém vzniku vytvoří atribut typu *ConcurrentQueue<Information>*, což je fronta s možností sdílení více vlákny, bez nutnosti další synchronizace. Do této fronty se ukládají jednotlivé hodnoty struktury *Information*, která obsahuje informace o nastalé události. Dále se vytvoří v samostatném vlákně při vzniku instance třídy *KeyLetter*, které se předá ukazatel na vytvořený atribut typu *ConcurrentQueue<Information>*. Tímto se zajistí schopnost třídy *KeyCatcher* předávat detekované události dále ke zpracování se zajištěním nejmenší možné nepřesnosti v měření času mezi stisky kláves. Tato nepřesnost bude dále popsána v kapitole 4.4.

4.2.4 Třída *KeyLetter*

Třída *KeyLetter* nám slouží pro detekci základních akcí z jednotlivých událostí, dle algoritmů popsaných v kapitole 3.2.2. Tato třída vytváří instanci třídy *KeyProcess*. V této třídě běží nekonečná smyčka, která vybírá jednotlivé hodnoty typu *Information* ze sdílené fronty. Tyto hodnoty pak nadále zpracovává v metodě *TestLetters()*, ve které jsou umístěny jednotlivé algoritmy pro detekci editace. Tato metoda také volá metody třídy *KeyProcess* s parametry popisujícími detekovanou editaci. Jakmile projde jedna hodnota typu *Information* touto metodou, je předána instanci třídy *KeyProcess* pro další zpracování.

[1] Sezením budeme rozumět událost začínající spuštěním doplňku a končící jeho ukončením.

4.2.5 Třída *KeyProcess*

Účelem třídy *KeyProcess* je analýza psaní a editace a následné zpracování a uložení analyzovaných hodnot. Třída obsahuje metody pro zjištění současných analyzovaných hodnot, ukládání těchto hodnot a metody pro načítání a ukládání uživatelského nastavení do XML souborů. Výsledky analýz se uchovávají ve struktuře *SessionData*. Předávání hodnot o událostech této třídě je zajištěno metodou *RcvKey(Information)*, analýza současných údajů se provádí voláním metody *ComputeActual()*. Třída dále obsahuje metody *RcvLetter(int)*, *DelLetter(int)* a *RcvFix()*, které slouží k příjmu informací o detekované editaci. Informace o množství stisknutých kláves za posledních 5 sekund je pravidelně předávána metodě *UpdateTaskPanes(int)*, náležící třídě popsané v kapitole 4.2.2. Informace o množství stisknutých kláves za posledních 1000 sekund je možno získat metodou *int[] GetLastValue()*.

4.2.6 Třída *DataLoader*

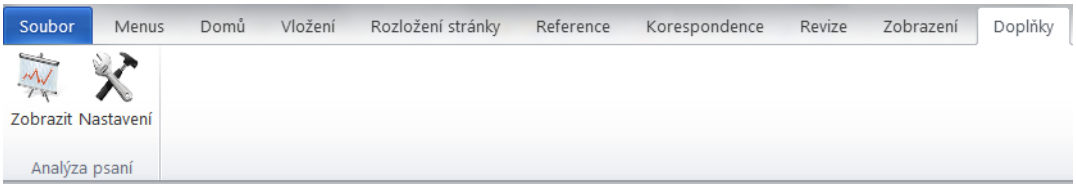
Třída *DataLoader* slouží k načtení a uchování dat o analýze minulých sezení. Data jsou uložena v souboru v textovém formátu a načítají se při spuštění doplňku. Po načtení těchto dat slouží tato třída ke komunikaci s většinou tříd uživatelského rozhraní a z toho důvodu obsahuje metody *Series GetSeriesAll(...)*, *Series GetSeriesLastAll(...)*, *Series GetSeriesDay(...)* a *Series GetSeriesLastDay(...)*. Tyto metody slouží k vytvoření instancí tříd *Series*, které jsou v uživatelském rozhraní využity k vykreslení jednotlivých grafů.

4.3 Uživatelské rozhraní

Uživatelské rozhraní aplikace bylo vytvořeno jako součást obou doplňků. Pro oba doplňky je uživatelské rozhraní shodné, s výjimkou okna popsaného v kapitole 4.3.4. Při tvorbě uživatelského rozhraní byly využity návrhy uživatelů popsané v kapitole 4.5, zvláště pak u nápovědy a zobrazení aktuální rychlosti psaní. Na obrázcích si teď popíšeme jednotlivá okna uživatelského rozhraní a jejich význam.

4.3.1 Přístup k uživatelskému rozhraní z aplikací MS Office

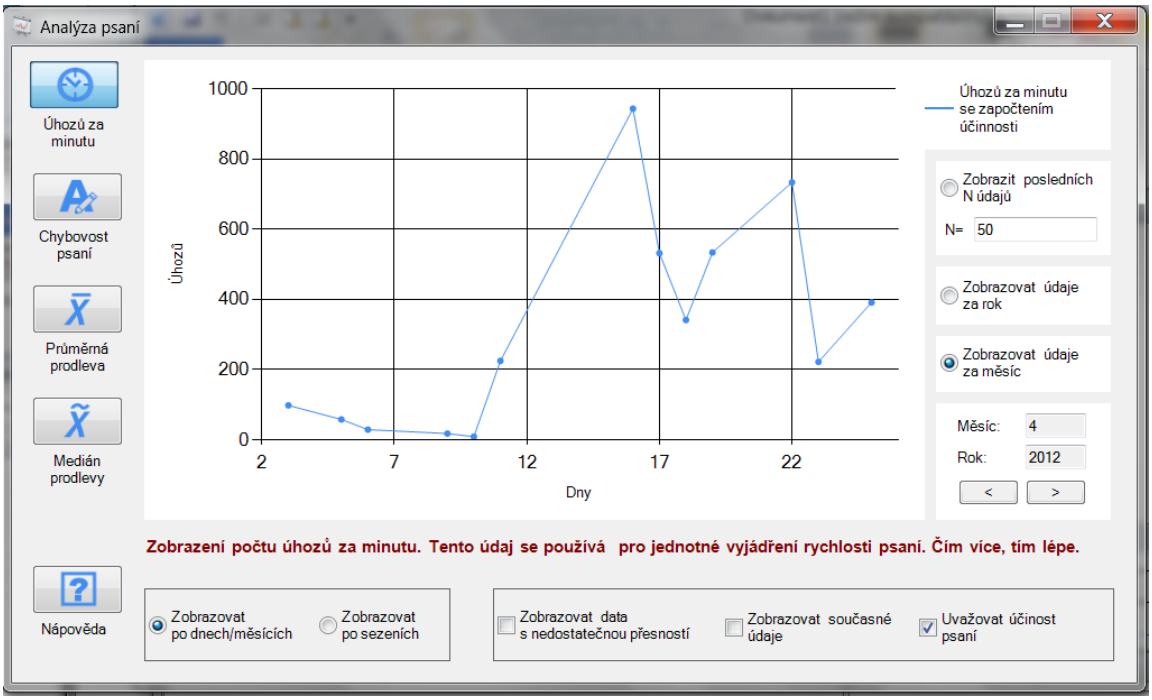
Na obrázku níže vidíme, že uživatelské rozhraní je přístupné z karty Doplňky. Hlavní okno tohoto rozhraní se spouští kliknutím na tlačítko Zobrazit. Kliknutím na tlačítko Nastavení se spouští okno používané pro nastavení doplňku. Popis grafické části tohoto přístupu ke zbytku uživatelského rozhraní je realizován ve zdrojovém kódu jako XML soubor.



Obrázek 4.4: Přístup k uživatelskému rozhraní doplňku z karty Doplňky.

4.3.2 Hlavní okno uživatelského rozhraní

Na následujícím obrázku vidíme hlavní okno uživatelského rozhraní. Toto okno má pevnou, uživatelsky neměnitelnou velikost, avšak tato velikost je nastavena tak, aby nepřesahovala velikost displeje současně vyráběných nejmenších notebooků (1024x600 px).



Obrázek 4.5: Hlavní okno uživatelského rozhraní.

Vidíme zde nalevo výběr témat zobrazovaných údajů a dole tlačítko nápovědy, jehož funkce bude vysvětlena v kapitole 4.3.3. Ve spodní a pravé části okna jsou umístěna různá nastavení, sloužící k přesnějšímu výběru a nastavení zobrazovaných dat. Význam a popisky těchto nastavení se mohou měnit dle vybraného tématu zobrazovaných údajů. U každého bodu lze zobrazit komentář k sezení, které bod reprezentuje. Hlavní část okna zabírá graf, který zobrazuje analyzované údaje v nastavené časové ose. Tento graf získává údaje komunikací se třídou *DataLoader*, tak jak bylo popsáno v kapitole 4.2.6. V uživatelském rozhraní zobrazujeme analyzované údaje vypsané v následující tabulce, tak jak byly popsány v kapitolách 3.2.1 a 3.3.5. Popisky však z důvodů lepší pochopitelnosti pro uživatele nemusí být stejné jako v kapitole 3.2.1.

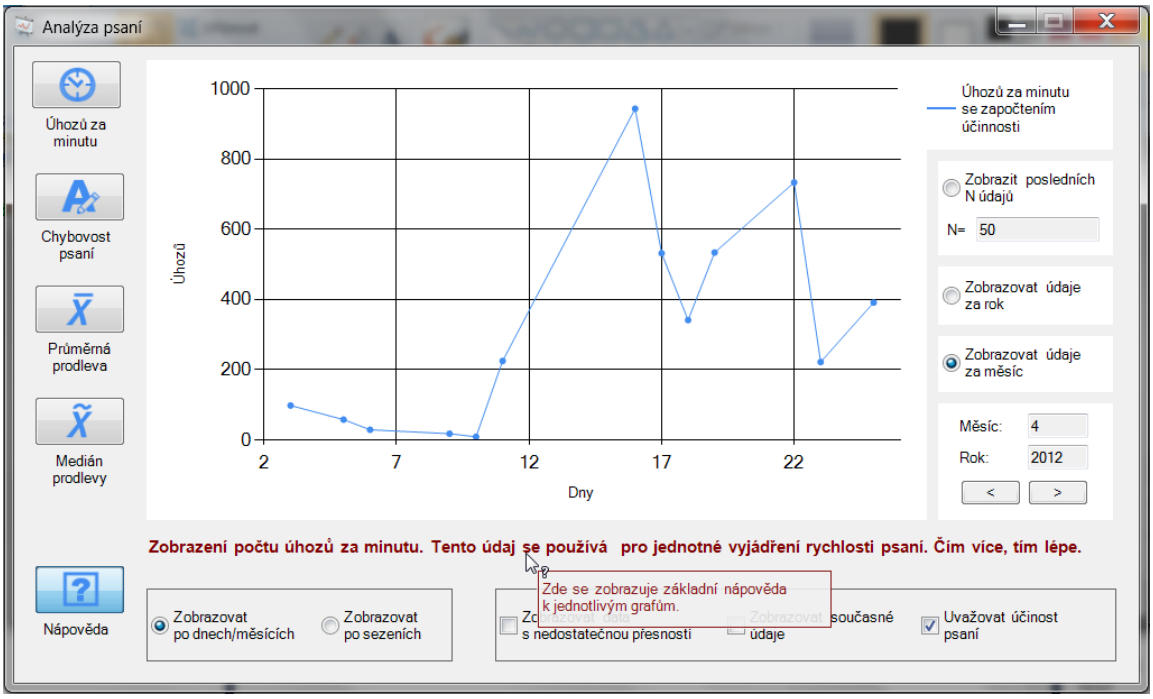
Počet úhozů za minutu.	Počet úhozů se započtením účinnosti.
Chybovost opravených znaků.	Využité pásmo.
Průměrná prodleva.	Odchylka od průměrné prodlevy.
Medián prodlevy.	Odchylka od mediánu prodlevy.

Tabulka 4.2: Seznam analyzovaných údajů zobrazovaných v uživatelském rozhraní.

Pod grafem se nalézá popisný text, který uživateli popisuje aktuálně zobrazený údaj a jeho význam.

4.3.3 Systém nápovědy k uživatelskému rozhraní

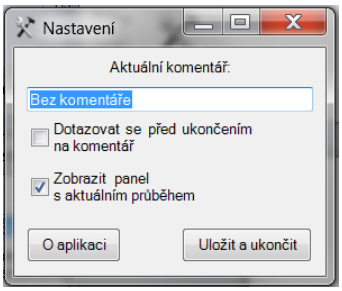
Systém nápovědy v doplňcích je spouštěn kliknutím na tlačítko Nápověda. Ukončen je kliknutím na libovolné jiné tlačítko v levém menu. Pokud je tento systém spuštěn, zobrazuje podrobný popis jednotlivých prvků při najetí myši nad daný prvek.



Obrázek 4.6: Nápověda.

4.3.4 Nastavení doplňků

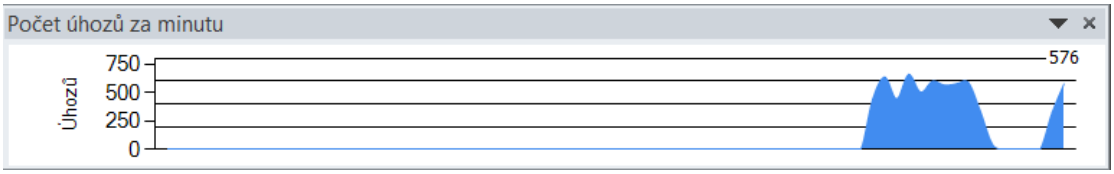
Okno Nastavení umožňuje nastavit uživatelské rozhraní doplňku a komentář k aktuálnímu sezení. Pokud nebude tento komentář zadán a je zatržena vrchní volba v okně nastavení, tak se z doplňku pro aplikaci MS Word zobrazí při ukončení aplikace okno, které vyzve uživatele k zadání tohoto komentáře. Toto okno po krátkém čase (5 s) samo zmizí, pokud není komentář zadán, a aplikace je ukončena. Popsané okno se nezobrazuje v doplňku aplikace Outlook z důvodu omezení schopnosti aplikace Outlook Ve verzi 2010 [**?**] oznamovat doplňkům ukončení aplikace a s tím souvisejícími doporučeními. Druhá volba umožňuje nastavit zobrazení panelu popsaného v kapitole 4.3.5.



Obrázek 4.7: Nastavení.

4.3.5 Zobrazení rychlosti psaní uživatele v reálném čase

Plovoucí panel zobrazení rychlosti psaní ukazuje rychlost psaní uživatele v průběhu několika posledních minut. Tento panel lze umístit mezi ostatní plovoucí panely a uživatel z něj vidí v grafu po pěti sekundách rychlost svého psaní. Údaje o aktuálním psaní jsou předávány přes třídy *ThisAddIn* a *Keyprocess*, tak jak bylo popsáno v kapitole 4.2.5.



Obrázek 4.8: Aktuální rychlost psaní.

4.4 Testování přesnosti detekce kláves

Při návrhu doplňků bylo potřeba uvažovat, jak přesné budou informace o aktuálním čase detekované doplňkem. Pokud by doplněk příliš zpomalil počítač, nebo by délka zpracování jedné klávesy byla příliš velká, mohlo by to vést jak k nepřesné analýze, tak i k nepoužitelnosti doplňku. Z tohoto důvodu byly provedeny dva druhy testů.

Prvním druhem testů bylo vytvoření doplňku, který pouze zjišťoval čas stisku klávesy bez dalších akcí a následná instalace tohoto doplňku. Několika (pěti) uživatelům byl předložen tento doplněk s otázkou, zda při jeho běhu pozorují rozdíl. Nikdo z uživatelů rozdíl nepozoroval a tudíž bylo předpokládáno, že i konečný doplněk by mohl být dostatečně rychlý. Tyto testy tedy pouze přibližně zjišťovaly, zda je možné doplněk vytvořit.

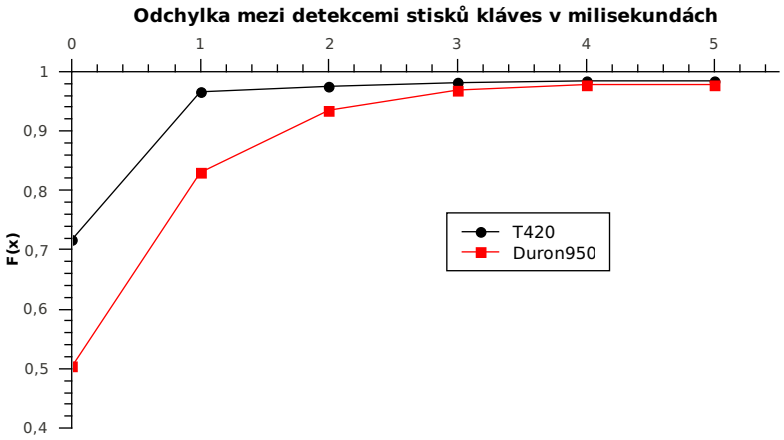
Druhý druh testů probíhal po dokončení implementace doplňků. Spočíval v měření rozdílu mezi okamžikem stisku klávesy detekovaným operačním systémem a okamžikem stisku klávesy detekovaným naším doplňkem pro aplikaci MS Word. Aplikace MS Word byla ve verzi 2010, 32-bitové. Pro zjištění času detekce operačním systémem byla použita funkce z kapitoly 2.1.2, nastavená pro detekci událostí stisku klávesy na úrovni operačního systému, která byla spuštěna ze samostatné aplikace. Tato metoda byla vybrána jako nejpřesnější dostupná metoda pro zjištění reálného okamžiku stisku klávesy. Tento druh testů probíhal na dvou různých počítačích, při co nejmenším počtu jinak spuštěných aplikací a za nasbírání alespoň 1000 vzorků. Základní parametry počítačů si zde uvedeme.

Název	**CPU**	**RAM paměť**	**Operační systém**	**Grafická k.**
T420	Intel Core $i5-2520M$	$6GB$	Windows 7 64 bit	Intel HD 3000
Duron950	AMD Duron 950	$512MB$	Windows XP 32 bit	GeForce2 MX

Tabulka 4.3: Základní parametry počítačů pro testování přesnosti detekce.

Můžeme si povšimnout značného rozdílu mezi použitými počítači. Tento rozdíl je záměrný a je z důvodu zajištění přesnosti detekce na co nejširším možném rozpětí počítačů.

Následující graf zobrazuje distribuční funkci rozdílu mezi námi naměřenými hodnotami pro oba počítače.



Obrázek 4.9: Graf distribuční funkce odchylky mezi detekcemi kláves.

Na grafu si můžeme všimnout, že většina (u obou počítačů přes 99%) naměřených hodnot se nelišila o více než 5 *ms*. Největší zjištěná hodnota odchylky byla 86 *ms* na počítači Duron950, avšak vzhledem k ojedinělosti této odchylky budeme považovat přesnost detekce kláves doplňky za dostatečnou. Uvedeme si zde pro přesnost tabulku popisující rozdíly mezi detekcemi na obou počítačích.

Počítač	**Průměrný rozdíl**	**Nejvyšší naměřená hodnota**
T420	$0,738ms$	$54ms$
Duron950	$1,182ms$	$86ms$

Tabulka 4.4: Průměrné rozdíly mezi detekcemi času stisků kláves.

4.5 Testování uživatelského rozhraní

Testování uživatelského rozhraní jsme si rozdělili na dvě fáze, abychom mohli v jedné fázi rychle opravovat případné chyby v implementaci a dodělávat možná vylepšení, a v další fázi testovat doplňky mezi větším počtem různých uživatelů.

4.5.1 První fáze testování

První fáze spočívala v testování za pomoci dvou dobrovolníků, kterým byly dodány první verze doplňků. Účelem této fáze bylo odstranit přehlédnuté nedodělky a zajistit základní přehlednost a snadnost použití doplňků. Komunikace s dobrovolníky probíhala denně a opakovaným zkoušením upravovaných verzí doplňků se tyto doplňky dokončovaly. Výsledkem této fáze byly nejenom úpravy grafické části doplňků, ale i například přidání nápovědy do obou doplňků.

4.5.2 Testování na větší skupině

Druhá fáze testování probíhala s doplňky upravenými během první fáze. Množství dobrovolníků (13) se rozeslaly doplňky s instrukcemi na vyplnění dotazníků ohledně uživatelského rozhraní doplňků. Tento dotazník sestával ze dvou druhů otázek, otázek s odpověďmi ve větách a otázek s odpověďmi číselnými. V otázkách s číselnými odpověďmi měl uživatel ohodnotit část doplňku číselnou stupnicí, používanou na základních a středních školách. Otázky s odpověďmi ve větách sloužily k přesnějšímu popsání názoru uživatele a k předání možných nápadů na vylepšení. Otázky s odpověďmi číselnými použijeme k celkovému posouzení uživatelského rozhraní doplňku. Instrukce k dotazníku k doplňku požadovaly po uživatelích, aby byl dotazník vyplněn po alespoň druhém použití uživatelského rozhraní doplňku. Všechny otázky dotazníku jsou uvedeny v příloze B.

Dobrovolníci pro druhou fázi testování byli vybráni tak, aby je bylo možno rozřadit dle dvou kritérií. Prvním kritériem byl věk a druhým kritériem byl průměrný počet úhozů za minutu, popsaný v kapitole 3.3.4. Druhé kritérium se zjišťovalo až při odevzdání dotazníků dle hodnot naměřených programem a rozřazovalo uživatele do dvou skupin, skupiny *rychlé* a skupiny *pomalé*. Skupina rychlých je skupina všech, kteří měli rychlost psaní vyšší než 200 úhozů za minutu, skupina pomalých obsahuje ostatní uživatele. Hraniční hodnota 200 úhozů za minutu byla určena z průměrné rychlosti psaní uživatelů (40) [**?**], kteří měli testy rychlosti psaní jako součást vstupních testů pro zaměstnání. Tato hodnota byla vyjádřena v počtu slov za minutu, popsaného v kapitole 2.2.1, pro naše potřeby si ji ale následovně zjednodušeně vyjádříme v počtu úhozů za minutu.

$$Počet\ úhozů\ za\ minutu = Počet\ slov\ za\ minutu \times 5$$

Zjednodušení tohoto vyjádření spočívá v myšlence, že jeden úhoz bude mít hodnotu jednoho znaku. I když díky tomuto zjednodušení není tento převod úplně přesný, pro naše potřeby však takto přesná hranice stačí. V následující tabulce si uvedeme rozřazení jednotlivých uživatelů do skupin dle jejich věku a rychlosti psaní.

	18—25 let	**25–40 let**
Rychlá skupina	4	3
Pomalá skupina	4	2

Tabulka 4.5: Rozřazení uživatelů do skupin.

Uvedeme si teď tabulku si použitými otázkami určenými k číselnému hodnocení grafického rozhraní doplňku a s průměrnou známkou od jednotlivých skupin.

	Přehlednost doplňků	**Grafická úprava doplňků**	**Grafická úprava nápovědy**
Rychlá skupina 18—25 let	$2,25$	$1,5$	2
Rychlá skupina 25—40 let	2	$1,5$	$2,5$
Pomalá skupina 18—25 let	$2,5$	$1,5$	$1,75$
Pomalá skupina 25—40 let	$2,67$	$1,67$	2

Tabulka 4.6: Průměrné hodnocení uživatelů.

Z těchto údajů můžeme vyvozovat, že uživatelské rozhraní doplňku je v praxi použitelné, i když je možné uvažovat o zlepšení přehlednosti doplňku. Textové otázky v dotazníku sloužily k přesnějšímu popsání uživatelských názorů a nápadů na vylepšení. Jeden z těchto nápadů byl zapracován do doplňků, a to zobrazení rychlosti psaní v reálném čase, popsané v kapitole 4.3.5. Kromě této úpravy pomohla tato fáze testování i k jiným, drobnějším úpravám, například k přidání zobrazení nápovědy po stisku klávesy *F1* v hlavním okně uživatelského rozhraní doplňku, popsaného v kapitole 4.3.2.

Kapitola 5

Závěr

Cílem této práce bylo navrhnout a implementovat nástroj pro analýzu psaní uživatele jako add-in, neboli doplněk, do aplikací MS Office. Tento cíl byl splněn vytvořením dvou takovýchto doplňků, včetně jejich uživatelského rozhraní. Při návrhu doplňku byly vytvořeny a použity metody pro analýzu jak rychlosti psaní uživatelů, tak i chybovosti psaní uživatelů. Při tvorbě těchto metod jsme vycházeli z již používaných metod, abychom zajistili jejich co největší bezchybnost. Tyto metody byly vytvořeny speciálně pro analýzu psaní různých uživatelů v produktech MS Word a Outlook tak, aby se zaručila jejich správná funkčnost pro dané aplikace při všech uvažovaných událostech.

Analyzované údaje jsou v uživatelském rozhraní zobrazeny ve formě grafů s časovou osou. Celkové zobrazení bylo navrhnuto s důrazem na dostatečnou přehlednost a pro zajištění zobrazení všech analyzovaných údajů ve formě srozumitelné méně i více pokročilým uživatelům.

Vytvořené uživatelské rozhraní doplňků bylo řádně otestováno uživateli a dle některých uživatelských připomínek upraveno. Výsledné uživatelské rozhraní bylo následně shledáno uživateli jako dostatečně přehledné a použitelné.

Jako další rozšíření této práce by bylo dobré vytvořit metody pro analýzu psaní uživatele i v ostatních aplikacích prostředí MS Office, případně metody pro analýzu textu v jiných aplikacích pro úpravu textu. Dalším možným rozšířením je přidání metody autentizace uživatele pomocí analýzy dynamiky psaní na klávesnici.

Příloha A

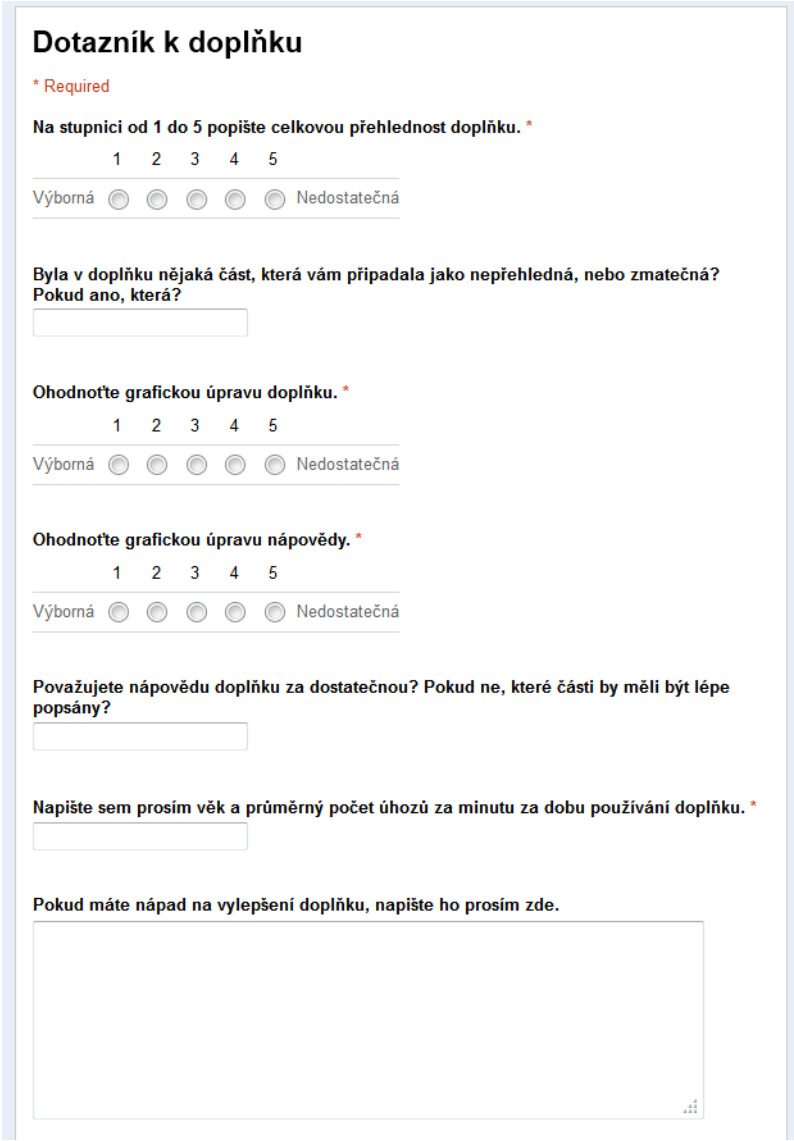
Obsah CD

Přiložené CD obsahuje několik souborů a adresářů, které jsou zde popsány.

`doc/`	Adresář se zdrojovými kódy k dokumentaci v LaTeXu.
`install/`	Adresář s instalačními soubory doplňků.
`src/`	Adresář se zdrojovými kódy k aplikaci.
`video/`	Adresář s videi ukazujícími práci doplňků.
`readme.txt`	Soubor s informacemi o aplikaci a o CD.
`install.txt`	Soubor s popisem instalace aplikace.
`bp.pdf`	Elektronická verze textu bakalářské práce.

Příloha B

Dotazník



Dotazník k doplňku

* Required

Na stupnici od 1 do 5 popište celkovou přehlednost doplňku. *

	1	2	3	4	5	
Výborná	○	○	○	○	○	Nedostatečná

Byla v doplňku nějaká část, která vám připadala jako nepřehledná, nebo zmatečná? Pokud ano, která?

Ohodnoťte grafickou úpravu doplňku. *

	1	2	3	4	5	
Výborná	○	○	○	○	○	Nedostatečná

Ohodnoťte grafickou úpravu nápovědy. *

	1	2	3	4	5	
Výborná	○	○	○	○	○	Nedostatečná

Považujete nápovědu doplňku za dostatečnou? Pokud ne, které části by měli být lépe popsány?

Napište sem prosím věk a průměrný počet úhozů za minutu za dobu používání doplňku. *

Pokud máte nápad na vylepšení doplňku, napište ho prosím zde.

Vytvořeno v Google Docs