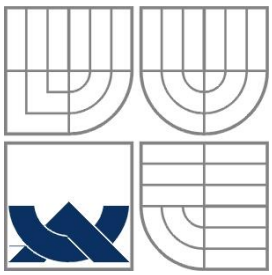
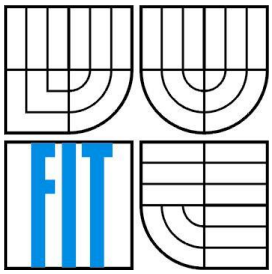


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

TITULKOVACÍ NÁSTROJ PRO BĚŽÍCÍ PŘEDNÁŠKU

TOOL FOR IMMEDIATE SUBTITLES IN RUNNING LECTURE

BAKALÁŘSKÁ PRÁCE
BACHELOR‘S THESIS

AUTOR PRÁCE — FILIP POKORNÝ
AUTHOR

VEDOUCÍ PRÁCE — Doc. Dr. Ing. DUŠAN KOLÁŘ
SUPERVISOR

BRNO 2016

Abstrakt

Tato práce se zabývá řešením problému Poradenského centra Alfons, který se týká jím poskytované služby, a to simultánního přepisu pro sluchově hendikepované studenty. Na základě pravidelné docházky do centra jsem provedl analýzu problému a následně navrhl řešení v podobě aplikace, kterou jsem následně naprogramoval. Tato aplikace má za cíl přejímat text od osoby realizující simultánní přepis a distribuovat ho studentovi v co nejvhodnější formě v doprovodu videopřenosu, který poskytuje obraz ze zařízení přednášejícího. Na závěr jsem aplikaci vyzkoušel se sluchově hendikepovanými studenty přímo ve výuce a následně jsem výsledky testování slovně ohodnotil.

Abstract

This thesis deals with solving problems of Alfons consulting center, specifically of their simultaneous transcript service for the hearing-impaired students. Thanks to my regular attendance in the consulting center, I analyzed the problem and designed an app solution that I then programmed. The goal of this app is to receive a text from people realizing the simultaneous transcript and give it to students in the most suitable form which they can use along with the speaker's video projection. The conclusion of the thesis shows the evaluation of my analysis after trying the app in lessons with deaf students.

Klíčová slova

Poradenské centrum Alfons, síťová aplikace, sluchově hendikepovaný studenti, RTMP, HLS, WebSocket, Open Broadcaster studio, Nginx

Keywords

Alfons consulting center, network application, hearing-impaired students, RTMP, HLS, WebSocket, Open Broadcaster studio, Nginx

Citace

Filip Pokorný: Titulkovací nástroj pro běžící přednášku, bakalářská práce, Brno, FIT VUT v Brně, 2016

Titulkovací nástroj pro běžící přednášku

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Doc. Dr. Ing. Dušana Koláře. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Filip Pokorný
14. května 2016

Poděkování

Na tomto místě bych rád poděkoval svému vedoucímu Doc. Dr. Ing. Dušanu Koláři za odbornou pomoc při tvorbě práce. Dále bych rád poděkoval své přítelkyni, která mě podporovala po celou dobu tvorby práce.

Obsah

1 Úvod

V dnešní době je společnost již velmi vyspělá, a proto se snaží poskytnout stejné podmínky k životu i zdravotně znevýhodněným lidem. Není tomu jinak ve školství. Zejména vysoké školy se snaží být obrazem pro celý školský vzdělávací systém daného státu a zakládají poradenská centra, která pomáhají hendikepovaným studentům. Tato centra se postupně vyvíjejí a rozrůstají, což láká hendikepované studenty vzdělávat se na dané škole. Z určitého hlediska by se dala posuzovat kvalita a vyspělost školy na základě existence těchto center při těchto školách. Konkrétně na Vysokém učení technickém v Brně již existuje poradenské centrum Alfons (dříve Přes bloky), které je mladé, a tedy nemá příliš velký vliv, zkušenosti a prostředky pro budoucí a stávající studenty. Na rozdíl od Masarykovy university, která se svým centrem Teiresiás patří mezi špičky v České republice.

Nemalou pomoc nabízí moderní technologie a informatika, které dokážou mazat rozdíly mezi normálními a znevýhodněnými studenty. Proto jsem se rozhodl v rámci završení mého bakalářského studia přispět k vývoji Poradenského centra Alfons. V rámci bakalářské práce jsem se tedy zaměřil na problém převodu mluveného slova do textové podoby pro studenty se sluchovým postižením, což je komplikace, se kterou se potýká centrum Alfons již dlouhou dobu. Zadavatel se snažil problém řešit různými dostupnými prostředky, ale výsledek byl vždy nedostačující. Rozhodl jsem se tento problém analyzovat, navrhnout řešení a následně ho realizovat. V rámci analýzy problému jsem komunikoval se zadavatelem, docházel na pracoviště a pokusil se zachytit co nejvíce informací k vypracování problematiky.

Samotná problematika se týká sluchově hendikepovaných studentů, kteří mají poruchu sluchu částečnou anebo úplnou. Studenti tedy neslyší vůbec, anebo slyší, ale za sebemenších zhoršených podmínek přestávají rozumět mluvenému slovu. Pro tyto studenty se musí provádět služba zvaná simultánní přepis. Jedná se o přepsání slovo od slova, co vyučující a reagující okolí vysloví.

Cílem bakalářské práce je zpracovat koncept aplikací, který bude přejímat text od člověka realizujícího simultánní přepis a distribuovat ho studentovi v co nejlepší formě, případně s dalšími informacemi, které napomůžou celou přednášku nebo cvičení lépe vstřebat. V návrhu je potřeba zohlednit dostupné prostředky a celkově celý kontext na Vysokém učení technickém v Brně.

Práce je rozdělena do sedmi kapitol. Druhá kapitola se zabývá technologiemi a použitelným softwarem, které byly pro vypracování práce použity. Popisované technologie a softwary jsou především síťové charakteru.

Ve třetí kapitole je detailně popsána problematika a potřeby Poradenského centra Alfons. Dále jsem v této kapitole popsal dosavadní praktiky a prostředky, které jsou zde využívány. Rovněž jsem zdůvodnil, proč jsou jednotlivé praktiky nevyhovující pro momentální situaci.

V další kapitole, tedy čtvrté, je vytvořen návrh konceptu aplikací, který má za cíl vyřešit popsanou situaci. Návrh je rozdělen na dvě části, kde první část popisuje účel jednotlivých prvků konceptu a druhá část popisuje princip a technologie k zhotovení jednotlivých prvků.

Následující kapitola popisuje implementaci jednotlivých částí konceptu. V rámci popisu jsem se zaměřil především na vytvořené uživatelské rozhraní a síťovou komunikaci. V rámci síťové komunikace jsou popsány všechny typy zpráv, které zasílají jednotlivé části mezi sebou. Dále jsem se zaměřil na popis důležitých algoritmů, které jsou klíčové pro chod jednotlivých aplikací. V popisu implementace serveru popisuji princip řízení celého konceptu.

Předposlední kapitola zahrnuje testování, jež je rozděleno na dvě části. První část popisuje pět zátěžových testů, které byly provedeny pro simulaci možných situací. Následující druhá část vypovídá o průběhu nasazení konceptu v reálném provozu, kde jsou také popsány možné komplikace, které mohou nastat v rámci kontextu VUT.

Pro svoji bakalářskou práci jsem si jako vedoucího zvolil Doc. Dr. Ing. Dušana Koláře. S panem Kolářem jsem pravidelně konzultoval řešení a problémy s ním spojené.

2 Použité technologie a software

2.1 Síťové protokoly

2.1.1 Transmission Control Protocol (TCP)

Protokol TCP je charakteristický pro svůj spolehlivý přenos. Jedná se o spojovanou službu. Před zahájením přenosu musí být ustanoveno spojení, které je dále udržováno. Jednotlivé zprávy jsou přenášeny, jako tzv. pakety, což je jednotka dat, která je sekvenčně rozdělena odesílatelem a následně opět složena příjemcem. Data předaná pro přenos jsou zapouzdřena do paketů, které jsou následně odeslány. Všechny pakety v rámci komunikace jsou číslovány, takže přijímací strana dokáže poznat, jestli přijaté pakety nebyly prohozeny, případně jestli nedošlo k výpadku některého z nich. Pokud dojde k výpadku, příjemce si může jednotlivé pakety znovu vyžádat. Protokol TCP je používán pro spolehlivý přenos v rámci sítě a využívá se v případech, kdy je zapotřebí spolehlivé doručení zaslaných dat. Jeho hlavní nevýhodou je rychlost, která je díky velkým režiím omezena. [1]

2.1.2 User Datagram Protocol (UDP)

Protokol UDP na rozdíl od TCP je nespojovaná služba a data jsou přenášena pomocí nesekvenčních datových jednotek, tzv. datagramů. V rámci přenosu dat není ustanoveno spojení, což má za následek nespolehlivý přenos. Ztrátu nebo případné prohození datagramů příjemce nezjistí, ale jednotlivé datagramy obsahují kontrolní součet, takže lze zjistit, jestli byla data přenosem narušena. Vzhledem k chybějící režii pro zajištění spolehlivého doručení dat je UDP rychlejší než TCP. Typickým příkladem použití jsou proudící přenosy dat, kde občasný výpadek dat nemá vliv na celkovou kvalitu. [1]

2.1.3 WebSocket

WebSocket je protokol umožňující obousměrnou komunikaci mezi jednotlivými TCP schránkami. Tento protokol je navrhnut pro komunikaci mezi webovým prohlížečem a serverem, která je založena na TCP protokolu, kde je provedeno ustanovení spojení (handshake) prostřednictví HTTP žádosti. WebSocket představoval jeden z prvků, který byl zaveden se specifikací HTML5 a stal se vhodnou metodou pro komunikaci v reálném čase, která snížila odezvu a objem přenášených dat. Do této doby se používala například metoda „Pollingu“. [2]

2.1.4 Real Time Messaging Protocol (RTMP)

RTMP je založen na TCP protokolu, který udržuje stále spojení, zajišťuje nízkou odezvu a co nejméně ztrátovou komunikaci. Datový tok je přenášen prostřednictvím paketů s dynamickou velikostí, tzv. chunků, jejichž délka je dohodnuta mezi klientem a serverem při nastavování počátečních hodnot připojení. Velikost těchto paketů bývá ustanovena na výchozích 64 bytů, ale lze ji navýšit i na 128 bytů. Standardně je tento protokol využíván k přenosu zvuku a obrazu, dále je nejčastěji provozován přes port 1935, ale výsledný použitý port je volně nastavitelný. Datové a kontrolní pakety jsou řetězeny kódovacím formátem Action Message Format (AMF). [3]

2.1.5 HTTP Live Streaming (HLS)

Jedná se o protokol vyvinutý společností Apple pro přenos zvukových a obrazových dat a je nejspíše jediným podporovaným formátem pro přenos, který zařízení této firmy podporují. Protokol HLS je od počátku otevřený a stal se alternativou k protokolu RTMP, jehož v poslední době vytlačuje.

Principem tohoto formátu je rozdělení přenášeného proudu do menších na sebe navazujících souborů formátu MPEG-TS. Z těchto souborů se sestavuje celý proud. Inteligentní přehrávač postupně podle potřeby stahuje a přehrává jednotlivé soubory. V případě proudu v reálném čase je seznam souborů opakovaně stahován a z něho jsou následně získávány odkazy na další nové soubory. [4]

2.2 Použitý software

2.2.1 Open Broadcaster studio

Open Broadcaster studio vychází z původní aplikace Open Broadcaster software, který byl kompletně přepsán a přizpůsoben pro více operačních systémů. Pro přepsání byly použity moderní metody psaní kódu a celková aplikace se stala optimalizovanější s mnoha novými funkcemi. Open Broadcaster studio je aplikace pro zachycení obrazu a jeho odesílání jako datový proud na zvolenou adresu, na níž se nachází server. Tento program je využíván především pro snímání počítačových her, kde je potřeba zachovat co nejlepší kvalitu, ale přitom dodržet rozumnou velikost proudících dat, jejichž velikost limituje internetové připojení. Proto má program velkou škálu nastavení kvality snímaného obrazu, možnosti úprav pořízených snímků za sekundu (fps) nebo zvolení vhodného kodeku. Mimo jiné lze v programu nastavit, co konkrétně se bude snímat. Uživatel může snímat celou obrazovku, výřez z ní anebo zvolené okno aktivního programu. Snímaný obraz lze následně ukládat do souboru nebo odesílat na zadanou adresu serveru prostřednictví protokolu RTMP.

Kromě samotného snímání obrazu lze nastavit spoustu speciálních efektů, jako např. vkládání různých grafických šablon a filtrů. Jednou z hlavních funkcí je i vložení textu, který se na snímaném obrazu iterativně opakuje. Text lze libovolně modifikovat barvou, velikostí, typem písma nebo i umístěním.

Aplikace je dobře vyvinutá a optimalizovaná. Mimo klasické instalace jsou k dispozici zdrojové soubory pod licencí Open source. Jsou podporovány operační systémy Windows (7, 8, 10), OSX a Linux. Tento projekt disponuje aktivním fórem, na kterém se řeší problémy všeho druhu. [5]

2.2.2 Nginx server

Nginx je softwarový webový server, který je schopný vytvořit HTTP a proxy server. Pracuje s protokoly HTTP (HTTPS), POP3, IMAP, SSL, SMTP. Jedná se o alternativu webového serveru Apach, který se považuje za těžkotonážní nástroj, na rozdíl od Nginx, který disponuje vysokým výkonem a malými nároky na paměť. Nginx je dostupný především na platformách Linux, ale existují i varianty pro Solaris, Mac OS a Windows.

Součástí Nginx je i RTMP modul, který nabízí podporu pro přenos zvukových a obrazových dat podporující protokoly RTMP, HLS, MPEG-DASH. V rámci těchto protokolů Nginx umožňuje přenos dat k více přehrávačům a zařízením jako je Adobe Flash player, iOS zařízení a Android telefony. Je podporován kodek H.264 pro video a ACC pro audio. Datový proud lze distribuovat pouze jedné stanici (unicast) i určité skupině stanic (multicast). Server disponuje asynchronní architekturou, díky

které se dosahuje vysokého výkonu a nízkých hardwarových nároků. Autorem tohoto modulu je Roman Arutyunyan. [6]

2.2.3 JW Player

Jedná se o velmi známý HTML přehrávač, který slouží k přehrávání audia a videa ve webovém prohlížeči. Tento přehrávač je podporován HTML5 a Adobe Flash Player standardem. Tato aplikace je open source a podporuje formáty FLV, H.264, MP4, VP8, AAC a MP3. Pro přenos jsou dále podporovány formáty RTMP, HLS a další. [7]

2.2.4 Qt

Jedná se aplikační rámec podporující více operačních systémů, jehož vývoj započal v roce 1990 norskými programátory Haavardem Nordem a Eirikem Chambe-Engem, kteří se rozhodli vytvořit tento objektově orientovaný grafický aplikační rámec. Původní společnost Trolltech byla postupně přeprodávaná a přejmenovávána novými vlastníky. Momentálně je původní společnost Trolltech přejmenována na The Qt Company a je celá vlastněna společností Digia Plc.

První verze aplikačního rámce Qt byla uvolněna v roce 1995 pod verzí Qt 0.90. Postupem času byl aplikační rámec vyvíjen a v nynější chvíli je dostupná verze 5.8. Celý koncept je od první verze vydáván jak pod licencí open source, tak i pod komerční licencí. Qt je k dispozici pod všemi možnými platformami, především Windows, Linux, iOS, Android, BlackBerry a další. Původně bylo Qt koncipováno pro jazyk C++, ale s přibývajícími verzemi bylo přizpůsobeno i pro další jazyky jako C#, Java a Python. [8]

2.2.5 MySQL databáze

Jedná se o databázový systém střední velikosti, který je schopen spravovat i velké objemy dat. Vyznačuje se velkou rychlostí, avšak na úkor omezené možnosti paralelního dotazování, absence vnořených dotazů a pohledů. I přes tyto nedostatky patří MySQL k nejrozšířenějším databázovým systémům, který podporuje většinu operačních systémů. Nejčastěji je používán v rámci webových aplikací. Komunikace s MySQL databází probíhá prostřednictvím jazyka SQL, který je doplněn o některá rozšíření. Tento databázový systém vlastní společnost Sun Microsystems, dceřiná společnost Oracle Corporation. Systém je k dispozici jak pod licencí GPL, tak pod komerční placenou licencí. [9]

3 Analýza problému

3.1 Zadavatel

Zadavatelem pro moji bakalářskou práci se stalo poradenské centrum Alfons, které je součástí Institutu celoživotního vzdělávání VUT v Brně a otevřeno bylo v roce 2007. Centrum zajišťuje poradenství a podpůrné služby pro uchazeče o studium a studenty se specifickými potřebami. [10]

3.2 Popis problému

Jednou skupinou hendikepovaných studentů jsou studenti s poruchou sluchu. Tito jedinci mohou být hendikepovaní částečně, kdy slyší, ale špatně doslýchají a sebemenší rušení jim dělá problém ve schopnosti rozumět mluvenému slovu, nebo úplně, kdy student neslyší zcela nic. Pro lidi s úplnou poruchou sluchu je potřeba třetí osoby, která jim překládá mluvené slovo do pro ně srozumitelné formy. Způsobů komunikace se sluchově postiženým je několik. Variantami jsou znakový jazyk, prstová abeceda nebo simultánní přepis.

Pro simultánní přepis je zapotřebí třetí osoba, která zvládá techniky rychlého psaní a převádí mluvené slovo do písemné formy na určitém zařízení, které dokáže snímat a následně zaznamenávat napsaný text. Student si poté přepsaný text čte a na základě tohoto textu by měl být schopen pochopit vše, co bylo v daném okolí vyřčeno. V případě, že student sám chce komunikovat s okolím a předat mu nějakou informaci, musí danou informaci sdělit přepisující osobě prostřednictvím znakového jazyka, pokud tuto techniku osoba ovládá, nebo napsat danou informaci na papír či jiný prostředek.

Týdně jsou uskutečněny desítky hodin přepisů pro sluchově hendikepované studenty na VUT. Do budoucna se předpokládá nárůst studentů s těmito potřebami a zároveň zvýšení počtu přepsaných hodin.

V době řešení problému je nesnadnou záležitostí vykonávat službu přepisu tak, aby byla maximálně efektivní jak pro studenta, tak pro vykonavatele přepisu, tzn. přepisovatele. Problémem je absence programu, který by pohodlně přebíral text napsaný přepisující osobou a v co nejlepší formě ho prezentoval studentovi.

3.3 Stávající řešení

Poradenské centrum Přes bloky zaměstnává více lidí, kteří vykonávají simultánní přepis, ale není pevně stanoveno, jakým způsobem tuto službu vykonávat. Důvod této situace je absence řešení, které by bylo pro každého přepisovatele vyhovující. Poradenské centrum doporučuje použití programu Polygraf, který však většina přepisovatelů na VUT odmítá.

3.4 Přepis prostřednictví aplikace Polygraf

Polygraf byl vyvinut Masarykovou univerzitou, konkrétně jeho střediskem Teiresiás, které má stejné zájmy jako Poradenské centrum Alfons.

Základní použití Polygrafu je přenos všeho, co přepisovatel napíše do svého zařízení, na zařízení studenta, které má operační systém iOS nebo Android. Studentovi se na jeho přístroji

zobrazuje přepsaný text. Tato základní funkcionalita byla vytvořena pro potřeby simultánního přepisu. Další funkcionalita programu Polygraf spočívá v přenosu obrazu vyučujícího na zařízení studenta přes aplikaci umístěnou na zařízení vyučujícího. K tomu má vyučující možnost přidat přepis jako titulky, které jsou tvořeny přes aplikaci na zařízení přepisovatele. Polygraf je tedy tvořen ze čtyř částí, které se nazývají: Polygraf Word AddIn, Polygraf, Polygraf Broadcaster a Polygraf Captions.

3.4.1 Polygraf Word AddIn

Je instalován samostatně jako doplněk do editoru MS Word a slouží ke sdílení přepisu se zařízeními studentů. Přepisovateli stačí po nainstalování této části zapnout jeho MS Word a začít psát. Připojeným studentům se potom přeposílá obsah napsaného dokumentu.

3.4.2 Polygraf Broadcaster

Přenos obrazu vyučujícího je možné uskutečnit dvěma způsoby. Jeden z nich je instalace speciálního hardwarového zařízení, které je připojeno na VGA nebo DVI výstup grafické karty a všechen obsah přeposílá do sítě.

Druhý způsob je instalace aplikace Polygraf Broadcaster, která snímá obsah obrazovky softwarově a přeposílá všechen obsah opět do sítě.

3.4.3 Polygraf Captions

Tato aplikace je doplněk k aplikaci Polygraf Broadcaster, jejímž účelem je komunikovat s aplikací Polygraf Word AddIn a přidávat na obrazovku zařízení přednášejícího titulky. Obraz je potom snímán i s titulky.

3.4.4 Nedostatky

Na první pohled se zdá, že Polygraf je ideální program pro uskutečnění simultánního přepisu, ale přesto od něj většina přepisovatelů dává ruce pryč. Hlavním problémem je, že program Polygraf Word AddIn po nějaké době zcela zpomalí zařízení, na kterém je služba vykonávaná. Tím pádem se zhoršuje přenos textu studentovi, data dochází s čím dál větším zpožděním a přepisovateli působí problémy při ovládání jeho zařízení. Jak pro studenta, tak pro přepisovatele je situace velmi nepříjemná. Důsledkem toho se snižuje kvalita přepisu. Míra obtíží je taková, že přepisovatel ani student nechtějí nadále program využívat.

Pro studenta je velmi náročné číst přepsané slovo a zároveň sledovat, co se promítá na plátno, a ještě k tomu, co vyučující píše na tabuli. Proto je velmi užitečný přenos obrazu vyučujícího s titulky. Když opomeneme předchozí problémy Polygrafu Word AddIn, nastávají obtíže s tím, že titulky uvidí všichni studenti na plátně a navíc i přednášející na svém zařízení. Potom je pro vyučujícího velmi obtížné, když se při ukazování schématu přes dataprojektor potýká s přimíchanými titulky. Ve výsledku se zlepšuje hodnota přepisu pro jednoho až dva hendikepované studenty, ale zhoršuje se kvalita přednášky pro sto ostatních studentů bez postižení a pro jednoho vyučujícího. Tato situace je neakceptovatelná, a proto je přenos obrazu s titulky zcela nepoužitelný.

3.4.5 Přepis prostřednictvím externí klávesnice

Po odmítnutí aplikace Polygraf se snaží přepisovatelé najít způsob, jak službu vykonávat. Jednou z možností je k zařízení studenta, což bývá většinou tablet s operačním systémem iOS nebo Android,

připojit externí klávesnici. V praxi to vypadá tak, že student má před sebou tablet a přepisovatel přes externí klávesnici provádí simultánní přepis. Nevýhodou této metody je, že se jí neřeší problém roztříštěností informací pro studenta. Dále bývá problém v tom, že přepisovatel není schopen psát, aniž by viděl, co píše. Proto je tato metoda vhodná pouze pro úzký kruh přepisovatelů.

3.4.6 Přepis prostřednictvím jedné obrazovky

Další metoda je ze všeho nejjednodušší, ale zároveň nejprimitivnější. Princip spočívá v umístění zařízení přepisovatele mezi něj a studenta (výhradně je používán notebook). Přepisovatel má natažené ruce na klávesnici a student vidí na obrazovku. Tato metoda je ale náročná pro oba jedince. Navíc opět není řešeno roztříštění informací určených studentovi, ke kterému se ještě přidávají nesnáze při orientaci v textu, který je od studenta přirozeně ve větší vzdálenosti.

4 Návrh nového řešení

4.1 Shrnutí požadavků a cílů

Cílem nově vytvořené aplikace je získat obraz ze zařízení vyučujícího a obohatit ho o přepsaný text v podobě titulků. Výsledek se následně bude prezentovat na zařízení studenta, který by měl ve výsledku dostat něco, co připomíná film s titulky. Touto aplikací se docílí centralizace informací na jedno místo. Tím se neslyšícímu studentovi ulehčí vstřebání informací a lepší spojení kontextu mezi vizuální a obsahovou stránkou přednášky.

Dalším cílem aplikace je co nejmenší zásah do výuky učitele. Vše by mělo být uděláno tak, aby vyučující v ideálním případě nevěděl, že se něco děje. Je zapotřebí vše co nejvíce zautomatizovat. Řídícím prvkem bude samotný přepisovatel, který inicializuje komunikaci. Přepisovatel je v daném kontextu osoba, která má nejvíce zkušeností a případně je zaškolená pro práci s vytvořenou aplikací. Navíc je to osoba, jejímž hlavním zájmem by mělo být co nejkvalitnější vykonání služby. Proto celý koncept nastavuje a ovládá. Ostatní účastníci přepisu mají jiný bod zájmu. Proto je potřeba, aby spuštění a následný chod byly co nejvíce autonomní. V kontextu tohoto řešení je zapotřebí navrhnout i autorizaci přepisovatelů. Bylo by nežádoucí, kdyby byl systém narušován anonymními přihlášeními, která by mohla vyvolávat nežádoucí operace.

Jedná se o koncept aplikací, který má za úkol napomáhat k výkonu jedné ze služeb poradenského centra. Tato služba je vykonávána lidmi, jejichž práce je následně finančně ohodnocena, a proto je zapotřebí sledovat a dokumentovat kvalitu provedené služby. Dále se okolo této služby neustále mění lidé, ať ze strany studentů nebo personálu. Na základě těchto poznatků je v rámci konceptu potřeba vytvořit jednoduché administrativní prostředí a registraci vykonaných služeb, případně jejich obsah.

Pro vykonání služby musí být přepisovatel fyzicky přítomný na přednášce, ale v běžné praxi by to tak nemuselo být. Stačí, když se pořídí zvukový záznam z probíhající přednášky a v reálném čase bude přenesen kamkoliv, kde přepisovatel právě pobývá. Pro přenos postačí mobilní telefon s externím mikrofonem připojeným přes Bluetooth. Proto je dobré myslet na budoucnost a navrhnout řešení, které by tuto variantu umožňovalo.

4.2 Abstraktní návrh

4.2.1 Přepisovatel

Přepisovatel má k dispozici aplikaci, která funguje jako textový editor. Po spuštění aplikace je nutno vyplnit přihlašovací jméno přepisovatele a heslo. Poté se aplikace připojí na server, v rámci něhož je vystaven nový záznam o právě probíhajícím přepisu. V praxi je na webových stránkách zveřejněno jméno přepisovatele. Student může následně přistoupit na příslušnou stránku, z níž bude odebírat obsah.

Aplikace přepisovatele připomíná textový editor, do kterého lze psát text. Obsah editoru bude snímán a zobrazován studentovi ve dvou formách, buď jako samotný textový proud, anebo textový proud doprovázený videopřenosem ze zařízení přednášejícího.

Přepisovatel si může zobrazit nabídku přednášejících, kterou mu poskytne server, a připojit se k jakémukoli zařízení přednášejícího. Student potom uvidí záznam obrazovky přednášejícího s vloženými titulky reprezentujícími simultánní přepis.

Editor pro psaní bude koncipován tak, aby napsaný text bylo možné libovolně modifikovat pro pohodlné psaní. V rámci těchto přizpůsobení lze měnit velikost, případně typ písma textu, podle potřeby. Odeslaný text se v editoru bude obarvovat pro lepší orientaci přepisovatele.

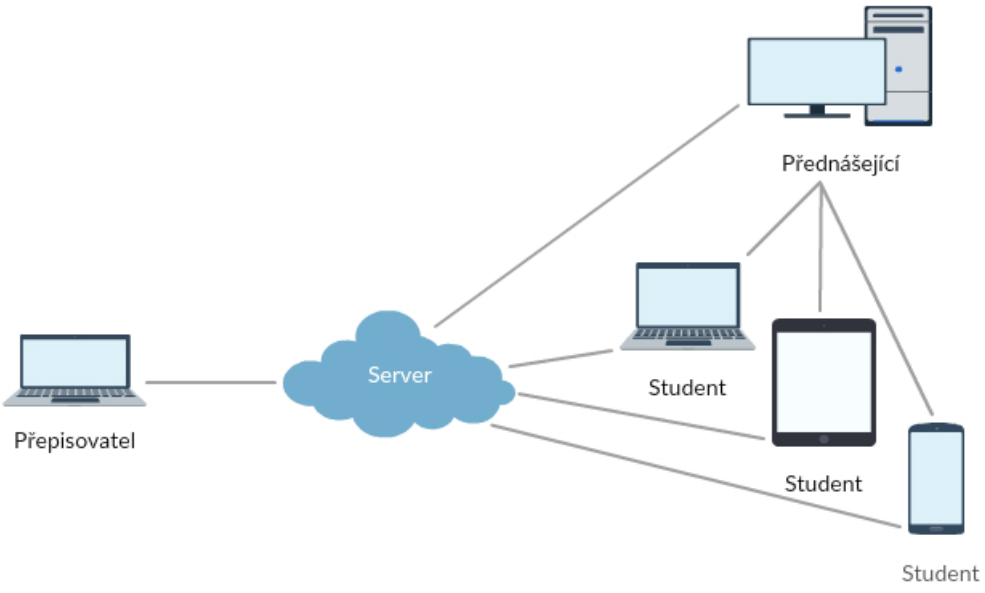
4.2.2 Přednášející

Role přednášejícího spočívá v poskytnutí obsahu své obrazovky, kde je promítán podpůrný materiál k výuce. Obsah je poskytnut v podobě videopřenosu, který je v reálném čase distribuován studentovi.

Přednášející má na svém zařízení nainstalovanou aplikaci, která je aktivována spuštěním uživatelem. Po spuštění je zobrazeno okno s nabídkami operací a současným stavem aplikace. Aplikace nadále pracuje zcela autonomně a není potřeba další obsluha ze strany vyučujícího. Po spuštění lze okno aplikace minimalizovat a následně k němu lze přistoupit ze skrytých ikon. Tím pádem by aplikace dále neměla nijak narušovat chod výuky a přednášející nemusí myslet na jakoukoliv obsluhu. Avšak v případě potřeby lze do aplikace zasáhnout. Zasahovat do chodu aplikace lze ze skrytých ikon, které jsou přístupné v dolní liště. Ve skrytých ikonách se bude nalézat minimalizovaná aplikace, reprezentovaná příslušnou ikonkou, po kliknutí pravým tlačítkem lze vybrat operace, nebo se informovat o aktuálním stavu.

4.2.3 Student

Student může použít jakékoliv zařízení, které má připojení k internetu a webový prohlížeč. To jsou především zařízení jako notebooky, tablety a mobily s libovolným operačním systémem. Student se k aplikaci připojí tak, že otevře svůj webový prohlížeč a vstoupí na webové stránky, kde bude vypsán seznam právě probíhajících přepisů. Jednotlivé přepisy budou identifikovány podle jména přepisovatele, který službu vykonává a který se autorizoval před začátkem přepisu.



Obrázek 4.1: Schéma konceptu

Student má k dispozici dvě varianty zobrazení přepisu. První z metod je zobrazení pouhého textového proudu. Po vybrání této varianty je studentovi zobrazena příslušná stránka. Na této stránce se nachází okno, ve kterém se zobrazuje tvořený text přepisovatelem. Zobrazovaný text lze libovolně upravovat podle potřeby. Druhou variantou je zobrazení textového proudu v doprovodu videopřenosu

z obrazovky přednášejícího. Tyto dvě varianty jsou dostupné po přístupu na příslušnou stránku, kde je videopřenos zobrazen klasicky ve video přehrávači a text je zobrazován pod oknem přehrávače v příslušném poli navázaném co nejblíže přehrávači. Ve výsledku student vidí obsah obrazovky vyučujícího a tvořený text vedle sebe. Schéma celého konceptu lze vidět na obrázku 4.1.

4.3 Praktický návrh

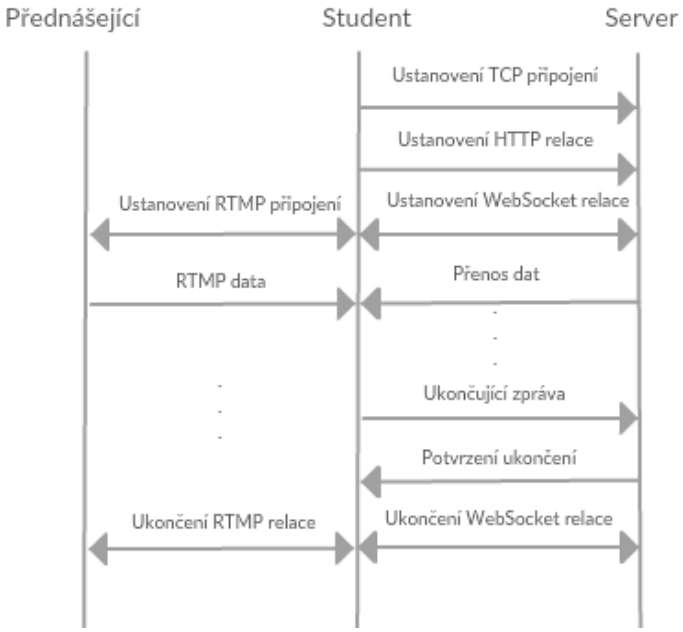
4.3.1 Prostředí studenta

Je důležité, aby aplikace byla pro studenta velmi flexibilní a použitelná skrze více operačních systémů. Student může pro přehrávání videopřenosu využít tablet, telefon nebo notebook bez jakýchkoliv předchozích příprav. Aby se daly všechny tyto požadavky splnit, rozhodl jsem se jako klienta zvolit webový prohlížeč, který musí podporovat HTML5. Webový prohlížeč je samozřejmostí skoro každého zařízení, které disponuje připojením k internetu, tím pádem se student nemusí zabývat instalací speciálního klienta pro příjem dat. Student jednoduše může přijít s jakýmkoliv zařízením, které vyhovuje specifikaci, a připojit se ke službě.

Student se ke službě připojí po přístupu k příslušným webovým stránkám. Tyto stránky budou vyžadovat autorizaci, kterou lze implementovat za pomoci jednoduché HTTP autentizace. Server vyzve klienta pomocí protokolu HTTP, aby zaslal v rámci požadavku na stránku také autentizační informace, tedy jméno a heslo. Po úspěšném přihlášení si student může vybrat z aktivních přepisovatelů a jejich nabízených služeb.

V případě přijímání textového proudu lze využít protokolu WebSocket, který je součástí HTML5. Stránka studenta obdrží síťovou adresu serveru a port, na který se připojí přes protokol WebScoket, a kde si zažádá o požadavek pro příjem dat od určitého přepisovatele. Identifikace přepisovatele je uvedena při předávání parametru stránky. Nadále student přijímá data od serveru.

Pokud je textový proud doprovázen i videopřenosem, potom je součástí stránky i video přehrávač, kterému byla předána RTMP adresa, prostřednictvím které lze zažádat o příjem videopřenosu. Tento požadavek je zaslán aplikaci přednášejícího, která na základě požadavku začne zasílat videopřenos. Popis komunikace je znázorněn na obrázku 4.2.



Obrázek 4.2: Schéma komunikace aplikace studenta

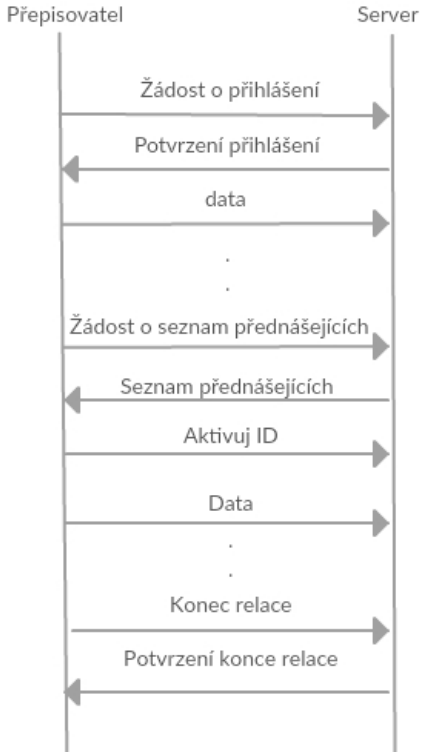
4.3.2 Aplikace přepisovatele

Tato aplikace je nejdůležitější částí celého konceptu. Skládá se z jednoduchého textového editoru, který má v sobě naimplementovanou síťovou komunikaci a potřebné funkce. Tato aplikace komunikuje pouze se serverem, který zpracovává odesílaná data a požadavky. Pro připojení k serveru je potřeba znát síťovou adresu nebo doménové jméno a port. Tyto údaje jsou volně nastavitelné uživatelem. Důležitou částí je samotné grafické rozhraní, které musí být uživatelsky přívětivé. Proto jsem se rozhodl tuto aplikaci naimplementovat v OOP v jazyce C++ s podporou Qt aplikačního rámce.

Po spuštění textového editoru je zadán první vstup v podobě přihlašovacích údajů. Program se přes nastavenou síťovou adresu pokusí spojit se serverem a zadané přihlašovací údaje zašle serveru, který v případě úspěchu provede registraci klienta (přepisovatele), v opačném případě bude uživatel upozorněn na neplatné přihlašovací údaje. Pokud nebude navázáno spojení se serverem, uživatel je na neúspěch upozorněn. Veškerá komunikace se serverem je realizována přes protokol UDP. Protokol UDP jsem zvolil z důvodu menšího zatížení sítě a koncept zpráv v podobě datagramů mi přišel vhodnější pro server, který zpracovává více klientů najednou. Zvolením UDP protokolu však vyplývá povinnost ošetřit možnou ztrátu dat, čí prohození jednotlivých datagramů. Opatření proti ztrátě a prohození datagramů lze předejít číslováním jednotlivých datagramů a následným potvrzováním ze strany příjemce.

Po úspěšné autorizaci lze začít tvořit simultánní přepis. Psaný text je snímán a po menších částech, standardně slovech, odesílán na server. Server text zpracovává a přeposílá na zařízení registrovaných studentů.

Pro připojení videopřenosu přednášejícího, vyvolá přepisovatel příslušnou funkci, která zašle požadavek na server. Server odpoví seznamem aktuálních volných videopřenosů, z kterých přepisovatel jeden vybere. Seznam obsahuje jméno přednáškové místnosti nebo jiný identifikační textový řetězec, unikátní identifikační číslo a čas registrace vyučujícího na serveru. Získaný seznam je prezentován skrz nové okno. Po zvolení příslušného přednášejícího je volba zaslána na server, který požadavek zpracuje. Popis komunikace je znázorněn na obrázku 4.3.



Obrázek 4.3: Schéma komunikace aplikace přepisovatele

4.3.3 Aplikace přednášejícího

Aplikace pro přenášejícího se skládá ze tří částí, a to z řídícího programu, upraveného OBS studia a Nginx serveru. Řídící program má především za úkol komunikovat se serverem, plnit jeho požadavky, případně informovat o stavu. Dále nastavovat/korigovat OBS studio a Nginx server. Pro registraci na server je zapotřebí síťová adresa nebo doménové jméno a port serveru, dále je nutno vyplnit jméno učebny, ve které se zařízení nachází, nebo jiný textový řetězec pro snadnější identifikaci. Tyto informace jsou volně nastavitelné uživatelem. Před registrací je zjištěna lokální síťová adresa, adresa zařízení, která je také součástí registrace. Na tuto adresu jsou klienti odkazováni při žádosti o videopřenos. Po úspěšné registraci aplikace naslouchá a čeká aktivaci od serveru. Po aktivaci řídicí prvek nastaví Nginx a OBS studio pro vysílání videopřenosu.

V případě aktivace je zapnut OBS, který snímá obrazovku, a přes RTMP je obsah předáván Nginx serveru. Nginx server přijímá registrace od zařízení studentů. Pokud nějaký ze studentů projeví zájem o videopřenos, potom je mu obsah přes protokol RTMP prostřednictví Nginx serveru přeposílán.

4.3.4 Server

Pro server je potřeba zařízení s veřejnou síťovou adresou, na kterou se lze dlouhodobě odkazovat jednotlivými aplikacemi. Server reprezentuje spojovací prvek, jenž zároveň řídí a nastavuje jednotlivé součásti. Server je rozdělen do dvou částí, jejichž data propojuje MySQL databáze. První část serveru je klasický HTTP server, který poskytuje data v podobě webových stránek. Duhou částí je aktivní démon komunikující s aplikacemi přepisovatele, studenta a přednášejícího.

Účelem HTTP serveru je provádět autorizaci a následně generovat webové stránky reprezentující aplikaci studenta. Tyto stránky jsou sestavovány pomocí jazyka PHP a podle potřeby jsou vyplňovány údaji jako síťová adresa videopřenosu nebo síťová adresa serveru. Zabezpečení lze realizovat prostřednictvím komunikace přes síťový protokol HTTPS, který je nástavbou síťového protokolu HTTP, jenž poskytuje šifrovanou komunikaci na základě certifikátu. Nastavení protokolu HTTPS poskytuje většina webových poskytovatelů na základě požadavku zákazníka.

Další částí, kterou HTTP server poskytuje, je administrativní prostředí, jež pracuje s databází MySQL. K tomuto prostředí má přístup pracovník poradenského centra. V rámci prostředí lze vytvářet, upravovat, mazat účty jednotlivých studentů a nebo přepisovatelů. V rámci účtů studentů, je zapotřebí zaznamenávat jenom uživatelské jméno a heslo. Pro vytvoření účtu přepisovatele je potřeba zadat: jméno, příjmení, uživatelské jméno a heslo.

Dále se zde přihlášený pracovník dozví historii prováděných služeb a to, co v průběhu těchto služeb bylo vytvořeno. Záznam o službě by měl obsahovat: unikátní identifikační číslo přepisovatele, datum, čas počátku, čas konce, počet pořízených znaků a jméno souboru, pod kterým se obsah přepisu nachází. Schéma databáze je viditelné v příloze číslo 2.

Druhou částí serveru je aktivní démon, který naslouchá ve vytvořené UDP schránce, kde vyčkává na komunikaci ze strany přednášejícího a přepisovatele. Začátek komunikace začíná autorizaci přepisovatele nebo přednášejícího. Přednášející je autorizován bez hesla, tudíž jako přednášejícího lze registrovat kohokoliv, kdo má přístup k aplikaci. V rámci registrace je serveru sdělena učebna, ve které se přednášející nachází, a lokální síťová adresa zařízení. Na základě těchto informací je vytvořen záznam v paměti serveru.

Registrace přepisovatele je prováděna na základě uživatelského jména a hesla. Validitu těchto údajů si server ověřuje v MySQL databázi. Po úspěšné registraci server přijímá data od přepisovatele, zaznamenává je a dále přeposílá příslušným studentům. Po skončení relace jsou přenesená data uložena do souboru a název souboru uložen s informacemi o celé relaci do MySQL databáze. Nově vytvořený záznam o provedené službě je vázán na konkrétního přepisovatele. V databázi u každého přepisovatele

je k dispozici příznak samotné aktivity a aktivity videopřenosu. Tyto příznaky jsou v reálném čase aktualizovány, takže HTTP server má přístup k aktuálnímu stavu skrz databázi. Krom příznaků je zaznamenávána síťová adresa zařízení přednášejícího v případě navázání videopřenosu na textový proud.

Kromě UDP schránky je dále vytvořena WebSocket schránka, která slouží ke komunikaci se zařízením studenta. Na této schránce jsou přijímány registrace studentů. V rámci registrace studenta je stanoveno, od kterého přepisovatele student data odebírá. Po úspěšné registraci dat nadále proudí pouze směrem od serveru ke studentovi.

Krom komunikace a zpracování zpráv je potřeba udržovat kontakt s jednotlivými klienty a kontrolovat jejich aktivitu. V případě odpojení nějakého klienta je potřeba celý koncept, zejména části navázané na odpojeného klienta, o této události zpravit. Změny musí být reflektovány i do databáze, na základě které pracuje HTTP server.

5 Implementace

V této kapitole se budu zabývat implementací jednotlivých dílčích aplikací, tedy aplikací pro přepisovatele, přednášejícího, studenta, řídicí server a administrativní prostředí. Celková implementace je celkově rozsáhlá a obsahuje spoustu algoritmů. Proto zde nebudu zabíhat do příliš velkých detailů, spíše se budu zaměřovat na klíčové části. Mezi klíčové části patří především síťová komunikace, logika jednotlivých důležitých algoritmů a grafického rozhraní s možnostmi, které nabízí.

Celková aplikace je složena z 5 částí, které mezi sebou komunikují různými prostředky. Nejčastěji se využívá síťová komunikace a jinak je použita databáze MySQL. V rámci síťové komunikace jsou použity protokoly UDP, WebSocket, RTMP a HLS. V jednotlivých kapitolách je odůvodněno použití vybraných síťových protokolů. V celém konceptu bylo také využito více programovacích jazyků, jsou to zejména C++, PHP a Javascript. Kromě programovacích jazyků bylo využito Qt aplikačního rámce pro tvorbu uživatelského prostředí. Tento aplikační rámec jsem použil v kombinaci s jazykem C++.

5.1 Implementace aplikace pro přepisovatele

Aplikace přepisovatele je naprogramována v jazyce C++ za použití Qt aplikačního rámce pro operační systém Windows. V případě potřeby je možné aplikaci předělat na jiný operační systém za předpokladu, že budou předělány algoritmy pro práci se sítí, zejména sokety, z důvodu nynějšího použití knihoven pouze pro OS Windows.

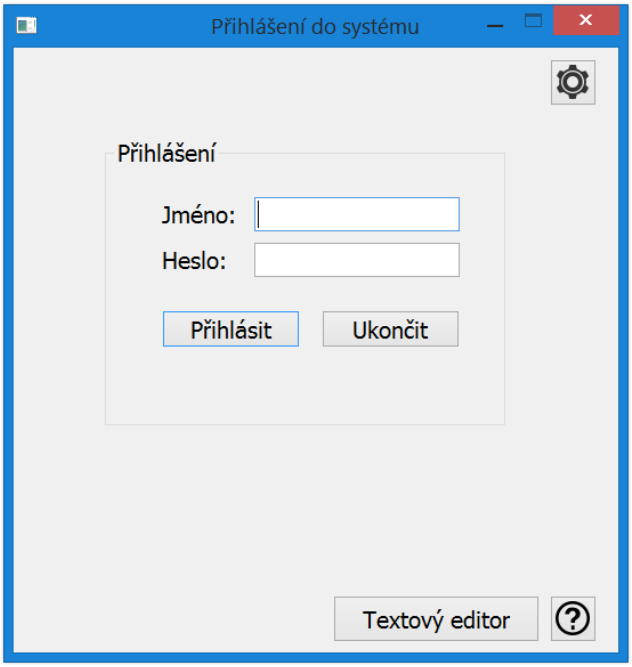
Aplikační rámec Qt primárně využívá sdílené kompilace (shared compilation). Tento fakt vyžaduje instalaci Qt knihoven do každého zařízení, na kterém se aplikace spouští. Proto jsem musel překompilovat knihovny Qt do nastavení statické kompilace.

5.1.1 Grafické rozhraní

Aplikace se skládá z více oken, která se uživateli zobrazují a skrývají podle aktuálního stavu. Po spuštění je zobrazeno úvodní okno vyzývající uživatele k přihlášení do systému, jak lze vidět na obrázku 5.1. Nachází se zde dvě políčka pro zadání uživatelského jména a hesla. Přihlášení se provede tlačítkem *Přihlásit*, případné vypnutí je provedeno tlačítkem *Ukončit*. Dále se pod těmito tlačítky nachází políčko, které je po spuštění aplikace neviditelné. Jeho úkolem je informovat uživatele o stavu jeho přihlášení, případně dostupnosti serveru, mimo jiné jsou zde zveřejněny i chybové stavy.

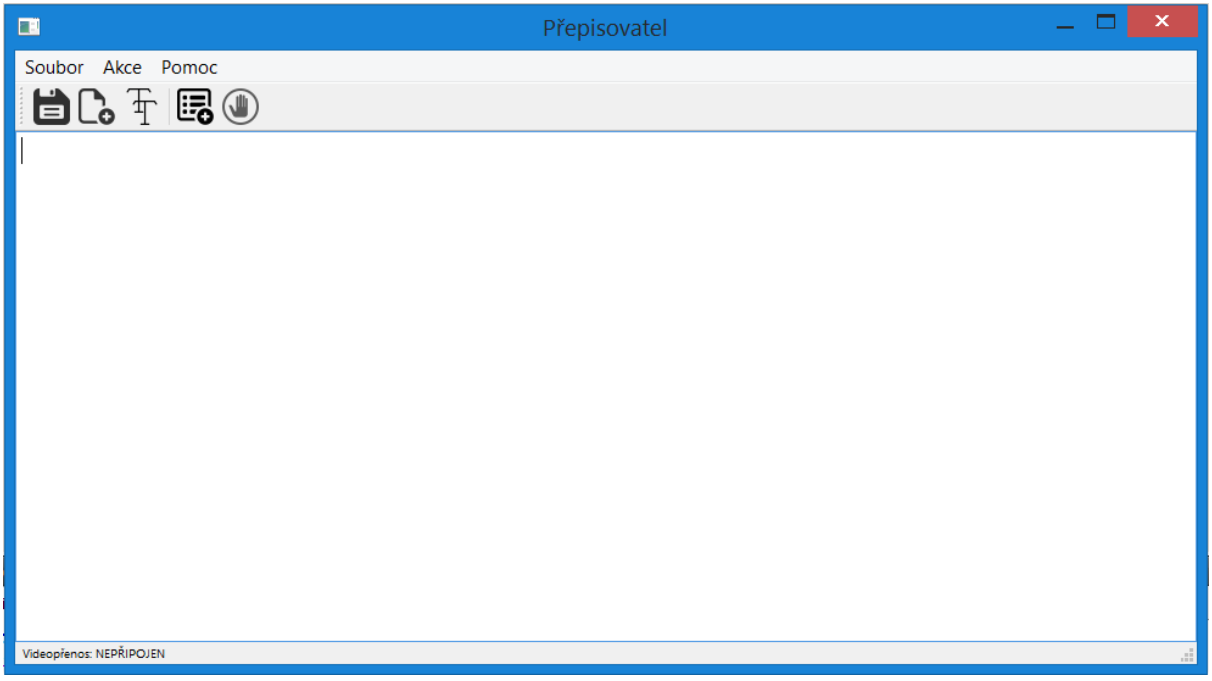
Okno dále nabízí nastavení připojení pod ikonkou ozubeného kolečka. Nastavení je zobrazeno v novém okně, kde je možnost změny síťové adresy, případně doménového jména a portu přihlašovacího serveru. Aplikaci lze použít také v režimu *Textový editor*, který z aplikace udělá klasický textový editor bez přihlašování na server a snímání napsaného textu. Posledním prvkem tohoto okna je tlačítko s ikonkou otazníku pro zobrazení informací o aplikaci.

V případě, že je uživatel úspěšně přihlášen nebo spustí aplikaci v režimu *Textový editor*, přihlašovací okno zmizí a zobrazí se okno samotného editoru, jak lze viděl na obrázku 5.2. Okno se skládá ze tří základních prvků.



Obrázek 5.1: Přihlašovací okno do systému

První z těchto prvků je menu, které se dále dělí do sekcí: *Soubor*, *Akce* a *Pomoc*. V sekci *Soubor* má uživatel na výběr z operací: *Nahrát*, *Uložit*, *Uložit jako* a *Smazat obsah*. Druhá sekce *Akce* nabízí operace *Přidat videopřenos*, *Zastavit videopřenos*, *Nastavení připojení*, *Nastavení písma* a *Odhlásit se*. Akce *Přidat videopřenos* otevře nové okno s dostupnými videopřenosy v seznamu, v němž uživatel může vybrat jeden přenos a aktivovat ho. Při úspěšném aktivování videopřenosu se zpřístupní akce *Zastavit videopřenos*, která je jinak znepřístupněná. Akce *Nastavení písma* otevře nové okno, kde si uživatel může nastavit typ písma, barvu a efekt textu. Operace *Odhlásit se* odpojí přepisovatele od serveru a vymění aktuální okno za původní okno po spuštění aplikace. V sekci *Pomoc* se nachází pouze jedna operace s názvem *O programu*.



Obrázek 5.2: Okno editoru pro snímání napsaného textu

Druhý z prvků v okně je prvek reprezentující nástrojovou lištu. V této liště se nachází vybrané operace z předešlého menu.

Třetí prvek okna je místo, v rámci něhož uživatel píše samotný text. Tento text je snímán a zeleně je obarvována ta část textu, která se již odeslala na server.

Ve spodní části nalezneme lištu, z níž lze vyčíst aktuální stav přenosu. Ten se může nacházet ve dvou stavech: *NEPŘIPOJEN* a *PŘIPOJEN*.

5.1.2 Důležité algoritmy

Při popisu logiky aplikace pro přepisovatele se zaměřím na algoritmus snímání textu, který je na celé aplikaci nejdůležitější a z hlediska implementace nejzajímavější. Dále upřesním princip obarvování odeslaného textu a obecně změnu písma, velikosti a efektu textu.

Snímání textu

Uživatel píše svůj text do objektu `QTextEdit`, který nemá v rámci Qt aplikačního rámce naimplementovanou žádnou metodu pro snímání, nebo vyhodnocování napsaného textu. Z tohoto důvodu jsem musel naimplementovat vlastní algoritmus, který snímá napsaný text.

Algoritmus je zaměřený na klasické psaní na konec souboru, není přizpůsoben k vepisování nebo mazání textu uprostřed již napsaného textu. Aplikace však zaznamenává a reflektuje mazání z konce, k čemuž při tvorbě simultánního přepisu nejspíše bude docházet. Při návrhu a implementaci algoritmu jsem počítal s tím, že přepisovatel bude vytvářet simultánní přepis, a proto bude psát lineárně bez zpětného opravování textu uprostřed souboru, které by již nemělo smysl.

Na zmíněný objekt `QTextEdit` je připojena metoda registrující stisknuté klávesy, které jsou následně vepsány do okna. Metoda se především zaměřuje na klávesy mezerník, tečka, vykřičník a otazník. Na základě stisku těchto kláves je zjištěna aktuální délka souboru, jež je porovnána s předchozím záznamem délky, rozdíl těchto hodnot má za následek zjištění délky textu, který přibyl od minulé kontroly. Následně je z konce souboru vyjmut takový počet znaků, který se rovná rozdílu již zmíněných délek. Tento text je rozeznán jako nově napsaný a je odeslán na server. Problém nastává tehdy, pokud nedojde k stisku snímaných kláves, tedy mezerníku, tečky, vykřičníku nebo otazníku. Potom by text přibýval, ale program by nedostal impuls k vyhodnocení situace. Z tohoto důvodu je v programu nastaven časovač, který pravidelně volá každých 300 milisekund metodu, která zjišťuje poslední čas vyhodnocení napsaného textu. Pokud je zjištěno, že čas poslední kontroly textu přesáhl interval 1000 milisekund, je provedeno vyhodnocení textového obsahu bez zmáčknutí jedné z již uvedených kláves.

Problém tohoto celého algoritmu nastává ve chvíli, kdy se začnou umazávat již odeslaná data. Potom by student mohl dostávat zkomolená slova, která jsou jinak napsána správně, ale v průběhu tvorby došlo k překlepu, který před opravením byl již odeslán. Situace je řešena snímáním, kromě již zmíněných kláves, i klávesy Backspace, jíž se předpokládá umazání vytvořeného textu. Pokud je klávesa zmáčknuta, je nastaven příznak mazání a při dalším zmáčknutí libovolné klávesy je zkontrolována celková délka textu. Pokud je délka od poslední kontroly menší, potom je serveru zaslána zpráva o smazání příslušného počtu znaků. Starý stav délky textu je tedy snížen na aktuální stav a další vyhodnocení vychází z této nové délky. Tím dojde k synchronizaci textu mezi přepisovatelem a serverem.

Obarvování textu

Obecný princip změny parametrů textu v objektu `QTextEdit` je založen na HTML značkách, které jsou do textového řetězce vkládány. Tyto značky nejsou v grafickém rozhraní vidět, ale fyzicky jsou

zaznamenány v řetězci. Pomocí metody `ToHTML` lze získat text s HTML značkami a upravit ho podle představ. Následné vložení textu s novým rozmístěním HTML značek text upraví podle potřeby.

Na tomto principu jsem založil i obarvování odeslaného textu na zelenou barvu. Na začátku textu je vložen HTML značka na obarvení textu na zelenou barvu a za posledními odeslanými daty se nachází ukončovací značka. Tím je všechen text za kurzorem zelený a další tvořený text bude původní předdefinované barvy.

Na základě tohoto principu je implementována i úprava písma, velikosti nebo efektu textu, které lze nastavit přes grafické rozhraní již zmíněnou akcí *Nastavení písma*.

5.1.3 Síťová komunikace

Prostřednictvím síťové komunikace komunikuje aplikace přepisovatele přes server s ostatními aplikacemi. Pro síťovou komunikaci jsem vybral protokol UDP.

Při každém spuštění je načítáno nastavení z předchozí relace. Není tedy potřeba při každém spuštění zadávat síťovou adresu nebo doménové jméno a port. Tyto parametry jsou zabaleny v JSON formátu a následně uloženy do souboru `Settings.txt`. Po nahrání nastavení ze souboru je připojen UDP soket, pod kterým se vede následná komunikace. Aplikace stále udržuje spojení se serverem a kontroluje si, jestli nebylo provedeno odhlášení. Tato kontrola spočívá v pravidelném vybírání schránky. Pokud je ve schránce nalezena zpráva od serveru o neautorizovaném přístupu, je proveden následný pokus o znovu přihlášení.

Zabezpečení komunikace

V případě využití aplikace se přepokládá síťová komunikace v otevřeném internetu. Tento fakt sebou nese povinnost zabezpečit síť proti útokům a případnému odposlechu komunikace. Pro zabezpečení komunikace jsem použil šifrovací algoritmus SHA1. Samotné šifrování je používáno při autorizaci, kdy je nebezpečné přenášet přihlašovací heslo sítí. V praxi to vypadá tak, že server zašle uživateli náhodně vygenerované číslo, které klient na základě svého hesla zašifruje. Vytvořená šifra se následně zašle zpět na server. Není tedy sítí přenášeno samotné heslo, ale pouze číslo, které bylo heslem zašifrováno. Server si vytvoří šifru na základě správného hesla, které má ve své databázi. Po tom, co server příjme vytvořenou šifru od klienta, porovná přijatou šifru se správnou a vyhodnotí, jestli klient použil správné heslo.

Krom samotné autorizace je zapotřebí šifrovat přenášená data kvůli případnému odposlechu. Z důvodu rozdílného programovacího jazyka mezi klientem a serverem jsem byl nucen naimplementovat vlastní šifrovací algoritmus, neboť jsem nebyl schopen najít knihovnu, která by měla naimplementovaný stejně fungující kódovací a dekódovací algoritmus v obou jazycích (PHP, C++). Vytvořil jsem tedy jednoduchý algoritmus, který není náročný na provedení, z důvodu možného zatížení serveru.

Základem pro zašifrování zprávy je původní šifra, která se znovu zašifruje za pomoci hesla. Tato operace je nutná z důvodu zveřejnění původní šifry v síti. Postup šifrování je následující. Vezme se první znak zprávy a je k němu přičten první znak z vytvořené šifry, pokud šifra není natolik dlouhá, aby vystačila na celou zprávu, po jeho dojití na konec je brána znovu od prvního znaku. Následně je odečtena délka do konce zprávy a přičtena hodnota následujícího znaku. Pokud se jedná o poslední znak, není nic přičteno. Při dešifrování je zvolen opačný postup, tedy se začíná posledním znakem a dešifruje se pozpátku. Výsledkem tohoto algoritmu je spleť znaků pokrývající celý rozsah 8 bitů pro uložení znaku.

Komunikace se serverem

Po nastavení a připojení soketu čeká aplikace na zadání uživatelského jména a hesla, pokud se uživatel nerozhodne pro práci pouze jako textový editor. Po zadání těchto informací započne komunikace mezi aplikací přepisovatele a serverem. Komunikaci zahajuje přepisovatel zprávou ve tvaru `T;HE;USERNAME`, kde položka `USERNAME` obsahuje uživatelské jméno přihlašujícího se uživatele. Program očekává zprávu ve tvaru `HELLOU;NUMBER`. Obsahem této zprávy je položka `NUMBER`, která nese náhodně vygenerované číslo pro následné zašifrování. Klient přijaté číslo zašifruje svým heslem a výsledek zašle zprávou ve tvaru `T;RE;USERNAME;HASH`. V případě správných dat zašle server zprávu `AUTHORIZED;ID`, pokud nejsou data správná, je registrace zamítnuta zprávou `NOTAUTHORIZED`. V případě úspěchu je obdrženo unikátní identifikační číslo v položce `ID`, pod kterým se uživatel na serveru vydává.

Dále komunikace pokračuje zasíláním dat serveru, data jsou šifrována již zmíněným algoritmem. Zprávy nesoucí data mají následující formát `T;DA;ID;MESSAGENUMBER;DATA`, kde položka `DATA` reprezentuje zaslaná data. Důležitou položkou je `MESSAGENUMBER`, který udává číslo zprávy. Server díky tomuto číslo pozná, jestli nebyla předchozí zpráva ztracena, případně prohozena. Potvrzení o přijetí vypadá ve tvaru `ACCEPTED;MESSAGENUMBER`. V případě, že server nepotvrdí přijetí zprávy, je následující datová zpráva obohacena o nepotvrzená data. Pokud počet nepotvrzených zpráv přesáhne 30, potom je uživatel varovným oknem obeznámen o ztrátě kontaktu se serverem.

Zpráva ve tvaru `T;DD;ID;MESSAGENUMBER;COUNT` slouží k odstranění již zaslaných dat. Používá se pro smazání znaků, které byly již odeslány. Položka `COUNT` udává počet znaků ke smazání. Zprávy o mazání jsou taktéž číslovány stejnou číselnou řadou jako datové zprávy. V případě nepotvrzení zpráv jsou datové zprávy obohaceny o zprávy mazání a naopak, aby nedošlo k mazání jiných dat.

Pokud nejsou zasílána žádná data, například z důvodu přestávky, je důležité stále udržovat spojení se serverem. Jinak by server odhlásil uživatele ze své databáze. Spojení se udržuje prostřednictvím tzv. oznamovacích zpráv. Tyto zprávy mají za účel pouze obeznámit server o stálé aktivitě. Oznamovací zprávy jsou zasílány ve tvaru `T;SH;ID`.

Další operací mezi klientem a serverem je vyžádání seznamu přístupných videopřenosů od přednášejících. Požadavek o tento seznam je realizován zprávou ve tvaru `T;GE;ID`. Server odpovídá v podobě seznamu, jehož každý záznam má tři položky. Zpráva má tvar `StreamerID;CLASS;TIME;....` Položka `StreamerID` udává unikátní identifikační číslo registrovaného přednášejícího, `CLASS` udává číslo přednáškové místnosti, ve které se zařízení nachází. A poslední položka `TIME` udává čas registrace aplikace přednášejícího. Pokud si uživatel z nabízeného seznamu vybere přednášejícího, kterého následně aktivuje. Pro tuto akci je zaslána na server zpráva ve tvaru `T;AS;ID;StreamerID`, kde `StreamerID` je unikátní identifikační číslo aplikace přednášejícího, který se má aktivovat. Tím však není tento proces zcela ukončen. Server kontaktuje aplikaci vybraného přednášejícího a až je operace potvrzena i aplikací přednášejícího, je zaslána přepisovateli zpráva ve tvaru `StreamerOn`, která indikuje zapnutí videopřenosu. V případě, že navázaný videopřenos zkolabuje nebo je ukončen vyučujícím, obeznámí server tuto skutečnost zprávou `StreamerOff`. Zpráva je obdržena i tehdy, když o ukončení navázaného proudu zažádá sám přepisovatel. Žádost o ukončení videopřenosu je zaslána ve tvaru `T;SS;ID`.

Pro ukončení relace je zaslána zpráva `T;OV;ID`. Toto oznámení je posláno při zvolení akce *Odhlásit se* uživatelem anebo je tato akce volána v destruktoru celé aplikace. Pokud je aplikace korektně ukončena před svým zavřením, obeznámí server o ukončení relace. V případě havárie, nebo náhlého odpojení od sítě, si tuto situaci řeší server sám, a to vypršením časového limitu.

5.2 Implementace aplikace pro vyučujícího

Aplikace vyučujícího se skládá ze tří částí. Uživatel má přístup k řídícímu prvku, který slouží k aktivaci a deaktivaci zbylých dvou částí a zajišťuje samotnou komunikaci se serverem. Tuto řídící část jsem implementoval v jazyce C++ za použití Qt aplikačního rámce. Zbylé dvě části jsou Nginx server pro platformu Windows a Open Broadcaster Software studio dále OBS. Tyto dvě části jsou staticky nastaveny pro funkčnost ve vytvořeném kontextu.

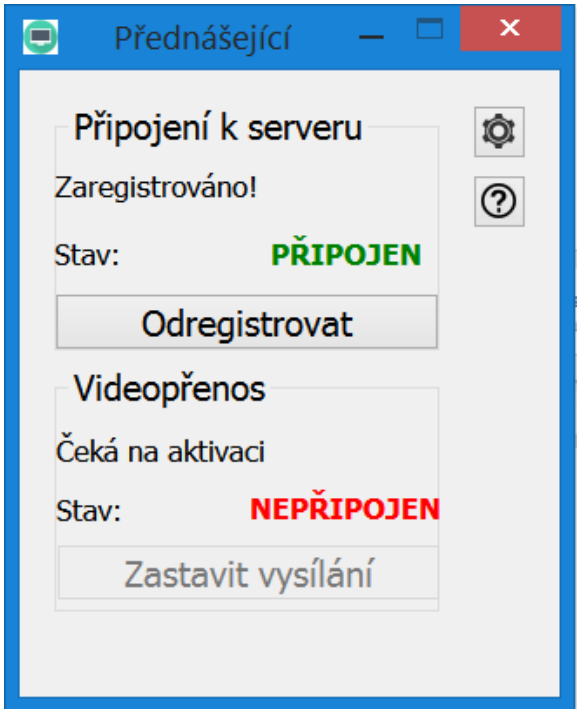
5.2.1 Grafické rozhraní

Grafické rozhraní bylo vytvořeno stejně jako aplikace přepisovatele pomocí nástroje Qt Designer a pro kompilaci bylo využito statické kompilace. Aplikace se skládá z jednoho hlavního okna, které je zobrazeno hned po spuštění, a dvou vedlejších oken. Jedno z oken obsahuje nastavení síťové adresy nebo doménového jména a portu, na kterém se nachází server. Druhé reprezentuje nápovědu a vysvětlení celé aplikace.

Hlavní okno, které lze vidět na obrázku 5.3, se skládá ze dvou oddílů. První oddíl oznamuje stav připojení k serveru a druhý oddíl zpravuje uživatele o stavu snímání obrazu. Oba oddíly se skládají ze dvou měnících se štítků a tlačítka. Jeden ze štítků oznamuje uživateli stav nebo akci, kterou aplikace právě provádí, mimo jiné jsou zde zveřejněny případné chybové stavy. Druhý štítek nabývá pouze dvou hodnot: PŘIPOJEN nebo NEPŘIPOJEN.

Po pravé straně se nacházejí další dvě tlačítka, jejichž význam je popsán obrázky. První s obrázkem ozubeného kolečka slouží k otevření již zmiňovaného nastavení připojení a druhé s ikonkou otazníku k popisu aplikace.

Aplikace je přizpůsobena k bez rušivému běhu, a proto při minimalizaci není odkaz na běžící aplikaci zveřejněn na hlavním panelu, jak je normálně zvykem, ale odkaz je ukryt do skrytých ikon (icon tray). Odtud lze po kliknutí pravým tlačítkem aplikaci ovládat. Jsou k dispozici operace *Ukončit*, *Obnovit*, *Odpojit od serveru* a *Zastavit vysílání*, pokud je vysílání aktivní.



Obrázek 5.3: Okno aplikace přednášejícího

5.2.2 Důležité algoritmy

V rámci popisu logiky aplikace se zaměřím na úpravy, které byly provedeny na aplikaci OBS proto, aby vyhovovala celému kontextu a chovala se podle potřeb. Dále se budu věnovat nastavení serveru Nginx a samotné komunikaci řídícího prvku se serverem Nginx a OBS.

Úprava OBS

Program OBS je využíván k zaznamenávání obrazu a zvuku z různých zdrojů, které jsou připojené k počítači. Nabízí velké spektrum nastavení, jež je dostupné z grafického rozhraní programu. Pro mé účely bylo potřeba odpojit veškeré prvky grafického rozhraní a přizpůsobit tomu samotný program. V rámci odpojování grafického rozhraní jsem vypnul různé náhledy, jejichž cílem bylo informovat uživatele o vzhledu výsledného videopřenosu a vlivu přidávajících se efektů. Tyto náhledy však vytěžovaly procesor a pro mé použití byly zbytečné. Dále bylo zapotřebí odpojit aktualizace a oznamovací upozornění, které zablokují program pro další akci. V případě, že se jednalo o dotaz pro uživatele, došlo k nastavení odpovědi podle potřeby konceptu. Na závěr jsem odpojil samotné okno programu a tím docílil toho, že program po spuštění běží jako program v pozadí bez grafického rozhraní.

Pro přenos programu na jiná zařízení bylo nezbytné přenášet s programem i samotné nastavení. Program má nastaveno ukládat a načítat nastavení ze složky, ke které je nastavena statická adresa. Složka se nachází v adresáři „`C:\Users\User\AppData\Roaming\obs-studio`“. Jedna možnost byla po spuštění řídícího prvku nakopírovat do tohoto adresáře potřebná nastavení. Tím by však mohlo dojít ke kolizi s nainstalovaným OBS studiem, které uživatel používá. Proto jsem v OBS upravil cestu k souborům obsahujících nastavení na cestu vedoucí k adresáři, kde se nachází spouštěcí soubory vytvořené aplikace a upraveného OBS.

Po zajištění cesty k nastavení bylo potřeba vytvořit potřebná nastavení. Hlavním prvkem nastavení byla adresa, kam se videopřenos má vysílat. Tato adresa má hodnotu „`rtmp://127.0.0.1/hls`“. Dále byl nastaven vysílací klíč a výchozí kodek x264 byl změněn na kodek H.264 kvůli menšímu datovému toku a podpoře tohoto formátu u mobilních zařízení.

Původním záměr bylo přijímat text tvořený u přepisovatele a vkládat ho do vytvářeného videopřenosu. Program OBS má v sobě implementované algoritmy, které promítají text do videopřenosu. Text je možno vložit staticky do konfigurace programu anebo nastavit čtení ze souboru. Při pokusech ukládat přijatý text do souboru, aby si ho OBS následně načítal, bylo docíleno „poskakování“ textu na obrazovce. Po delším zkoumání jsem přešel na absenci průběžného čtení textu ze souboru podle potřeby vykreslování do obrazu. Text je na začátku celý načten do vyrovnávací paměti, z kterého se následně čte. Při změně souboru je volána metoda, která původní text ve vyrovnávací paměti vymaže a nahraje novou verzi souboru. Touto operací text na obrazovce poskočí. Pokud je změna provedena po každém přijatém slově, což je přepokládané dělení textu na jednotlivé zprávy, poskočí text na obrazovce dvakrát až třikrát za sekundu. Řešením této situace bylo vyplnit soubor bílými znaky, které se následně zaměňovaly za přijatý text. Bohužel jsem narazil na limit programu, jenž nedokázal přijmout větší soubor než 1100 znaků. Nepodařilo se mi nikde vyhledat nastavení, které by upravovalo tento limit. Výsledkem byl neúspěch této idey a následné přeorganizování celého konceptu do stávající podoby.

Nastavení Nginx

Ačkoliv Nginx server je open source licence, nebylo potřeba upravovat zdrojové soubory. Jediné nastavení bylo prováděno v souboru „`.\nginx\conf\nginx.conf`“, jenž slouží k nastavení serveru pro práci, kterou má obstarávat. Hlavním úkolem Nginx serveru v našem kontextu bylo přijímat

RTMP proud od OBS studia a dále ho distribuovat naslouchajícím studentům. Tuto vlastnost samotné OBS nemá, a proto je zapotřebí využít Nginx jako prostředníka. Kvůli komplikacím, kterým se více budu věnovat v kapitole „5.3.2 Videopřenos a textový proud“, bylo zapotřebí nastavit Nginx ke konvertování protokolu RTMP na HLS. Soubory potřebné k provozování HLS jsou ukládány do složky „`.\nginx\tmp\hls`“. Dále byla nastavena velikost přenášených částí (chuck size) na co nejmenší hodnotu z důvodu získání co nejkratšího zpoždění. Velikost jsem tedy nastavil na 100 kilobytů. Mimo jiné byl změněn standardní port RTMP protokolu 1935 na 7888 z důvodu občasného obsazení a zablokování ochrannými prvky operačního systému. V příloze číslo 1 se nalézá obsah konfiguračního souboru „`.\nginx\conf\nginx.conf`“.

Řídící prvek

Řidicím prvkem je již zmíněný naimplementovaný program, který má za úkol komunikovat se serverem, poskytnout uživateli grafické prostředí a ovládat zbylé dva prvky. Vzhledem k tomu, že OBS a Nginx jsou staticky nastaveny a s jejich konfigurací není třeba nijak hýbat, řídícímu programu stačí čtyři základní operace: zapnout/vypnout OBS a zapnout/vypnout Nginx. Pro tyto operace jsem využil příkazovou řádku OS Windows. Aplikace využívá funkci `WinExec`, která přijímá příkazy příkazového řádku a vykonává je. Tato funkce je volána s parametrem `SW_HIDE`, který pouští okno příkazového řádku autonomně.

Pro spuštění OBS je použit příkaz příkazové řádky „cd `OBS//bin//32bit & obs32.exe`“. Následné zavření OBS je provedeno příkazem příkazové řádky „`taskkill /IM obs32.exe /F`“. K spuštění Nginx slouží příkaz příkazové řádky „`cd nginx & start nginx`“ a k ukončení „`cd nginx & nginx -s stop`“.

5.2.3 Síťová komunikace

Pro komunikaci se serverem byl zvolen protokol UDP stejně jako u aplikace přepisovatele. Tyto dvě aplikace se odkazují na stejnou síťovou adresu a port serveru.

Vzhledem k nespolehlivosti UDP paketů je zapotřebí kontrolovat stálou aktivitu klienta a serveru. Proto klient v pravidelných intervalech, které jsou nastaveny na 15 sekund, zasílá zprávy oznamující aktivitu, na které server odpovídá. Pokud neaktivita serveru přesáhne 120 sekund, je vypnuto snímání obrazu, pokud je aktivní. Nadále se pak aplikace považuje za odhlášenou od serveru.

Ukládání nastavení sítě a následné načítání jsou koncipovány stejně jako u aplikace přepisovatele viz kapitola 5.13.

Komunikace se serverem

Aplikace je po spuštění nastavena na automatické přihlášení na server. Komunikace je zahájena zprávou `L;RE;CLASS;IP`, kde položka `CLASS` udává učebnu, v které se vyučující nachází, a hodnota `IP` je síťová adresa v lokální síti. Server následně odpovídá zprávou `AUTHORIZED;ID` nebo `NOTAUTHORIZED`. Pokud je přihlášení úspěšně schváleno, součástí zprávy je položka `ID`, pod kterou se aplikace dále vydává. Další postup spočívá ve vyčkávání na aktivační zprávu.

Vzhledem k stálému udržování komunikace, je nastaven časovač, který každých 15 sekund zasílá oznamující zprávu o stálé aktivitě. Oznamující zpráva je ve tvaru `L;NO;ID;STATE`. Položka `STATE` udává aktuální stav aplikace, který nabývá hodnot `ACTIVE` nebo `NOTACTIVE`. Do zpráv je přikládán aktuální stav z důvodu synchronizace mezi aplikací a serverem, která může být narušena nespolehlivostí UDP protokolu.

Při zaslání oznamující zprávy aplikace nečeká na odpověď serveru. Vybírání schránky je prováděno v pravidelných intervalech o velikosti 1000 milisekund, kdy jsou zpracovány všechny zprávy ze schránky. Ve schránce se nachází zprávy ze serveru, které mohou nabývat 4 hodnot. První hodnota `OK` vyjadřuje správný stav aplikace. Hodnotou `STOPSTREAM` server vyzývá aplikaci k zastavení videopřenosu, naopak `STARTSTREAM` nařizuje zahájení videopřenosu. Poslední hodnota, kterou může zpráva nabývat, je `NOTAUTHORIZED`, jež vyjadřuje chybějící přihlášení k serveru. Na základě této zprávy se aplikace automaticky pokusí o znovu přihlášení. Při zahájení, případně ukončení videopřenosu, je server obratem obeznámen zprávou ve tvaru `L;AS;ID` při zahájení nebo `L;DS;ID` při ukončení.

5.3 Prostředí pro studenta

Prostředí pro studenta je koncipováno jako webová stránka dostupná pod URL adresou. Nabízená služba není veřejná a je pouze pro studenty se sluchovým postižením, z tohoto důvodu je student při přístupu na URL adresu vyzván k přihlášení. Přihlášení je uskutečněno přenášením přihlašovacích údajů v HTTP hlavičce. Po úspěšném přihlášení se studentovi zobrazí seznam aktivních přepisovatelů a režimy, které jsou k dispozici. Na výběr může být maximálně ze dvou režimů. První z režimů je případ, kdy je nabízen pouze textový proud. Jedná se o příjem pouhého textu, který reprezentuje simultánní přepis. Druhý režim obsahuje textový i videopřenos. Tento režim je aktivní, pokud přepisovatel svoji službu spároval se záznamem obrazovky přednášejícího.

5.3.1 Textový proud

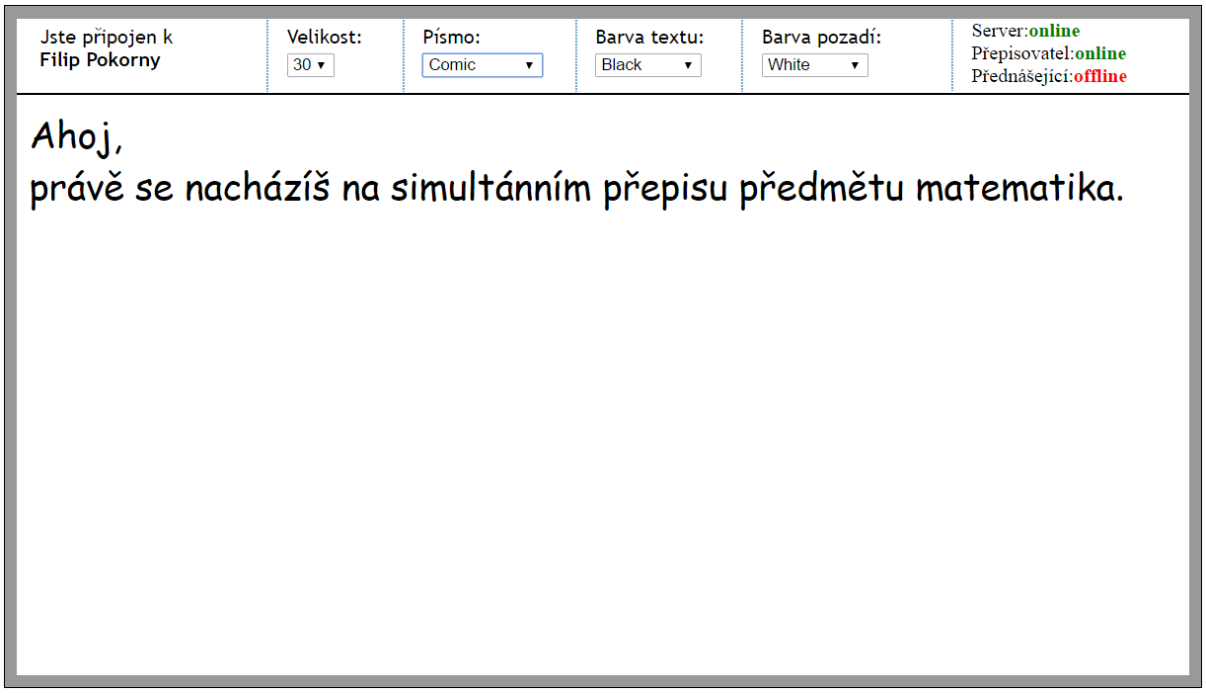
Stránka je jednoduché koncepce, jak lze vidět na obrázku 5.4, kde převážnou část obsahuje panel pro zobrazení přijatého textu. Zbytek stránky vyplňuje ovládací panel, kde si student může upravovat zobrazovaný text podle libosti. Lze upravovat velikost, typ písma a barvu textu. Student má možnost si změnit i pozadí textu pro lepší viditelnost. Všechny tyto úpravy textu jsou prováděny dynamicky pomocí jazyka Javascript. Při změně jedno z parametrů je volána příslušná funkce, která nastaví text pro nově zvolenou hodnotu.

Při implementaci síťové komunikace jsem se rozhodoval mezi dvěma způsoby komunikace. Jednou z nabízených možností byl *Polling*, který pracuje na principu iterativního dotazování serveru na nová data. Alternativou byla technologie *WebSocket*, která je součástí specifikace HTML5. Pro implementaci jsem zvolil *WebSocket* z důvodu lepší odezvy a menší zátěže serveru.

Při načtení stránek je provedena inicializace a navázání spojení se serverem, které se provede vytvořením objektu typu `WebSocket`. Tomuto objektu je zadáván parametr ve tvaru adresy serveru. Součástí adresy je protokol, síťová adresa, port, jméno skriptu a parametr uživatelské jméno přepisovatele, kterým stránka identifikuje přepisovatele, od něhož chce přeposílat data. Velká část adresy je generována na základě dat v MySQL databázi, kterou si načte HTTP server při vytváření stránky pro studenta. Ve výsledku URL adresa může vypadat následovně: „ws://164.132.51.221:9000/server.php?id=xpokor62“. Dále jsou vytvořeny událostní funkce `onmessage` při přijmu nové zprávy, `onerror` při neúspěšném připojení na server, `onopen` při úspěšném připojení k serveru a nakonec `onclose` při uzavření spojení. Uvedené události jsou uživateli znázorňovány v ovládacím panelu pod položkou „`Server`“.

Po ustanovení spojení (handshake) jsou data přenášena již jedním směrem, a to od serveru ke studentovi. Zprávy jsou zabaleny do JSON formátu pro následné jednodušší zpracování. Nejdůležitějším typem zpráv jsou zprávy typu *data* nesoucí tvořený text, který je promítán do okna

studenta. Dále lze přijmout zprávu typu *delete*, jež nese počet znaků, které se mají smazat z konce již přijatého řetězce. Server studentům zasílá i informační zprávy označené typem `SystemMessage`. Informační zprávy mohou nabývat 4 hodnot: `T_OFFLINE`, `T_ONLINE` při odpojení/připojení přepisovatele a `L_OFFLINE`, `L_ONLINE` při odpojení/připojení přednášejícího. Tyto změny jsou uživateli znázorňovány v ovládacím panelu pod položkami *Přepisovatel* a *Přednášející*.



Obrázek 5.4: Webová stránka pro příjem textového proudu

5.3.2 Videopřenos a textový proud

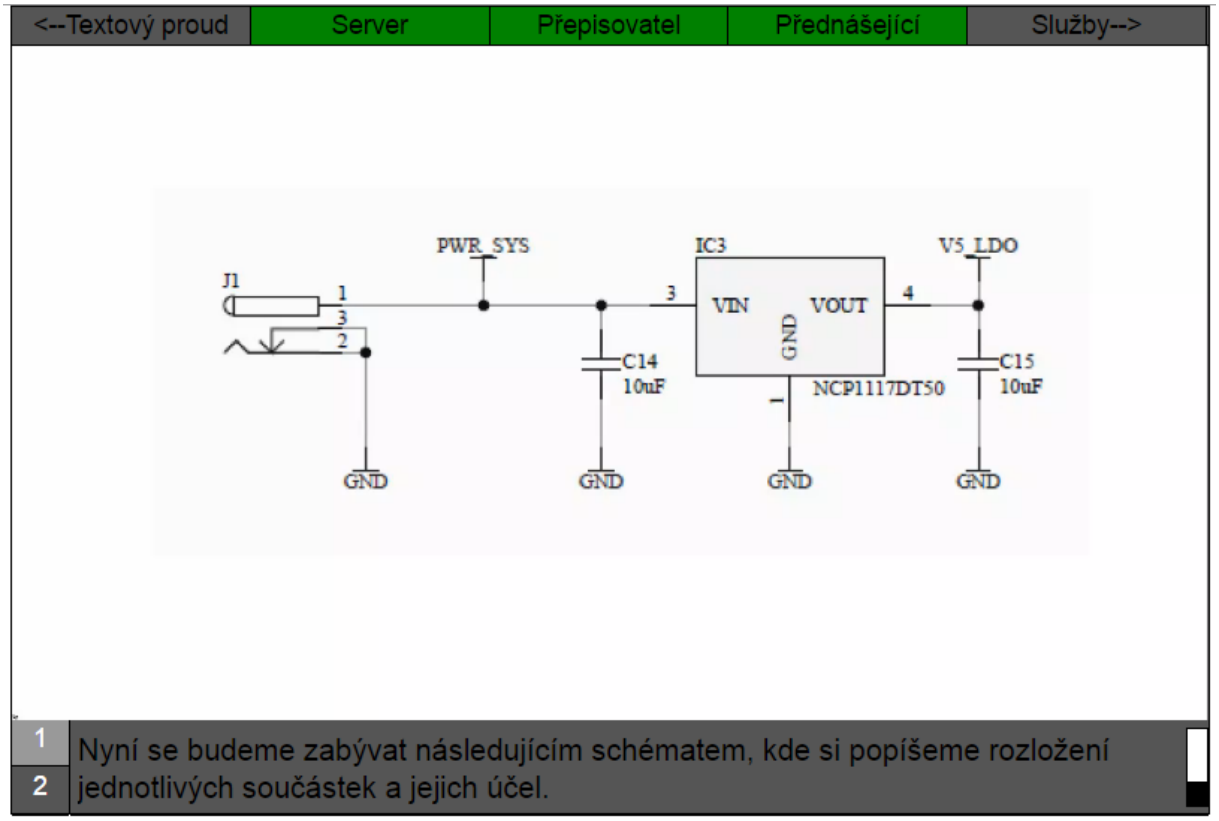
Stránka obsahující textový proud i videopřenos je z hlediska implementace velmi podobná předešlé kapitole zabývající se textovým proudem. Proto se zaměřím pouze na implementační záležitosti ohledně videopřenosu a krátce na koncept stránek.

Pro příjem obrazu jsem zvolil volně dostupný JW Player, jehož nastavení vyplňuje HTTP server při vytváření stránky. Nejdůležitější částí nastavení je vysílací adresa videopřenosu.

Aplikace byla původně navrhnuta pouze pro příjem videopřenosu přenášeným protokolem RTMP. Po následném testování jsem zjistil, že především mobilní zařízení s operačním systémem Android a iOS nemají podporu aplikace Flash 11+, jež podporuje RTMP protokol. Situaci jsem vyřešil zavedením alternativního protokolu HLS, který má však vyšší odezvu. Zatímco při protokolu RTMP se odezva pohybuje kolem 2 sekund, protokol HLS disponuje vyšší odezvou, a to kolem 7 sekund. Na zařízení Android je k dispozici prohlížeč Dolphin, který dodatečně podporuje RTMP protokol, ale u mobilních zařízení s operačním systémem iOS nezbývá než zůstat u protokolu HLS.

Implementace týkající se příjmu a zpracování textového proudu je skoro identické jako u předešlé kapitoly. Z pohledu koncepce stránky postrádají panel pro modifikaci zobrazeného textu a panel s indikací stavu připojení. Místo toho má student na výběr ze dvou modelů zobrazování textu, mezi kterými může průběžně přepínat. První je klasické pole, kde přibývá text a student pomocí posunu jezdce může nahlédnout do historie. V druhém modelu je text zobrazován jako přibývající řádky, které popojíždí ze zdola nahoru a zobrazují minimálně jedno slovo. Pro toto posouvání jsem použil existující Javascriptový doplněk jménem *jQuery Scrollbox*, který vytváří efekt přechodu jednotlivých prvků.

Princip posunu je v pětičlenném seznamu, jenž rotuje své položky. Každá z pěti položek má svoje unikátní ID, na něž je odkazováno. Program si pomatuje, jaká položka má přijít právě na řadu. Jakmile program rozezná nové slovo na základě mezery nebo ukončovacího znaku, nahradí obsah položky, která se má právě objevit, přijatým slovem/slovy a zobrazí položku. Zpracování stránky zobrazující textový proud a videopřenos lze vidět na obrázku 5.5.



Obrázek 5.5: Webová stránka pro příjem videopřenosu a textového proudu

5.4 Implementace serveru

Implementaci serveru lze rozdělit do dvou částí. První částí je klasický HTTP server, který poskytuje rozhraní pro studenta a administrativu celého systému. Pro jeho zpracování byly použity klasické programovací jazyky PHP, HTML, CSS a Javascript. Druhou částí je démon, který aktivně běží v pozadí. Jeho primárním úkolem je naslouchat na UDP a WebSocket schránkách, vyřizovat autorizace a přijímat/přeposílat data. Tento démon celkově řídí celý koncept a jednotlivé prvky se na něj odkazují. Výsledky svých operací a následných změn ukládá do svých proměnných nebo do MySQL databáze, odkud si data přejímají jednotlivé PHP skripty a distribuují je dále studentům nebo správci systému, dále koordinátor.

Aktivního démona jsem se rozhodl naimplementovat jazykem PHP z důvodu jednoduchého zacházení s WebSocket schránkami a MySQL databází. Jeho primárním vstupem je UDP schránka, která je připojená na port 7878 a WebSocket schránka připojená na port 9000.

5.4.1 Důležité algoritmy

Obsluha přepisovatele

Server je implementován tak, aby dokázal obsluhovat více vykonávaných služeb najednou. Předpokládá se připojení libovolného množství přepisovatelů. Proto si server jednotlivé aktivní přepisovatele drží v seznamu přihlášených uživatelů. Před samotným přidáním do seznamu aktivních přepisovatelů se provádí autorizace, která má dvě fáze. V první fázi server příjme přihlašovací jméno, pro které vygeneruje náhodné číslo, jež zašle registrujícímu se přepisovateli. Tento stav je potřeba uložit z důvodu navázání na další fázi autorizace. Druhá fáze pokračuje po přijetí zašifrovaného náhodného čísla. Na základě uložených dat z první fáze se ověřuje identická síťová adresa a port, což je jeden z bezpečnostních prvků. Server ve výsledku disponuje dvěma seznamy, kde v prvním zaznamenává probíhající registrace, a pokud je autorizace úspěšná, záznam z právě probíhaných registrací přesune do druhého seznamu již autorizovaných/přihlášených přepisovatelů.

Vzhledem ke koncepci UDP protokolu není server schopen zjistit, jestli byl uživatel odpojen anebo je stále aktivní. Proto si server ukládá dobu poslední aktivity přepisovatele a pokud doba neaktivity přesáhne nastavených 120 sekund, je záznam o přepisovateli smazán ze seznamu aktivních přepisovatelů. Jednotlivé změny, jako přihlášení a odhlášení přepisovatele, jsou souběžně promítány do MySQL databáze, kde se nastavuje položka `Active` na `true` nebo `false`. Čas poslední aktivity se průběžně aktualizuje zasíláním dat anebo oznamovacích zpráv.

Nedílnou součástí komunikace jsou samotná data, tedy kousky textu reprezentující simultánní přepis. Tyto části textu jsou postupně skládány a zaznamenávány do řetězce s dynamickou velikostí, který je vytvořen v rámci záznamu o aktivním přepisovateli. Po tom, co jsou data zaznamenána, server přepošle obsah příslušným studentům, kteří odebírají textový proud daného přepisovatele. Po ukončení relace přepisovatelem nebo uplynutí mezní doby neaktivity jsou poskládaná data vyhodnocena. Pokud je počet zaznamenaných znaků vyšší jak 500, potom je obsah řetězce uložen do souboru pod jménem „UserName_DateOfStart_TimeOfStart.doc“. Tento název je i s parametry celé relace uložen do MySQL databáze.

Další operací, kterou přepisovatel může provést, je navázání videopřenosu na uskutečňovaný přepis. Přepisovatel vybere jednoho přednášejícího a svoji volbu sdělí serveru. Následně server kontaktuje vyučujícího a až po potvrzení příkazu vyučujícím jsou změny promítnuty do systému. V rámci změn je kontaktován přepisovatel o vyřízení požadavku. Všem studentům připojeným na odebírání dat od přepisovatele je zaslána systémová zpráva o aktivaci videopřenosu. Důležitou částí je však uložení dat do databáze, kde je k záznamu přepisovatele nastavena položka `Stream` na `true` a dále také uložena lokální síťová adresa přednášejícího do položky `IP_Stream`. Při ukončení videopřenosu je položka `Stream` znegována na hodnotu `false`.

Obsluha přednášejícího

K registraci přednášejícího, na rozdíl od přepisovatele, není potřeba přihlašovacích údajů. Proto server drží v paměti pouze jeden seznam, v němž se nachází aktivní přednášející. U jednotlivých položek tohoto seznamu je kontrolována doba poslední aktivity a v případě přesahu 120 sekund je přednášející odhlášen ze strany serveru. Přednášející a server spolu udržují pravidelný kontakt, ve kterém si sdělují stav, a tím se synchronizují. Tímto opatřením je docíleno nežádoucího běhu videopřenosu. Server kontroluje obě strany, jak přepisovatele tak přednášejícího, a v případě ztráty kontaktu jednoho z nich je zastaven videopřenos a o situaci je obeznámeno zařízení studenta. Dále jsou změny uloženy i do MySQL databáze. Server je jediný prvek, u kterého se nepřepokládá havárie, proto spravuje všechna data a všechny připojené klienty.

Obsluha studenta

Obsluha studenta je založena na pravidelné kontrole WebSocket schránky připojené na port s číslem 9000, kde je očekáván požadavek o ustanovení spojení (handshake). Součástí požadavku je uživatelské jméno přepisovatele, od kterého student vyžaduje data. Po ustanovení spojení je zařízení studenta registrováno do seznamu. Položka v seznamu obsahuje číslo soketu, na kterém bylo ustanoveno spojení a uživatelské jméno přepisovatele.

Server při přeposílání dat od konkrétního přepisovatele na zařízení studenta zná přihlašovací jméno přepisovatele a přeposílaná data. Pro dokončení operace potom stačí projít seznam registrovaných studentů a porovnat přihlašovací jména. Pro tuto operaci je provedeno rozdvojení procesů (fork), neboli vytvoření nového procesu, který je kopií hlavního procesu programu. Rodičovský proces pokračuje dál v programu a zpracovává další UDP zprávu, zatímco synovský proces se postará o rozeslání dat, poté je ukončen.

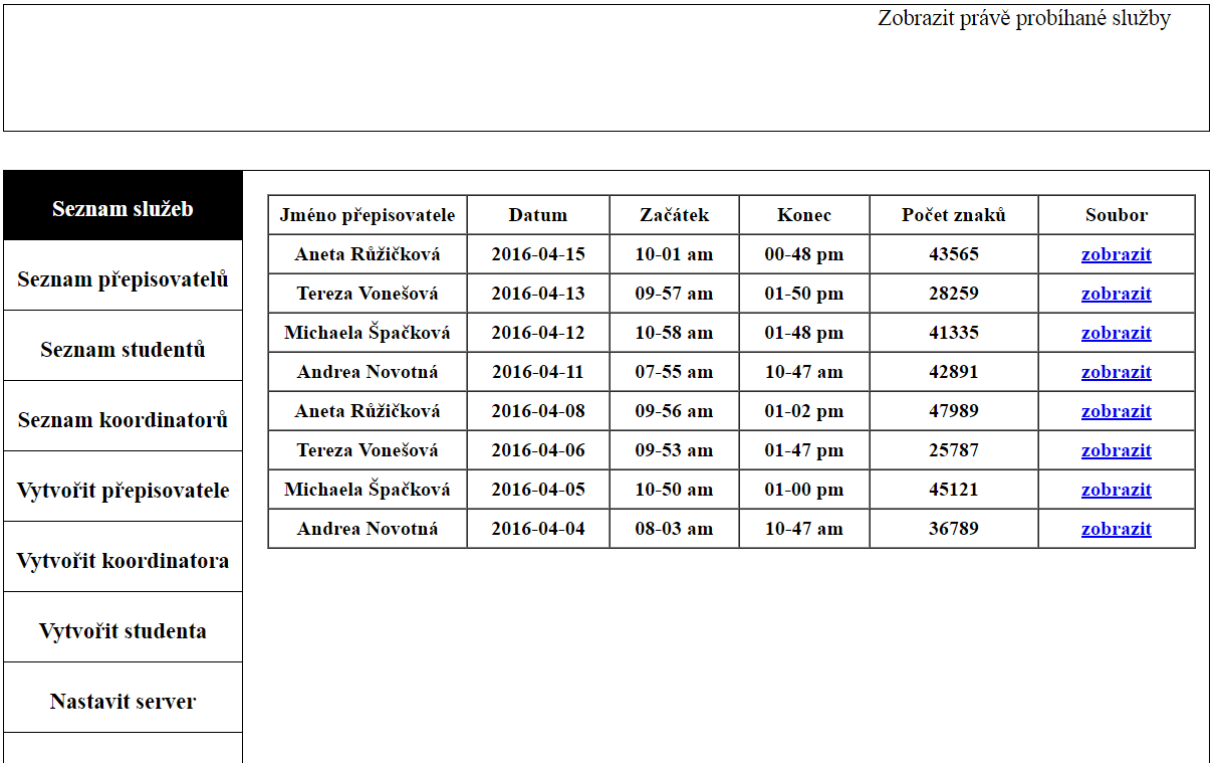
Nezbytnou součástí celého procesu je provádění údržby registrovaných studentů. Kontrola spočívá v projití jednotlivých záznamů, zejména jejich vytvořených schránek. Údržba je opět prováděna rozdvojením (fork) hlavního procesu, z důvodu časové náročnosti při kontrole jednotlivých schránek. Pokud existuje schránka, jejíž klient se odpojil nebo bylo spojení zkrátka ztraceno, potom je číslo schránky zaznamenáno. Takto jsou zaznamenány všechny „odumřelé“ schránky. Mezi rodičovským procesem a synem jsou vytvořeny dvě sdílené paměti. Jedna je o velikosti jednoho bajtu. Tato schránka nabývá pouze hodnoty `true` nebo `false`. Pokud je nalezen minimálně jeden odpojený student, potom je sdílená paměť nastavena synovským procesem na `true`. Rodič v pravidelných intervalech kontroluje hodnotu této sdílené paměti, a pokud má hodnotu `true`, potom jsou v druhé paměti uložena nová data pro převzetí. Do druhé sdílené paměti, o velikosti 200 bajtů, jsou synovským procesem ukládána čísla „odumřelých“ schránek. Rodičovský proces tuto paměť vybírá a na základě získaných dat identifikuje záznamy, uzavře schránky a smaže záznamy ze seznamu.

5.4.2 Databáze

Pro databázi jsem zvolil již zmiňovanou MySQL databázi, která je nainstalovaná na stejném stroji jako běží démon a http stránky. Databáze je nastavena na práci s dotazy pocházejícími pouze z adresy právě používaného stroje, neboli z „loopu“. Návrh databáze lze vidět v příloze číslo 2.

5.4.3 Administrativní rozhraní

Součástí celého konceptu je webová administrace, k jejímuž přístupu jsou potřeba přístupové údaje, které jsou specifikovány v MySQL databázi v tabulce `Coordinator`. Přihlášení je stejné jako pro studenta, akorát při zadání přihlašovacích údajů pro koordinátora je uživatel přesměrován na prostředí administrace. Přihlášený uživatel má k dispozici výpis všech provedených služeb. U jednotlivých služeb je vypsán datum a čas počátku služby, čas ukončení a počet znaků napsaných v relaci. Lze zobrazit text vytvořený v průběhu vykonávání služby, případně text stáhnout v textovém dokumentu. V rámci administrace lze vytvářet, upravovat a mazat účty pro studenty a přepisovatele. Zpracování administrace je přiblíženo obrázkem číslo 5.6.



Obrázek 5.6: Webová administrace

6 Testování

6.1 Zátěžový test

Před samotným uvedením konceptu do výuky jsem se rozhodl provést pár zátěžových testů, a zjistit tak chování aplikace v nestandardních podmínkách. Bylo provedeno celkem pět testů, které se zaměřovaly na potenciální krizová místa konceptu.

První simulace se zaměřovala na velký počet klientů ze strany studenta. V budoucnu by se koncept mohl použít při různých konferencích nebo školeních pro sluchově hendikepované lidi. Při těchto událostech se očekává mnoho klientů, a proto jsem se tuto situaci snažil nasimulovat. Prostřednictví nových záložek a zařízení, která jsem měl k dispozici, jsem vytvořil více jak 30 klientů a následně zkoušel realizovat simultánní přepis. Výsledkem byl bezproblémový chod, bez jakéhokoliv zpoždění, případně výpadku. V rámci tohoto testu bylo zasíláno poměrně velké množství dat, které jsem následně umazával a snažil se vytvořit zatížení.

Druhý test měl za úkol vytvořit nepříznivé podmínky v podobě zahlcené sítě, konkrétně nejbližšího síťového uzlu. V testovaném případě to byl router, který jsem se snažil vytížit po stránce internetového připojení. Dále jsem se snažil vytížit router posíláním dat mezi zařízeními v lokální síti. Výsledkem bylo mírné zvýšení ztrátovosti jednotlivých paketů a datagramů, ale díky implementovanému potvrzování dat, nebyla žádná ztracena. Test měl za následek nárůst zpoždění, které bylo i tak snesitelné.

Principem dalšího testu bylo nasimulovat velké množství ztracených datagramů při cestě mezi přepisovatelem a serverem. Vytvořit tuto situaci standardní cestou je náročné, proto jsem uměle zahazoval určitý poměr doručených datagramů na serveru. Postupným navyšováním o 25 %, 50 %, 75 % a 90 % jsem sledoval chování konceptu. Se zvyšujícím procentem ztráty se navyšovalo zpoždění mezi napsaným textem u přepisovatele a následným zobrazením u studenta. Avšak i přes velkou ztrátovost nebyl problém doručit data v pořádku a ve správném pořadí. Z tohoto testu vyšlo zjištění potřeby relativně dobrého síťového připojení, minimálně takového, které zajistí snesitelnou míru ztrácení. Tuto míru bych stanovil do 10 %.

Následující test byl zaměřen na přenos velkého množství dat. Za pomoci kopírování do editoru jsem se snažil v co nejkratší době zaslat co nejvíce znaků. Následně jsem sledoval chování editoru přepisovatele, kde se dalo očekávat postupné „zasekání“ vzhledem k nárůstu zpracovávaných dat v okně editoru. Po zaslání více jak 200 000 znaků bylo psaní v editoru plynulé, bez jakéhokoliv zasekávání. Po ukončení relace jsem porovnal počet znaků uložených na serveru a počet znaků napsaných v editoru. Výsledkem nebyla ztráta ani jediného znaku.

Posledním testem bylo vyzkoušení konceptu na Fakultě informačních technologií. První test byl proveden na síti *Vutbr,* která bez problémů zpracovávala textový proud s minimální odezvou. Problém nastal při přenosu videa, kde vlivem nastaveného limitu, který byl v rámci účtu rychle vyčerpán, byla rychlost rapidně snížena a videopřenos nebylo možné nadále využívat. Po připojení na síť *Eduroam* byl tento problém vyřešen.

6.2 Testování v reálném provozu

V době vypracovávání bakalářské práce studovali na Vysokém učení technickém v Brně dva studenti využívající simultánní přepis Poradenského centra Alfons. Jeden ze studentů studoval na Fakultě

elektrotechniky a komunikačních technologií (FEKT) a druhý na Fakultě strojního inženýrství (FSI). Při zhotovování poslední části bakalářské práce jsem studenty navštívil v jejich hodinách a pokusil se uvést navržený a naimplementovaný koncept v praxi.

První zkouška byla provedena v jedné z učeben na FEKT. Cílem bylo vyzkoušet základní režim pouhého přenosu textového proudu, a tak jsem pro co největší jednoduchost zařízení připojil na síť *Vutbr*. Síť *Vutbr* je jednoduchá na připojení a na základě informacích od spolužáků z různých fakult bych vyvodil i převážné využití této sítě ze strany studentů. Tato síť je vhodná pouze pro přenos samotného textového proudu z důvodu limitu přenesených dat. Výsledek nebyl příliš dobrý, koncept fungoval, ale v rámci sítě byla velmi intenzivně ztrácena odesílaná data. Ztrátovostí datagramů a paketů docházelo k nepřijatelnému zpoždění až k výpadkům připojených WebSocket schránek. Jednou z příčin mohl být fakt, že učebny FEKTu bývají často zanořeny pod úroveň země, a tedy se nejspíše vytváří špatné podmínky pro přenos Wi-Fi vln. Po konzultaci se studenty místní fakulty jsem byl utvrzen o špatném internetovém připojení v místních učebnách v rámci sítě *Vutbr*. Situaci nenahrává ani fakt, že síť v průběhu přednášek bývá místními studenty přetížena. V průběhu přednášky bývá mnohdy problém s načítáním webových stránek, natož provozováním dalších síťových služeb. Závěrem prvního testování bylo, že přes místní nabízenou síť *Vutbr* není možné službu spolehlivě provozovat.

Dalším testováním jsem se snažil vyhnout síti *Vutbr*, čímž mi vznikl problém s připojením k internetu. Problém jsem vyřešil využitím privátního připojení, které často bývá výhradně pro učitele. Tyto přípojky se nacházejí poblíž katedry nebo přímo v katedře. Po připojení vlastního routeru do této sítě bylo provozování konceptu nadále bezproblémové. Zlepšením přístupu k internetu koncept fungoval, jak se očekávalo. Po úspěšném provozování pouze textového proudu, byl vyzkoušen i videopřenos, který taktéž po celou přednášku fungoval bez problémů. Zde je však možné narazit na zádrhel, kdy počítač vyučujícího a zařízení studenta není ve stejné lokální síti. Vysoké učení technické vlastní síťovou adresu skupiny B, která disponuje velkým množstvím adres. Tyto adresy jsou mimo jiné poskytovány pro většinu školních počítačů, výjimkou nejsou ani počítače v přednáškových a cvičících místnostech. Tím pádem by na VUT neměl být problém s provozováním video proudu.

Dále jsem zkoušel provozovat koncept prostřednictví sítě *Eduroam*, která oproti síti *Vutbr* poskytla taktéž dostačující připojení pro video a textový proud. Problém by mohl nastat v případě využití sítě větším množstvím studentů, kteří naplno vytíží kapacity sítě. Potom by mohly nastat již zmíněné problémy jako při připojení na síť *Vutbr*.

Problémy nastaly na FSI, kde vzhledem k staršímu zařízení přednáškových místností není síťová infrastruktura příliš optimalizovaná a přípojky dostupné z katedry poskytují internet, který je vázaný na síť *Vutbr*. Zde je jediným východiskem síť *Eduroam*.

Dalším řešením by mohlo být zřízením lokální Ad-hoc sítě prostřednictvím routeru, na který by se připojila všechna zařízení, tedy zařízení přepisovatele, přednášejícího a studentů. Potom by potřebný server byl spuštěn na zařízení přepisovatele a ostatní by se na tento server odkazovali.

6.3 Vyhodnocení testování

V případech, kdy se povedlo zajistit dobré internetové připojení, si studenti aplikaci pochvalovali a byli s ní spokojeni. Ocenili jednoduchou obsluhu a přizpůsobení prostředí podle sebe. Byl taktéž oceněn komfort, který aplikace poskytla přepisovateli a studentovi, kteří již nemuseli sedět co nejblíže k sobě, kvůli sdílení obrazovky. Student taktéž mohl zaujmout jiné místo, z kterého měl lepší výhled na tabuli, případně prezentaci, a naopak přepisovatel zaujal místo, kde mohl lépe slyšet vyučujícího. Po ukončení služby byl napsaný text ihned dostupný z administrativního prostředí pro koordinátora, což naopak ocenilo Poradenské centrum Alfons.

V rámci testování jsem narazil na pár drobných chyb, které jsem posléze opravil. Při komunikaci s přepisovateli a studenty jsme dospěli k pár vylepšením, která by do budoucna mohla koncept dále zkvalitnit a udělat více uživatelsky přívětivý. Jedním z návrhů byl zveřejněný seznam připojených studentů v aplikaci přepisovatele, dále vzájemné zasílání zpráv mezi aplikací přepisovatele a studentů. Studenti měli i pár nápadů, jak by bylo možné dále upravit uživatelské rozhraní pro ještě větší pohodlí. V budoucnu by bylo při rozvoji aplikace vhodné spolupracovat se sluchově hendikepovanými studenty a naslouchat jejich nápadům a připomínkám.

7 Závěr

Cílem této bakalářské práce bylo zanalyzovat problém Poradenského centra Alfons, navrhnout nové řešení a následně řešení naimplementovat. Řešený problém spočíval v absenci použitelné aplikace v podmínkách VUT, která by napomáhala v realizaci služby zvané simultánní přepis pro sluchově hendikepované studenty v době přednášek a cvičení. Výsledný produkt byl vyzkoušen v reálném provozu. Bakalářská práce byla po celou dobu tvorby konzultována s vedoucím práce a poradenským centrem Alfons.

Implementace celého konceptu aplikací byla realizována v jazycích C++, PHP, JavaScript, HTML a CSS. V částech implementovaných v jazyce C++ bylo využito aplikačního rámce Qt. Kromě implementovaných částí se některé aplikace skládají z programů pod licencí Open source, které v případě potřeby byly upraveny. Tyto programy jsou Open Broadcaster Software Studio a Nginx server s RTMP modulem. Dalšími potřebnými programy pro správný chod konceptu jsou přehrávač JW Player a databáze MySQL.

Výsledkem úspěšné implementace je tedy koncept aplikací skládající se ze čtyř částí. Jedna z částí je aplikace pro přepisovatele, která je implementována pro platformu Windows a slouží k vytváření simultánního přepisu. Druhá část je aplikace pro přednášejícího, která je taktéž pro Windows a poskytuje záznam obrazovky přednášejícího v podobě videopřenosu. Třetí částí je webová aplikace pro studenta, která je poskytována webovým serverem. Tato aplikace prezentuje přijímaný textový proud a videopřenos v co nejlepší formě studentovi. Poslední čtvrtá část je server, který zajišťuje obsluhu předešlým třem částem. Části pro přepisovatele a vyučujícího byly testovány na operačním systému Windows verze 7,8,10 (32 i 64 bitové verze).

Jedněmi z hlavních priorit navrženého řešení byly jednoduchost a flexibilnost. Z tohoto důvodu je řídícím prvkem server, na který se jednotliví uživatelé odkazují se svými aplikacemi. Server řídí celý chod a zaznamenává potřebná data, která by se jinak musela ručně ukládat na sdílené úložiště. Další myšlenkou tohoto návrhu byla možnost vykonávat simultánní přepis bez fyzické účasti přepisovatele na přednášce. Prostřednictví telefonu nebo jiného zařízení by se přenášel zvukový záznam z přednášky, na základě kterého by přepisovatel psal text. Tyto myšlenky vedly k návrhu, který ke své funkčnosti potřebuje relativně stabilní internetové připojení. Jak se ukázalo, vlivem velkého množství připojených studentů nemusí být samozřejmostí.

Proto jako rozšíření této bakalářské práce navrhuji vytvořit modul, na základě kterého by bylo možné realizovat simultánní přepis bez internetového připojení, tedy pouze na lokální síti. V rámci bezpečnosti se mi podařila naimplementovat veškerá autorizace. Komunikace mezi přepisovatelem a serverem je navíc chráněna proti odposlechu. Dalším rozšířením by mohla být analýza bezpečnostních nedostatků a následné zavedení patřičných opatření.

Na základě závěrečného testování v reálném provozu, které bylo za jistých podmínek úspěšné, bych vyhodnotil vypracovanou bakalářskou práci za úspěšnou. V této myšlence mě utvrzuje i fakt, že studenti s poradenským centrem Alfons vytvořený řešení hodnotili pozitivně. Do budoucna doufám, že bude opuštěno od stávajících nepohodlných praktik a koncept bude plně využíván k službě simultánní přepis.

Literatura

[1] SOSINSKY, Barrie A. *Mistrovství - počítačové sítě: [vše, co potřebujete vědět o správě sítí]*. Vyd. 1. Brno: Computer Press, 2010. ISBN 978-80-251-3363-7.

[2] About HTML5 WebSocket. In: *Websocket.org* [online]. Kaazing, 2016 [cit. 2016-03-29]. Dostupné z: http://www.websocket.org/aboutwebsocket.html

[3] OZER, Jan. What Is a Streaming Media Protocol? In: *Streaming Media: Online video news, trends, and analysis* [online]. Streaming Media, 2012 [cit. 2016-03-29]. Dostupné z: http://www.streamingmedia.com/Articles/Editorial/What-Is-../What-Is-a-treaming-Media-Protocol-84496.aspx

[4] CALETKA, Ondřej. Zranitelný FFmpeg: místo přehrávání videa odesílá soubory. In: *Root.cz: informace nejen ze světa Linuxu* [online]. Internet Info, 2016 [cit. 2016-03-29]. Dostupné z: http://www.root.cz/clanky/zranitelny-ffmpeg-misto-prehravani-videa-odesila-soubory/

[5] *Open Broadcaster Software* [online]. 2016 [cit. 2016-03-29]. Dostupné z: https://obsproject.com/index

[6] Directives - nginx-rtmp-module Wiki. *GitHub: How people build software?* [online]. San Francisco: GitHub, Inc., 2014 [cit. 2016-03-29]. Dostupné z: https://github.com/arut/nginx-rtmp-module/wiki/Directives

[7] What is JW Player? In: *Publisher Support: JW Player* [online]. Longtail Ad Solutions, Inc., 2015 [cit. 2016-03-29]. Dostupné z: https://support.jwplayer.com/customer/portal/articles/1403727-what-is-jw-player-

[8] About Qt. *Qt Wiki* [online]. 2015 [cit. 2016-03-29]. Dostupné z: https://wiki.qt.io/About_Qt

[9] MySQL. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2016 [cit. 2016-03-29]. Dostupné z: https://cs.wikipedia.org/wiki/MySQL

[10] Handicap poradna. *Přes bloky: Poradenské centrum pro studenty VUT* [online]. Brno, 2016 [cit. 2016-04-22]. Dostupné z: http://presbloky.cz/cs/clanek/handicap-poradna

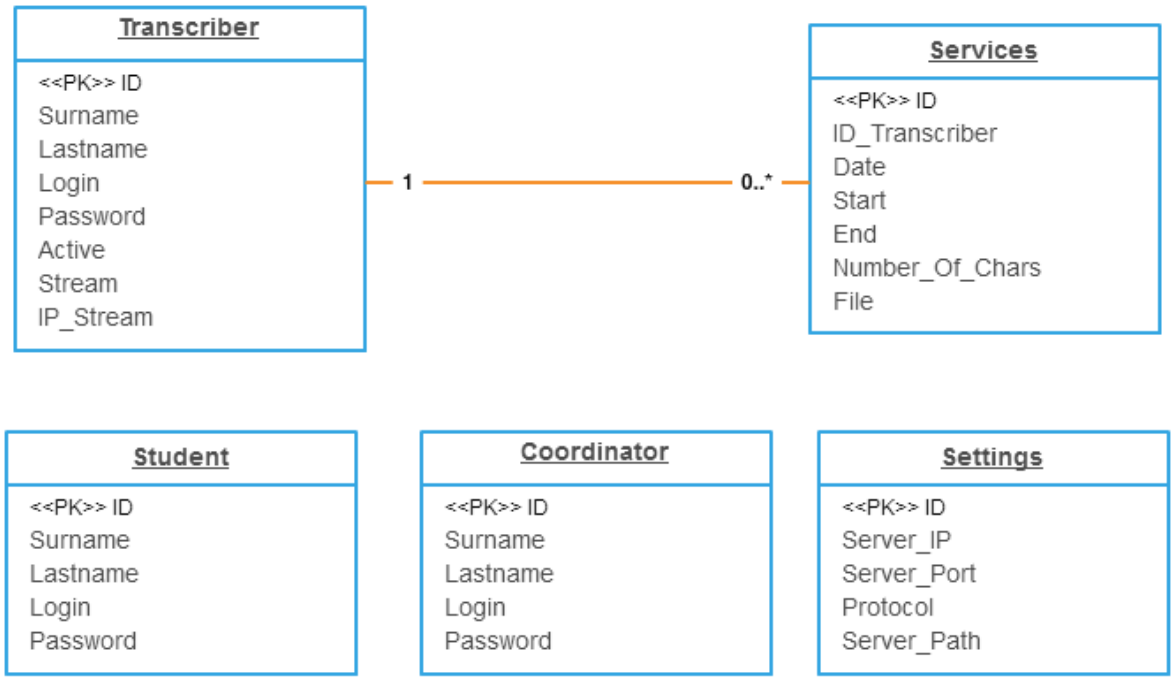
Seznam příloh

Příloha č. 1 - Konfigurační soubor Nginx

```
#user  nobody;
worker_processes  1;
events {
    worker_connections  1024;
}
http {
    server {
        listen      8080;
        location /hls {
            # Serve HLS fragments
            types {
                application/vnd.apple.mpegurl m3u8;
                video/mp2t ts;
            }
            root ./tmp;
            add_header Cache-Control no-cache;
        }
    }
}
rtmp {
    server {
        listen 7888;
        chunk_size 100;
        application hls {
            live on;
            hls on;
            hls_path ./tmp/hls;
            hls_fragment 1s;
        }
    }
}
# ffmpeg -loglevel verbose -re -i movie.avi  -vcodec libx264
#    -vprofile baseline -acodec libmp3lame -ar 44100 -ac 1
#    -f flv rtmp://localhost:7888/hls/stream
```

Následující příloha zobrazuje nastavení Nginx serveru, který je umístěno v souboru „`.\nginx\conf\nginx.conf`“. Prvních pět řádků reprezentuje základní nastavení pro spuštění Nginx serveru. Následuje stanovení HTTP serveru, který slouží k obsluze HLS proudu, a určení jeho portu na standardní hodnotu 8080. V rámci tohoto nastavení je stanoven adresář pro uložení jednotlivých částí proudu (chunky) a jejich typu. Dále je zde stanoven soubor nesoucí seznam jednotlivých částí (playlist), viz kapitola 2.15. Následuje stanovení samotného RTMP serveru, který je nastaven na port 7888. V rámci RTMP je nastavena velikost jednotlivých zasílaných částí proudu na 100 kilobajtů. Další část stanovuje protokol HLS a jeho nastavení. V rámci tohoto nastavení je stanovena cesta pro ukládání jednotlivých částí proudu a jejich časová velikost na jednu sekundu. Poslední tři řádky reprezentují nastavení převodu protokolu RTMP na HLS.

Příloha č. 2 - Schéma MySQL databáze



Příloha č. 3 - Obsah DVD

/BP – Bakalářská práce ve formátu PDF
/BIN – Přeložené binární soubory pro operační systém Windows 32 bitové verze
/DOC – Programová dokumentace (Doxygen)
/SRC – Zdrojové soubory
readme.txt – Stručný návod na zprovoznění aplikace