UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

Systém centralizovaného ukládání dat

Jan Šimůnek

Diplomová práce
2013

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2012/2013**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Jan Šimůnek**
Osobní číslo:	**I11416**
Studijní program:	**N2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Systém centralizovaného ukládání dat**
Zadávající katedra:	**Katedra softwarových technologií**

Zásady pro vypracování:

Cílem této práce je vytvoření aplikace pro centralizované ukládání dat s úložištěm na principu cloudu. Jedná se o aplikaci typu client-server. Serverová část bude spuštěna na serveru, kde bude přístupné úložiště a klientská část na jednotlivých počítačích případně noteboocích.

Aplikace bude umožňovat následující funkcionality:
1. Registraci a správu uživatelů.
2. Synchronizaci a datové přenosy mezi klientem a serverem.
3. Správu vybraných složek k synchronizaci v určitých intervalech nebo při zjištění změny ve složce.
4. Zálohování vybraných složek s možností verzování.
5. Správu vybraných složek sdílených mezi uživateli.

V úvodní části je nutné provést analýzu podobných řešení a porovnání s nově navrhovaným systémem, který bude tvořit předmět této práce. Úvodní část musí obsahovat analýzu navrhovaného řešení, která bude obsahovat popis použitých technologií, návrh databáze a aplikačního řešení. Pro vytvoření aplikace bude využit skriptovací jazyk C ♯ a databáze Oracle nebo MySQL.

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. LACKO, Luboslav. Oracle. Správa, programování a použití databázového systému. Brno: Computer Press, 2007.
2. GROFF, James R. a WEINBERG, Paul N. SQL kompletní průvodce. Brno: Computer Press, 2005.
3. BRYLA, Bob; LONEY, Kevin. Mistrovství v Oracle Database 11g. Jiří Huf. Vydání první. Brno: Computer Press, a.s., 2009.
4. LONEY, K., THERIAULT, M. Mistrovství v Oracle. Praha, Computer Press, 2002.
5. PIRKL, Josef. Řešené příklady v C ♯, aneb, C ♯ skutečně prakticky. 1. vyd. České Budějovice: Kopp, 2005, 300 s. ISBN 80-7232-265-6.
6. ESTRADA, Jorge A. a Tomáš HOLEČEK. C Programujeme profesionálně: pro zkušenější programátory. Brno: Computer Press, a.s., 2003. ISBN 8025100855.
7. KENT, Jeff. Microsoft visual C ♯ 2005: bez předchozích znalostí : průvodce pro samouky. Vyd. 1. Překlad Petr Dokoupil. Brno: Computer Press, 2007, 310 s. ISBN 978-80-251-1584-8.

Vedoucí diplomové práce: **Ing. Miloslav Macháček, Ph.D.**
Katedra informačních technologií

Datum zadání diplomové práce: **31. října 2012**
Termín odevzdání diplomové práce: **17. května 2013**

Univerzita Pardubice Česká republika

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

prof. Ing. Antonín Kavička, Ph.D.
vedoucí katedry

V Pardubicích dne 15. listopadu 2012

Prohlášení autora

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 17. 05. 2013 Jan Šimůnek

Poděkování

Děkuji Ing. Miloslavu Macháčkovi, Ph.D. za vedení celé mé práce. Dále děkuji rodičům, spolužákům a kamarádům za podporu během studia.

Anotace

Práce se zabývá synchronizací mezi více počítači a zálohováním dat. Data jsou uložena v centralizovaném úložišti dat. Obsahuje analýzu konkurenčních řešení. V práci je podrobně popsána vytvořená aplikace.

Klíčová slova

C#, WCF, MySQL, synchronizace dat, zálohování dat.

Title

Centralized data storage system.

Annotation

The thesis deals with the synchronization between multiple computers and data backup. Data are stored in a centralized data repository. The thesis contains an analysis of competing solutions. The thesis described created application in detail.

Keywords

C#, WCF, MySQL, data synchronization, data backup.

Obsah

Seznam zkratek

Db4o	Database for objects
DBMS	Database managment system
HTTP	Hypertext Transfer Protocol
MVC	Model-View-Controller
SOA	Service-Oriented Architecture
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SSL	Secure Sockets Layer
XML	Extensible Markup Language

Seznam obrázků

Seznam tabulek

Úvod

Tato práce se zabývá poměrně rozsáhlým tématem o synchronizaci a zálohování dat. Cílem práce je navrhnout a implementovat funkční aplikaci, která bude splňovat zadané požadavky synchronizace a zálohování. Služba bude řešit problematiku synchronizace dat mezi více počítači v kombinaci s centrálním serverem.

První část této práce obsahuje základní informace o dané problematice. Dále informuje o použitých technologiích a nabízí podrobnou analýzu konkurenčních řešení, která nabízí podobné funkce, i když velmi často s jinými vlastnostmi, které jsou běžnému uživateli skryty v pozadí aplikace. Samozřejmě bylo provedeno porovnání konkurenčních řešení s vyvíjenou aplikací. Dále byla navržena možná řešení některých nedostatků.

Druhá část poměrně podrobně popisuje nejprve jednotlivé funkce, které jsou v aplikaci implementovány. Jedná se o funkce, které jsou nutné ke správnému běhu celého programu a o většině z nich uživatel vůbec neví, protože běží na pozadí aplikace. Později jsou zobrazeny funkce, které může uživatel využívat. Jde hlavně o různá nastavení synchronizace a zálohování. Aplikace také poskytuje nástroje pro základní správu uživatelů.

Použití této aplikace je cíleno do firemní sféry. Ideálně do menšího podniku, například s 10 až 50 zaměstnanci. Tento počet není ovšem nijak závazný. Aplikace samozřejmě zvládne i mnohem více uživatelů. Pokud by však uživatelů mělo být velké množství, bylo by mnohem vhodnější využít jako úložiště cloud. Tato aplikace není k využití v cloudu určena. Pokud by tak mělo být, bylo by nutno upravit některé části kódu.

Použití této aplikace by měli firmy upřednostnit před konkurencí hned z několika důvodů. Aplikace je totiž zaměřena na cenu, rychlost a bezpečnost. To je kombinace, kterou na trhu téměř nelze nalézt. Tyto výhody jsou dány tím, že by aplikace měla být umístěna uvnitř společnosti přímo na serveru, který již většina firem vlastní. Proto největší část ceny na využití tohoto programu zbývá pouze na diskové úložiště. Jedinou nevýhodu nalezneme v tom, že ke správné funkčnosti bude využíván správce sítě firmy. Rychlost programu bude poté závislá pouze na technologických prostředcích síťové infrastruktury v podniku. A největší zabezpečení může být dáno tím, že data nikdy neopustí firmu. Není totiž nutná komunikace přes internet. Podrobnější informace lze nalézt v následující kapitole v analýze konkurenčních řešení.

1 Teoretická část

V této části budou určeny požadavky na aplikaci a budou představeny použité technologie. Dále bude provedena rozsáhlá analýza konkurenčních řešení, provedeno porovnání s vlastní aplikací a navržení řešení případných nedostatků.

1.1 Požadavky na aplikaci

Před zahájením tvorby aplikace byl vytvořen seznam požadavků. Požadavky jsou tvořeny hlavně funkcemi klientské aplikace, ale samozřejmě k plné funkčnosti se velmi často využívá komunikace se službou na serveru.

Požadavky funkční:

- Bude umožněno vybrat libovolné složky k synchronizaci.
- Bude zajištěna automatická synchronizace neustálým sledováním složky nebo v zadaných intervalech.
- Systém bude nahrávat soubory na server, ale také se postará o jejich stažení.
- Systém bude umožnovat registraci a správu uživatelů.
- Bude možno vytvořit zálohu libovolné složky nebo souboru.
- Bude umožněno pravidelně zálohovat data.
- Bude umožněno zálohy procházet a stahovat.
- Bude možné sdílet synchronizované složky mezi různými uživateli.

Požadavky nefunkční:

- Bude vytvořeno rozhraní, ke kterému se budou moci připojit různé platformy.
- Bude vytvořena klientská aplikace.
- Systém nebude zbytečně přenášet data.
- Budou využity moderní technologie.

1.2 Použité technologie

Při vývoji bylo použito mnoho rozličných technologií. Obě aplikace jsou naprogramovány v programovacím jazyku C#. Pro komunikaci mezi aplikacemi byla využita technologie WCF. Data byla ukládána do databází Db4o a MySQL.

1.2.1 .NET

Architektura .NET byla vyvinuta firmou Microsoft a vypuštěna do světa v roce 2002. Cílem nové platformy bylo zjednodušit a sjednotit vývoj aplikací. Technologie .NET se stala hlavním konkurentem platformě Java od firmy Sun Microsystems. Microsoft .NET znamená novou generaci systému vývoje aplikací pro operační systémy Windows. Programovací jazyky pro vývoj .NET aplikací jsou: C#, Visual Basic .NET, F#, J#, IronPython, a další.

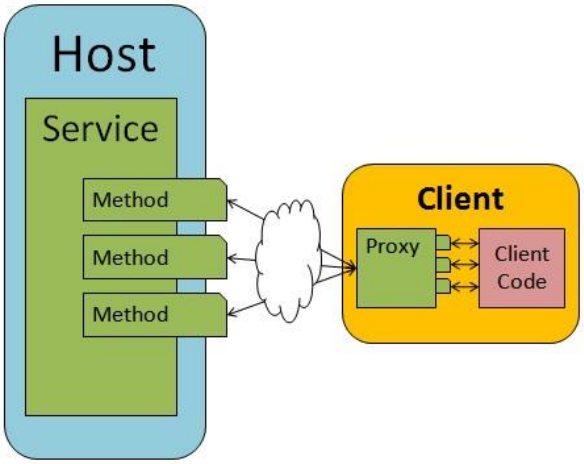
Pomocí programovacího jazyk C# lze programovat klasické konzolové programy, které pro svůj vstup a výstup využívají příkazový řádek. Zajímavější jsou však aplikace s využitím knihoven Windows.Forms. Jejich použitím lze vytvářet oblíbené formulářové aplikace pro Windows. [1]

1.2.2 WCF

WCF je robustní technologie umožňující tvorbu distribuovaných aplikací prakticky na libovolném komunikačním protokolu. Dostupnost této technologie začíná od .NET Frameworku 3.0. Sjednocuje většinu dřívějších technologií určených pro komunikaci jako například .NET remoting včetně webových služeb a dalších.

WCF lze definovat jako Service-Oriented Architecture, neboli servisně orientovaná architektura. WCF je jednotný framework, pomocí kterého vytváříme SOA aplikace. Zjednodušeně lze říci, že se jedná o komunikaci (výměnu dat) aplikace s aplikací, nebo lépe, služby se službou, s klientem. Konkrétněji tedy jsou WCF distribuované aplikace, nejčastěji jako server (WCF služba) a klienti (aplikace napsané v .NET jazyce).

WCF je obrovské zjednodušení práce pro programátory a také nabízí mnoho vlastností, které danou technologii ještě zlepšují. Co se týče rychlosti komunikace, tak je na tom WCF naprosto stejně, ne-li lépe. Další velkou výhodou se stává jednoduchá možnost nastavení bezpečnosti a dalších vlastností služby. [2] [3] [4]



Obrázek 1 – Ukázka komunikace služby WCF a Klienta

Zdroj: http://relentlessdevelopment.wordpress.com/2010/03/30/wcf-overview/

Služba je systém poskytující jeden nebo více endpointů (koncových bodů), které slouží pro příjem a odesílání SOAP zpráv (messages). Služba publikuje metadata. Tvoří ji tři základní části:

- Třída služby, což je samotná implementace služby.
- Hostovací prostředí neboli místo, kde služba poběží.
- Endpoint.

Endpoint, což je ve službě místo, které slouží k přijímání a odesílání zpráv. Skládá se ze tří částí. Adress, Binding a Contract. Adresa říká, kde služba běží, tedy kam budou zasílány zprávy. Binding označuje, jakým způsobem bude služba komunikovat, tedy jaký komunikační protokol bude zvolen, jaké kódování, ale také výběr bezpečnosti, sessions, transakcí atd. Contract specifikuje rozhraní, které služba poskytuje, metody a další. Contract je nezávislý na volbě adresy a bindingu.

SOAP zprávy jsou zprávy zasílané mezi klientem a serverem. Také jsou nezávislé na přenosovém protokolu. Jedná se o požadavek a odpověď po komunikačním kanálu.

Metadata slouží k popisu služby. Tento popis specifikuje všechna data, pomocí kterých může být nakonfigurován klient. Díky tomu klient pozná, na jaké adrese a jakém protokolu služba běží.

Service contract

Service contract definuje metody, které může klient zavolat. Označuje se atributem [ServiceContract], který se přidá před rozhraní nebo třídu. Jednotlivé metody se označí atributem [OperationContract]. [2] [3]

```
[ServiceContract]
interface IMyContract
{
   [OperationContract]
   string MyMethod(string text);

   //Will not be part of the contract
   string MyOtherMethod(string text);
}
```

Obrázek 2 – Ukázka zdrojového kódu ServiceContract

Zdroj: Programming WCF Services [4]

```
class MyService : IMyContract
{
   public string MyMethod(string text)
   {
      return "Hello " + text;
   }
   public string MyOtherMethod(string text)
   {
      return "Cannot call this method over WCF";
   }
}
```

Obrázek 3 – Ukázka zdrojového kódu implementace rozhraní

Zdroj: Programming WCF Services [4]

Toto rozhraní poté implementuje třída umístěná na serveru. V jednotlivých metodách, které jsou definovány rozhraním, se doplní tělo, které bude vykonávat požadovanou funkci. Tyto metody jsou později volány klientskými aplikacemi.

Data contract

Data contract definuje, které datové typy se budou posílat pomocí služby mezi klientem a serverem. Popisuje strukturu těchto přenášených dat. Pokud nastane událost, že je nutné posílat jiné než základní datové typy, například int, string, musí být definován data contract. Před třídou, případně strukturou nebo výčtovým typem se přidá atribut [DataContract]. Členy data contractu jsou pak označeny jako [DataMember]. [2] [3]

```
[DataContract]
struct Contact
{
   [DataMember]
   public string FirstName;

   [DataMember]
   public string LastName;
}
```

Obrázek 4 – Ukázka zdrojového kódu DataContract

Zdroj: Programming WCF Services [4]

Message contract

Message contract popisuje strukturu SOAP zprávy. Může být typový nebo netypový. Umožňuje specifikovat, jestli zpráva půjde do hlavičky nebo těla zprávy.

```
[MessageContract]
public sealed class FindBooksRequest
{
    private string userName;
    private string bookTitle;

    [MessageHeader]
    public string UserName
    {
        get { return userName; }
        set { userName = value; }
    }

    [MessageBodyMember]
    public string BookTitle
    {
        get { return bookTitle; }
        set { bookTitle = value; }
    }
}
```

Obrázek 5 – Ukázka zdrojového kódu MessageContract

Zdroj: http://www.vyvojar.cz/Articles/457-wcf-kontrakty.aspx

Message contracts jsou na rozdíl od data contracts určené pro specifikaci operace služby. Nejsou určené pro znovupoužití a sdílení. Message contract se definuje přidáním atributu [MessageContract] k deklaraci třídy. Jednotlivé části zprávy musí být definovány atributy [MessageHeader] jako hlavičku a [MessageBody] jako tělo zprávy. [2] [3]

Binding popisuje způsob komunikace endpointů WCF služeb. Skládá se z několika složek, které charakterizují způsoby komunikace. Například ve složce transport se vybírá protokol, který bude použit pro komunikaci služby s klienty.

Složky tvořící binding:

- transport (TCP, Named Pipes, HTTP, MSMQ, ...),
- encoding (text, binary, ...),
- security,
- reliable session,
- transakcie (context flow).

Binding	Interoperability	Security	Session	Transactions	Duplex	Encoding
BasicHttpBinding	Basic Profile 1.1	Transport, Message, Mixed	No	No		Text
WSHttpBinding	WS	Transport, Message, Mixed	Transport, Reliable Session	Yes		Text
WSDualHttpBinding	WS	Message	Reliable Session	Yes	Yes	Text
WSFederationHttpBinding	WS-Federation	Message, Mixed	Reliable Session	Yes	No	Text
NetTcpBinding	.NET	Transport, Message, Mixed	Reliable Session, Transport	Yes	Yes	Binary
NetNamedPipeBinding	.NET	Transport	Transport	Yes	Yes	Binary
NetMsmqBinding	.NET	Message, Transport	No	Yes	No	
NetPeerTcpBinding	Peer	Transport	No	No	Yes	
MsmqIntegrationBinding	MSMQ	Transport	No	Yes		

Tabulka 1 – Způsoby komunikace

Zdroj: http://www.vyvojar.cz/Articles/459-wcf-bindings.aspx

protokoly a jejich vlastnosti:

- BasicHttpBinding – pro komunikaci webových služeb splňujících WS-Basic Profile.
- WSHttpBinding – zabezpečený a interoperabilní binding bez podpory duplex kontraktů.
- WSDualHttpBinding – s podporou duplex kontraktů.
- WSFederationHttpBinding – podpora protokolu WS-Federation.
- NetTcpBinding – zabezpečený a optimalizovaný binding pro komunikaci WCF aplikací.
- NetNamedPipeBinding – komunikace WCF aplikací v rámci jednoho PC.
- NetMsmqBinding – komunikace pomocí MSMQ.
- NetPeerTcpBinding – vícepočítačová komunikace.
- MsmqIntegrationBinding – komunikace mezi WCF aplikací a již existující MSMQ aplikací. [5]

1.2.3 Databáze

Pojmem databáze se myslí úložiště dat. V době začátků byly databáze členěny hlavně hierarchicky, případně síťově. V roce 1970 byly položeny základny dnes nejrozšířenější databáze relační. V dnešní době se kromě databází relačních začínají prosazovat databáze objektové a objektově-relační. Tyto databáze mají velké množství výhod, protože není zcela nutné znát příkazy používané v databázích relačních, ale je možné ukládat přímo celé objekty. V kombinaci s objektově programovacím jazykem dochází k obrovskému zrychlení a zjednodušení práce s databází. [6]

Databáze Db4o

Databáze Db4o je produktem americké společnosti a představuje objektově orientovanou databázi. Jádro tvoří .NET knihovna a všechna data jsou ukládána do souboru na pevném disku. Dotazovat se na uložené objekty lze pomocí jazyku/rozhraní "query by example" (QBE) nebo pomocí vnitřního dotazovacího rozhraní "Query API". Pro někoho se možná stane zásadní nevýhodou nemožnost dotazovat se na data pomocí SQL. Práce s objektovou databází nabízí mnohem jednodušší práci s daty než klasické relační databáze. [7]

Databáze MySQL

Relační databáze MySQL je typu DBMS. Vychází z deklarativního programovacího jazyka SQL. Jedná se o Open Source. Díky své licenci a rychlosti se velmi často využívá právě databáze MySQL. Jedná se o jednoduchý, malý a rychlý databázový systém. Každá databáze MySQL obsahuje tabulky. Každá tabulka má podle potřeby definované sloupce a také řádky, které tvoří skutečná data. Data jsou do databáze vkládána a vypisována z databáze pomocí speciálních příkazů. Práce s tímto systémem se dá využít v C, C++, Java, Perl, PHP, Python, Tcl, Visual Basic nebo .NET. [8] [9]

Hlavní důvody použití databází:

- Databáze poskytuje rychlejší přístup k datům než soubory.
- Databáze umožňuje přímý přístup k datům.
- Databáze má zabudovaný mechanismus pro paralelní přístup k datům.
- Databáze má zabudovaný systém uživatelských práv.
- Databáze umožňuje pomocí dotazů snadno extrahovat množiny dat, která vyhovují zadaným kritériím.

V této aplikaci bylo nutné využít databázi. Bylo nutné ukládat data o uživatelích, jejich uživatelských rolích, ale také informace o synchronizovaných a zálohovaných souborech. Nejprve byla využita objektová databáze Db4o. Její použití je naprosto bezproblémové a velmi jednoduché. Později byla databáze Db4o vyměněna za relační databází MySQL. Hlavním důvodem se stalo zadání diplomové práce, které obsahuje možnost výběru z databází MySQL nebo Oracle.

1.3 Pojmy

Cloudové úložiště - Služba, která řeší problém synchronizace souborů a jejich uložení mezi vašimi zařízeními.

Synchronizace souborů – Pokud uživatel chce mít aktuální soubory v několika zařízeních, musí provádět jejich synchronizaci. Typicky se provádí synchronizace pomocí prostředníka, serveru. V jeho úložišti jsou data nejen uložena, ale také jsou neustále aktuální. Když nastane změna souboru v PC1, proběhne odeslání této změny na server. Poté se změna oznámí všem ostatním počítačům, které uživatel využívá. Problém nastává ve chvíli kdy na PC1 dojde ke změně souboru, ten je odesílán na server, ale server má aktuálnější soubor. Dochází k tzv. konfliktu synchronizace.

Zálohování – Pojem záloha má význam v archivaci aktuálních dat. Existuje několik typů záloh. Nejznámější a nejpoužívanějším typem je úplná záloha, kde se provede záloha všech požadovaných dat. Další typy inkrementální a diferenciální provádí zálohy jen změněných dat. Zálohování dat se obvykle provádí na různá úložná média, vzdálené počítače nebo do cloudových úložišť.

Token – Tento pojem se aplikaci vyskytuje jako jednoznačný bezpečnostní identifikátor uživatele. Po úspěšné autentifikaci a autorizaci obdrží uživatel nový token. Ten má svou dobu platnosti a během jeho platnosti může využívat služby poskytované serverem. Po jeho vypršení se automatiky požádá o nový token. Token se skládá z 128bitového celého čísla generovaného pomocí funkce Guid.

Webová služba – Služba pracující na webovém serveru, která nemá uživatelské rozhraní. Tato služba může obsahovat webové metody, které jsou poskytovány dalším aplikacím. Webové služby jsou někdy také nazývány webové služby XML, což je značkovací jazyk. Tyto služby jsou součástí světa Microsoft .NET, ve kterém komunikují pomocí protokolu HTTP a SOAP protokolu.

1.4 Analýza dostupných řešení na trhu

Na softwarovém trhu se v dnešní době vyskytuje velké množství produktů, které poskytují podobnou funkcionalitu jako aplikace, která byla vytvářena jako předmět této práce. Podobné aplikace lze rozdělit do dvou hlavních skupin. První skupinu budou zastupovat aplikace, které využívají jako své hlavní úložiště cloud. Druhou skupinu zastoupí aplikace, které slouží k synchronizaci a zálohování mezi dvěma zařízeními.

Jelikož má tato aplikace primární určení do menších firem, předpokládejme, že podnik nedisponuje vlastním cloudovým úložištěm. Tím se samozřejmě výrazně sníží celková cena, kterou firma musí za hardwarové vybavení zaplatit. Přesto jsou zde zástupci s cloudovým úložištěm. Hlavním důvodem k jejich výběru je velmi podobná funkcionalita klientských aplikací. Neznalý uživatel nezná principy, které jsou využity k ukládání dat na serveru a v konečném měřítku ho to ani nemusí zajímat. Pro běžné uživatele jsou nejdůležitější funkce, které poskytuje klientská aplikace.

1.4.1 Dropbox

Dropbox využívá cloud computingu a umožňuje tak uživateli ukládat a sdílet soubory a složky s ostatními uživateli po internetu pomocí synchronizace souborů. Jedná se o velmi oblíbenou aplikaci mezi uživateli. Zaměřuje se na uživatele domácí i firemní. Poskytuje speciální služby pro firemní sféru. Tato služba se jmenuje Dropbox for Business. Cena začíná na 795 USD za rok pro pět osob a úložná kapacita není omezena. Pro firmu s padesáti zaměstnanci by to znamenalo více než 127 000 Kč ročně. O zabezpečení dat se stará 256bitové AES šifrování a ke komunikaci se využívá SSL protokol. [10]

Obrázek 6 – Dropbox

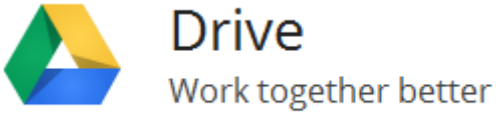
Zdroj: https://www.dropbox.com/business

Shrnutí aplikace:

- vysoká rychlost,
- výborné zabezpečení dat i komunikace,
- neomezená kapacita,
- vysoká cena,
- aplikace pro mnoho platforem.

1.4.2 Google drive

Další obrovskou službou v oblasti ukládání dat a synchronizace v cloudovém úložišti je Google drive. Oproti aplikaci Dropbox nabízí firma Google rozdílnou cenovou politiku. Cena není určena jen počtem uživatelů, ale i kapacitou úložiště. Základních 5 GB nabízí za cenu 50 USD za uživatele na rok. Pro padesát uživatelů se jedná o částku kolem 50 000 Kč za rok. Navýšit kapacitu lze až do 16 TB pro uživatele. Tuto kapacitu nabízí za cenu přesahující 27 000 Kč ročně. Pro zajištění bezpečnosti a spolehlivosti jsou šifrována spojení se servery Google, ukládání souborů v reálném čase. Dochází souběžně s přenosem dat replikovaní do dalšího úložiště. Obsahuje také integrované obnovení dat po havárii. [11]



Obrázek 7 – Google Drive

Zdroj: https://www.google.com/intl/cs/enterprise/apps/business/products.html

Shrnutí aplikace:

- rychlost nižší než u aplikace Dropbox,
- výborné zabezpečení dat i komunikace,
- omezená kapacita,
- nízká cena za uživatele, cena za kapacitu podle potřeby,
- aplikace pro mnoho platforem.

1.4.3 Box

Posledním zástupce cloudových úložišť určených do podnikové sféry se stává Box. Tato služba poskytuje podobné služby jako předchozí dva zástupci. Zabezpečení je na podobné úrovni. Také poskytuje 256bitový SSL protokol pro přenosy dat a 256bitové AES šifrování pro uložení souborů v datovém úložišti. Určitě tato služba nabízí nejzajímavější cenovou nabídku pro menší firmy. Nabízí 1 TB úložného prostoru pro 3 až 500 uživatelů za cenu 13 EUR za jednoho uživatele na měsíc. Pro malý podnik se jedná o velmi zajímavou nabídku. Pokud by se ale jednalo o firmu s padesáti zaměstnanci, přesahovala by cena 200 000 Kč za rok. [12]

Obrázek 8 – Box

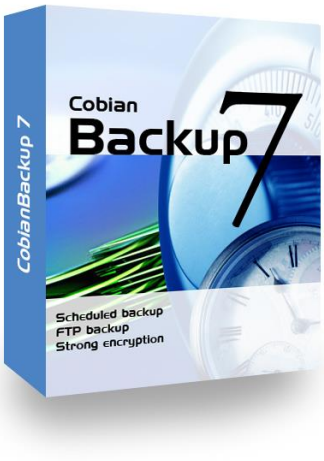
Zdroj: https://www.box.com/s/nfzony6vrrhlo98i07f1/1/424734612

Shrnutí aplikace:

- slušná rychlost,
- výborné zabezpečení dat i komunikace,
- pro menší firmu velmi zajímavá cenová nabídka za 1 TB úložného prostoru,
- pro větší firmu pouze individuální plán,
- aplikace pro mnoho platforem.

1.4.4 Cobian Backup

Program Cobian Backup slouží k zálohování dat. Poskytuje sice všechny typy zálohování (úplná, inkrementální, diferenciální), ale neumí synchronizaci. Diferenciální (rozdílový) typ zálohy vytvoří nejdříve plnou zálohu, další záloha však bude přidávat jen soubory změněné či vytvořené od poslední plné zálohy. Díky tomu se ušetří velké množství místa. V případě obnovy pak stačí zkopírovat do původního umístění plnou zálohu a poslední rozdílovou zálohu. V případě nastavení častého pravidelného zálohování by docházelo k udržování téměř aktuálních dat. Aplikace poskytuje ukládání zálohovaných souborů, jak na lokálním počítači, tak LAN nebo na FTP. Také poskytuje šifrování a možnost zabalení do archivu ve formátu *.zip. Aplikace se nabízí zdarma jako freeware. [13]



Obrázek 9 – Cobian Backup

Zdroj: http://www.megadownloads.nl/statisch/afbeeldingen/uploads/downloads/336(1).jpg

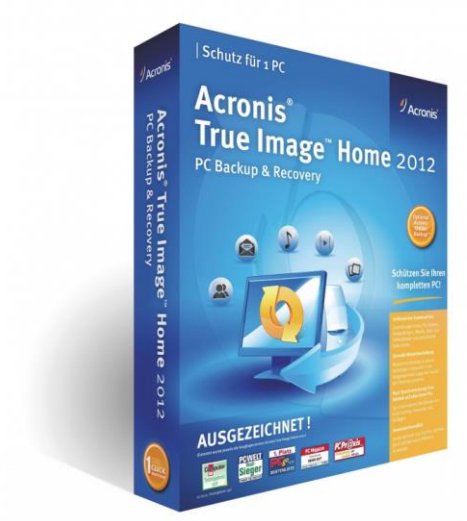
Pokud by byla aplikace využita ve firmě a byl by přístupný po síti server nebo alespoň FTP, bylo by možné zálohy provádět na podobném principu jako při synchronizaci.

Shrnutí aplikace:

- neposkytuje synchronizaci,
- nastavení záloh na velmi vysoké úrovni,
- se správným nastavením lze získat téměř plně aktuální data,
- zabezpečení pomocí šifrování a zaheslování souborů,
- aplikace se nabízí jako freeware.

1.4.5 Acronis True Image

Podobných aplikací jako Cobian backup, které poskytují jako svou hlavní funkcionalitu zálohování, lze na trhu nalézt mnoho. Úplně jiný princip zálohování poskytuje program Acronis True Image. Samozřejmě zvládne podobné zálohování složek, stejně jako ostatní, ale hlavním rozdílem je, že dokáže vytvořit image z celého pevného disku. Image vytvoří i s master boot table, která obsahuje informace o umístění operačního systému na pevném disku. Takto se provede záloha celého pevného disku. Záloha obvykle trvá okolo patnácti minut a obnova stejně dlouho. Tímto způsobem lze klonovat pevné disky. Neposkytuje však ochranu záloh heslem a šifrováním. Aplikace je poskytována za cenu 1052 Kč bez DPH za jednu licenci. [14]



Obrázek 10 – Cobian Backup

Zdroj: http://www.devler.eu/2011/09/07/acronis-true-image-home-2012/

Shrnutí aplikace:

- neposkytuje synchronizaci,
- výborné zálohování,
- záloha celého pevného disku do image,
- padesát licencí by stálo přes 50 000 Kč.

1.4.6 Porovnání konkurenčních aplikací

Konkurenční aplikace se rozdělují podle funkčnosti do dvou kategorií. Rozdíl ve funkčnosti znamená rozdíl mezi synchronizací a zálohováním. Synchronizaci lze pochopit jako dokonalejší zálohování, ale ne vždy je využití synchronizace vhodnější. Problém nastává při synchronizaci složky, ve které jsou soubory na sobě závislé. V případě změny jednoho z nich dojde k synchronizaci. Pokud by uživatel později chtěl vrátit vše do původní podoby, musel by ručně hledat jednotlivé soubory a porovnávat jejich čas změny tak, aby získal celou složku před změnou dat. Jako ideální příklad si lze představit synchronizování složky, ve které programátor vyvíjí aplikaci. Každá aplikace obsahuje mnoho souborů. Pokud některý změní, provede se synchronizace. Poté pokud provede nějakou chybu a chce se vrátit k původnímu stavu, musí hledat v historii verzí ten správný soubor. Kdežto záloha se provede jako celek v daném čase.

Pokud by si firma měla vybrat vhodný nástroj k synchronizaci do cloudového úložiště, má poměrně velkou nabídku služeb, mezi který mi si může vybírat. Rozdíly mezi aplikacemi DropBox, Google Drive a Box jsou poměrně zanedbatelné. Drobné rozdíly jsou mezi rychlostmi. Zabezpečení je na velmi slušné úrovni u komunikace, ale také uložení samotných dat se šifruje. Hlavní rozdíly jsou v ceně a kapacitě. Velmi důležitým kritériem se také stává počet uživatelů této služby v podniku. Do velmi malého podniku by se daly doporučit služby aplikace Box, která nabízí nejzajímavější cenu. Při padesáti porovnávaných uživatelích záleží volba pouze na kapacitě, kterou jednotliví uživatelé potřebují. Pokud by dostačovaly menší kapacity úložišť, stává se nejlepší variantou Google Drive. Ale pokud by firma potřebovala větší kapacity, určitě by využila služby aplikace Dropbox, který poskytuje neomezené úložiště a cena není o mnoho vyšší než cena služeb u společnosti Google.

Aplikací pro zálohování lze na trhu nalézt velké množství. Většina z nich poskytuje podobné funkce týkající se rozdílných typů záloh, různého plánování, ale také umístění, kam se zálohovaná data umístí. Samozřejmě lze zálohovaná data různě zabezpečit, ale tuto funkci neposkytují již tyto programy. Zálohy na firemní server lze jednoduše umístit, ale o zabezpečení se bude muset postarat server, respektive správce sítě.

Programy Cobian Backup i Acronis True Image nabízí velmi podobné funkce, co se týká klasických záloh. Cobian Backup má hlavní výhodu ve své ceně. Jedná se totiž o freeware aplikaci. Acronis True Image nabízí velmi zajímavou funkci navíc. Dokáže naklonovat celý pevný disk.

Při prvním pohledu na tuto funkci, si každý běžný uživatel řekne, že se jedná o zcela zbytečnou funkci. Když pomineme zálohování pevného disku pro běžné kancelářské uživatele a přesuneme se do výrobní části, nalezneme obrovské využití. Každá firma, která využívá k výrobě roboty, se čas od času setká s problémem, že robot přestal pracovat. Robot většinou obsahuje zcela běžný počítač i s pevným diskem. Než přijede specializovaná firma a robota opraví, výroba stojí. Nejčastější závadou se samozřejmě stává zničení pevného disku. Pevné disky nikdy nevydrží vytížení, které výrobní linky

mají. Záloha tohoto pevného disku trvá obvykle méně než u běžných počítačů, protože disk obsahuje méně dat. Jedná se o délku zhruba deseti minut. Takto vytvořenou zálohu lze po zjištění závady využít a během následujících deseti minut může robot znovu pokračovat v práci. Výběr správné aplikace do firmy záleží přímo na možnostech využití jejich funkcí.

1.4.7 Porovnání s vlastní aplikací

Aplikace, která byla vyvíjena jako předmět této diplomové práce, kombinuje funkce nabízené všemi dříve zmíněnými službami. V porovnání s cloudovými službami nenabízí sice výhody vyplývající z úložiště cloudu, ale jinak poskytuje téměř stejné funkce. Funkce zálohování jsou v aplikaci velmi jednoduché, protože tvoří pouze doplňkovou službu.

Hlavní výhodou cloudu jsou téměř automatické možnosti uložení na mnoha různých místech po světě. Tato aplikace je určena přímo do firemní sítě. Ideálně by nebyla přístupná z internetu. Tím se velmi posílí bezpečnost aplikace i celé firmy. Většina podniků má mezi svými pobočkami vytvořené VPN tunely, aby byla umožněna přímá síťová komunikace, což poskytuje další zabezpečení dat. Výše zmíněná cloudová úložiště poskytují sice zabezpečenou komunikaci, ale stále přes internet. Vlastní aplikace tedy získá obrovskou výhodu právě tím, že bude nasazena přímo na server uvnitř podniku. Tento server stejně již většina firem vlastní z důvodu centrálního ukládání dat z výroby a obchodu.

Další obrovskou výhodu nabízí v oblasti kapacity úložiště a odvíjené ceny. Není nutné platit ani paušální poplatky za uživatele ani za potřebnou kapacitu. Pokud začne kapacita úložiště na serveru docházet, správce sítě zakoupí další pevné disky, případně celé úložiště a připojí. Dále lze disky v úložišti zrcadlit a zajistit tím vyšší bezpečnost proti ztrátě dat na nich. Komunikace uvnitř firmy by sice nemusela být zajištěna, ale přesto aplikace zabezpečenou komunikaci využívá. Zabezpečení lze dále zvýšit. Toto je podrobněji popsáno v kapitole zabezpečení aplikace. Poslední obrovský rozdíl nalezneme v rychlosti přenosu dat. Uvnitř firmy bude přenos dat mnohem rychlejší než po internetu třeba přes celý kontinent.

V porovnání s aplikacemi určenými primárně pro zálohování snad ani nelze vlastní aplikaci srovnávat. Služba zálohování, kterou poskytuje vlastní aplikace, zvládá pouze základní zálohování. Hlavním důvodem je primární funkce synchronizace. Proto není nutné využívat zálohování. Samozřejmě v některých případech se záloha stává vhodným doplňkem plné funkcionality, a proto ji tato aplikace poskytuje. Ostatní zálohovací programy poskytují různé typy záloh, velmi pokročilé plánování a různá úložiště. Tato aplikace poskytuje jednoduché zálohy, možnost naplánovat pravidelné zálohování v libovolný čas během týdne a zálohování vždy probíhá přímo do úložiště na serveru.

Hlavní výhody v porovnání s konkurenčními službami:

- Bezpečnost:
    - Data jsou umístěna přímo na serveru uvnitř firmy.
    - Data nemusí opustit firmu.
    - Šifrovaná komunikace.
- Cena:
    - Není nutné platit měsíční poplatky.
    - Cena se neodvíjí ani od kapacity ani od počtu uživatelů.
    - Náklady na aplikaci jsou z největší části v hardwaru, který ale podnik velmi často již vlastní.
- Kapacita
    - Žádné limitování kapacity ze strany poskytovatele.
    - Kapacita je limitována pouze hardwarovými prostředky ve firmě.
- Rychlost:
    - Přenos dat není limitován rychlostí internetu.
    - Rychlost je limitována pouze hardwarovými prostředky ve firmě.
- Dostupnost:
    - Při přenosu přes internet může být některý z uzlů využitých k přenosu dat nedostupný, a proto nebude prováděn žádný přenos dat.
    - Odstranit poruchu uvnitř firmy řeší správce sítě.
    - Nedostupnost uvnitř firmy znamená mnohem větší problémy než nefunkční synchronizaci a zálohování. Problém je v zastavení celé výroby.
- Služby:
    - Služby synchronizace jsou podobné s konkurenčními.
    - Zálohování do centrálního úložiště.

Nevýhody v porovnání s konkurenčními službami

- Bezpečnost:
    - Data nejsou umístěna ve více vzdálených úložištích.
    - Data nejsou v úložišti šifrována.
- Cena a kapacita:
    - Kapacitu úložiště musí hlídat správce sítě. Navýšení kapacity pouze po rozhodnutí vedení firmy o nákupu nového úložiště.
- Dostupnost:
    - Jakýkoliv problém uvnitř firmy musí řešit správce sítě.
    - Vyšší zatížení serveru.
- Služby:
    - Služby poskytované specializovanými firmami mají tým techniků, kteří neustále monitorují stav celého úložiště a problémy mohou vyřešit.
    - Zálohování je pouze na velmi jednoduché úrovni.
    - Pravidelné zálohování pouze jednou týdně.
    - Nemožnost určení místa zálohy.

Možná řešení nevýhod této aplikace

Většinu nevýhod této aplikace lze vyřešit. Některé jsou v mnoha podnicích řešeny automaticky, ale záleží na konkrétním použití.

Jednotlivá úložiště má většina serverů minimálně chráněno diskovými poli RAID, která podstatně zvyšují bezpečnost. Dále by bylo možné data synchronizovat s úložištěm v jiné pobočce pomocí tunelu VPN. Jedná se ale o poměrně nákladnou záležitost. Data nejsou v úložišti šifrována z několika důvodů. Hlavním je rychlost použití dat. Zašifrovaná data musí server nejprve dešifrovat, což ho zbytečně zatěžuje. Šifrovat data není nutné z důvodů nutnosti správného zabezpečení serverů ve firmě. K serverům by nikdy neměl mít přístup nepovolaný pracovník.

Nákup nových pevných disků případně celého pole musí vždy schválit vedení dané společnosti. Většinou s tím ale není žádný problém, protože se jedná o jednorázovou investici, která se v porovnání s paušálními poplatky stává mnohem výhodnější.

Využití této aplikace uvnitř firmy přidá mnoho práce správcům sítě. U konkurenčního softwaru by pouze nainstalovaly programy na klientské počítače. Zde bude muset správce hlídat dostupnou velikost úložiště a podle toho včas reagovat. Navíc může nastat nějaký problém například nekompatibilitě po aktualizacích operačního systému serveru a tyto problémy musí řešit. Na druhou stranu je toto vše jeho běžnou pracovní náplní a s každým takovým problémem by si měl šikovný správce sítě velmi rychle poradit.

Vytížení serveru sice vyšší opravdu je, ale přenos dat po internetu bude muset také projít síťovou infrastrukturou celé firmy, takže ten rozdíl se dá zanedbat.

Služby synchronizace jsou velmi obdobné jako u konkurence. Cloudová úložiště neposkytují přímou možnost zálohování. Sice umožňují verzování historie souborů, ale stále se nejedná o plnohodnotnou zálohu. Zálohování bylo takto jednoduše navrženo, protože v mnoha firmách je nutné provést zálohu jednou týdně. Typicky na konci pracovního týdne. Nastavení libovolného času poskytuje pouze z důvodu různé pracovní doby zaměstnanců. Poskytovat každodenní nebo zálohy každou hodinu je v kombinaci se službou synchronizace zcela zbytečné. Také byl velmi kladem důraz na co největší jednoduchost nastavení. Ve specializovaných aplikacích pro zálohování lze nastavit mnoho různých nastavení. Nejdůležitějším je úložiště zálohy. Záloha na stejném počítači ztrácí smysl. Proto možnost nastavení umístění zálohy vlastní aplikace neumožňuje a všechny zálohy jsou umístěny přímo v úložišti na serveru.

Pokud by společnost na některých nastaveních navíc trvala, byla by samozřejmě možnost individuální úpravy celé aplikace. Implementace jednotlivých nastavení pouze v klientské aplikaci by neměl být problém a tato možnost by mohla být některé společnosti zajímavá.

2 Implementace aplikace

Aplikace, která tvoří předmět této diplomové práce, slouží k synchronizaci dat mezi klientskou aplikací a serverem. Další poskytovanou službou této aplikace je jednoduché zálohování. Skládá se ze dvou samostatných projektů: klientské aplikace a serverové části.

2.1 Postup vývoje

Vývoj této aplikace byl poměrně dlouhý a ne vždy směřoval tím správným směrem. Postupně byly vytvořeny tři verze. První dvě nebyly nikdy zcela dokončeny, protože se vždy vyskytl nějaký problém, který by nebylo možné překonat, nebo existovalo mnohem lepší a často jednodušší řešení.

První verze byla špatná už od počátku. Velmi špatně byla zvolena už technologie k přenosu dat mezi aplikacemi. Tvorba dat ke komunikaci, ale i poté samotný přenos, byly velmi náročné operace pro systém. Také tato technologie byla náchylná k tvorbě chyb. Jednalo se o socketovou komunikaci, kde přenášená data byla pole bytů. Téměř vždy bylo nutné přenést více informací zároveň, což znamenalo nutnost do bytového pole spojit různé informace na jedné straně a na druhé straně je opět rozdělit. Toto bylo velmi neefektivní a náchylné na chyby. Proto později nastala volba pokračovat pomocí jiné technologie. Konkrétně se jedná o Windows Communication Foundation, což je obrovská technologie určená k síťové komunikaci napříč mnoha platformami.

Druhá verze již byla založena na zmíněné technologii WCF. Základním prvkem této technologie je služba, která poskytuje koncový bod. Na tomto bodě služba publikuje metadata, která slouží k popisu služby. Pomocí této technologie bylo vytvořeno rozhraní služby, která poskytovala všechny nutné metody pro komunikaci. V této verzi byla správně zvolená technologie. Komunikace mezi aplikacemi se velmi zjednodušila, ale samotná tvorba rozhraní byla poměrně náročná z důvodu obsáhlosti celé technologie. Při tvorbě nastala chyba ve špatném pochopení principu přenášení souborů. Po zjištění tohoto problému muselo nastat velké rozhodnutí, zda celou aplikaci přepisovat nebo raději začít znovu a lépe. Protože by přepisování stávající aplikace znamenalo velké riziko vnesení dalších chyb, byla zvolena varianta další verze.

Třetí verze se stala verzí poslední. Tato verze postupně dostávala další funkce, které byly zvoleny v požadavcích až do dnešní podoby. Celkově byla poměrně hodně změněna architektura celé aplikace. Také došlo k podstatnému zlepšení grafického rozhraní pro uživatele, aby bylo více příjemné pro práci.

2.2 Struktura aplikace

Aplikace byla rozdělena do dvou samostatných projektů. Prvním projektem je serverová strana nazvaná ServerService poskytující veřejné rozhraní WCF služby určené ke komunikaci s klientskou aplikací. Dále tento projekt obsahuje třídy nutné ke komunikaci s databází, práci s diskovým úložištěm, synchronizaci a zálohování.

Druhý projekt nazvaný ClientDesktop obsahuje klientskou desktop aplikaci. Ta obsahuje především grafické uživatelské rozhraní a komunikuje s rozhraním služby předchozího projektu.

2.2.1 Serverová strana

Projekt tvořící serverovou stranu obsahuje již dříve zmíněné rozhraní WCF služby. Toto rozhraní slouží ke komunikaci s klientskými aplikacemi a také ho implementuje další třída, pomocí které se provádí operace na serveru. Rozhraní obsahuje metody nutné ke všem funkcím, které jsou ke správné funkčnosti aplikace nutné.

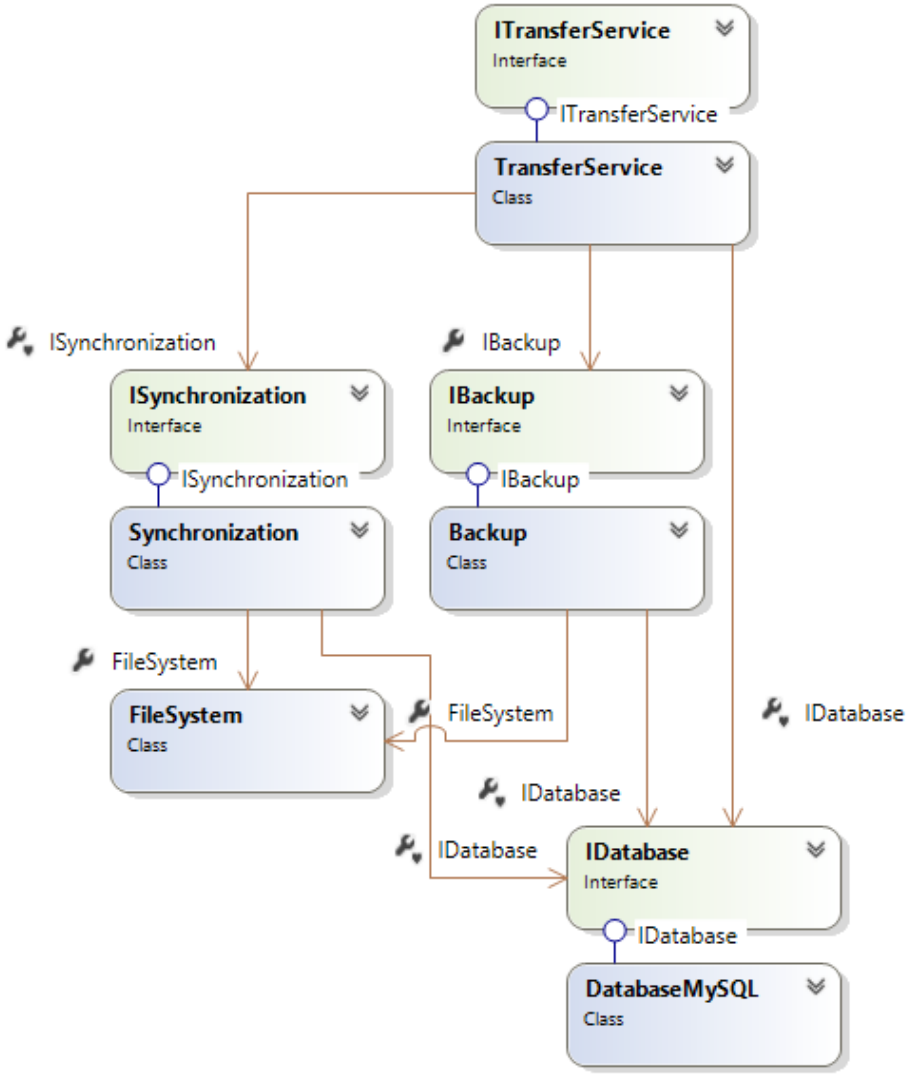
```
[ServiceContract]
interface ITransferService
{
    [OperationContract]
    bool Connect();

    [OperationContract]
    Response Autorizace(String nick, String heslo);

    [OperationContract]
    Response Reautorizace(Request pozadavek);

    [OperationContract]
    RemoteFile DownloadFile(RequestMessage request);

    [OperationContract]
    ResponseMessage UploadFile(RemoteFile request);

    [OperationContract]
    ResponseFiles isNewFileAtServer(Request pozadavek);

    [OperationContract]
    ResponseFiles getFileList(Request pozadavek);

    [OperationContract]
    ResponseMessage backupToServer(RemoteFile request);

    [OperationContract]
    RemoteFile backupFromServer(RequestMessage request);

    [OperationContract]
    ResponseFiles getBackupFileList(Request request);
```

Obrázek 11 – Ukázka rozhraní služby WCF

Zdroj: vlastní

Mezi nejdůležitější metody patří metody k připojení, autentifikaci a autorizaci uživatele. Bez těchto metod by nebylo možné využívat další funkce. Dále obsahuje metody k získání ustáleného stavu po startu klientské aplikace, metody odeslání a přijetí souboru a metody k vytváření a stahování zálohovaných souborů.

Rozhraní dále obsahuje velké množství metod k pomocným činnostem. Jedná se o metody pracující s uživatelovými složkami, sdílení složek mezi různými uživateli a informace o uživatelích. Dále jsou v rozhraní definovány DataContracty a MessageContracty. Třídy, které jsou viditelné i v klientských aplikacích. Slouží k předávání požadovaných dat mezi aplikacemi. Velmi často tyto třídy pouze obalují další třídy, čímž je dosaženo komunikace požadavku a odpovědi. MessageContracty jsou využity pouze pro přenos souborů. Hlavním důvodem bylo, že mohou přenášet stream. Tím se dosáhne přenosu souborů po blocích, který je mnohem vhodnější z důvodu úspory místa v operační paměti.



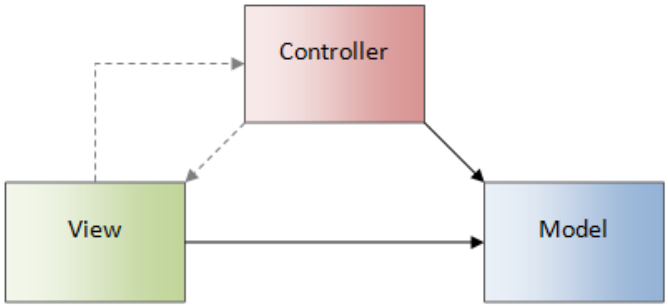
Obrázek 12 – Zjednodušený diagram serverové aplikace

Zdroj: vlastní

Třída TransferService implementuje rozhraní WCF služby. Tato třída je řídící třídou v celém projektu. Dle potřeby komunikuje s rozhraními jednotlivých částí. Jedná se o rozhraní k synchronizaci, zálohování a komunikaci s databází. Třída FileSystem slouží k práci fyzickým úložištěm serveru.

2.2.2 Desktop aplikace

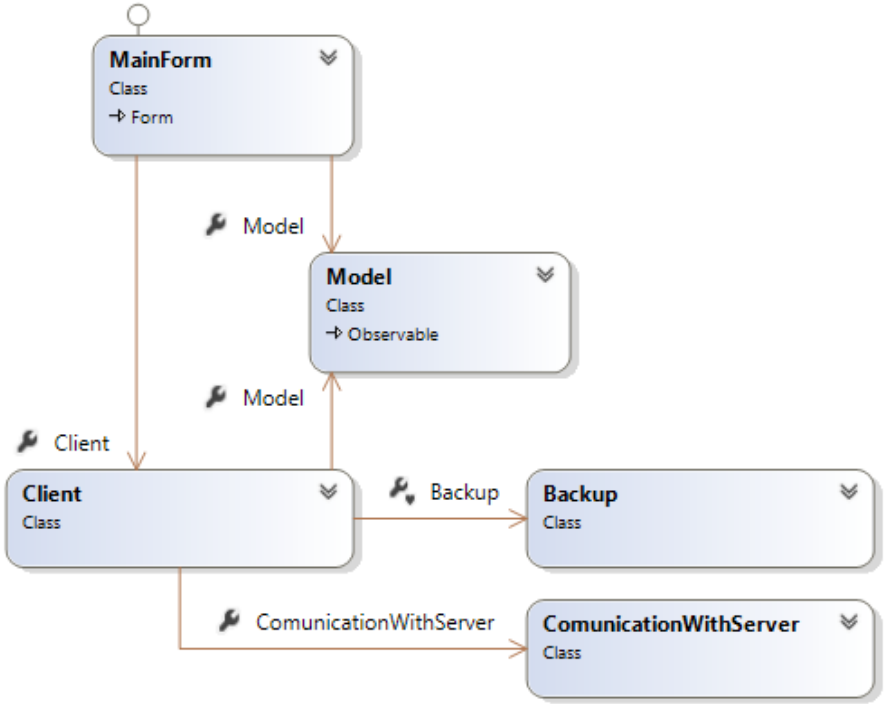
Desktopová aplikace byla navržena podle návrhového vzoru Model-View-Controller. Architektura tohoto návrhové vzoru se skládá ze tří logických částí. Model reprezentuje data a logiku aplikace, View zobrazuje uživatelské rozhraní a Controller má na starosti tok událostí v aplikaci a veškerou aplikační logiku. [15]



Obrázek 13 – Návrhový vzor MVC

Zdroj: http://www.zdrojak.cz/clanky/uvod-do-architektury-mvc/

V této aplikaci obsahuje Model všechna potřebná data, která jsou později zobrazována v grafickém rozhraní. View obsahuje všechny formuláře v projektu. Controller obsahuje veškerou logiku včetně samotné komunikace se službou serverové aplikace. Pokud dojde ke změně dat ve vrstvě Model, dojde k předání informace ke všem zaregistrovaným třídám. V tomto případě hlavně grafickému rozhraní.

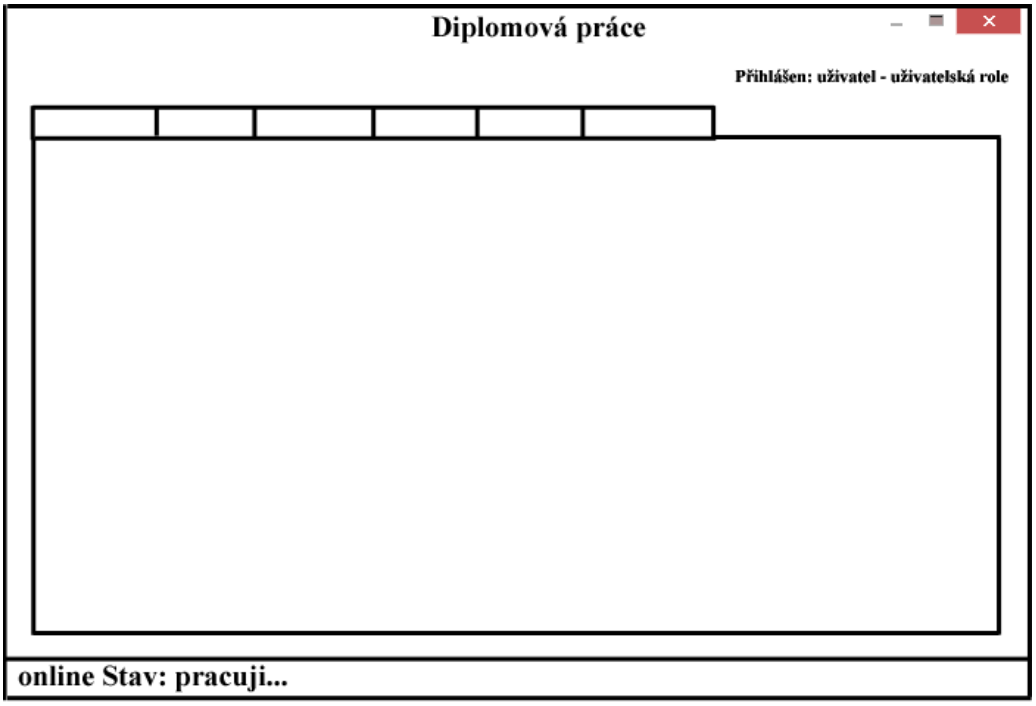


Obrázek 14 – Zjednodušený diagram desktopové aplikace

Zdroj: vlastní

Na předchozím obrázku lze vidět zjednodušený diagram aplikace. Dobře viditelný je návrhový vzor MVC. K informování zaregistrovaných posluchačů o změně dat v Modelu se využívá další návrhový vzor, konkrétně Observer. MainForm reprezentuje vrstvu View, Model obsahuje data a Client reprezentuje Controller. Client dále komunikuje s ostatními třídami aplikace. Nejdůležitější z nich je CommunicationWithServer, která slouží ke komunikaci se službou. Jelikož každá síťová komunikace má nějaké zpoždění, bylo nutné do aplikace zakomponovat vlákna. Pokud by aplikace vlákna neobsahovala, docházelo by k zamrzání celé aplikace během každé komunikace. Protože dochází k síťové komunikaci velmi často, stala by se aplikace nepoužitelnou. Občas jsou vlákna použita tak, aby docházelo k různým činnostem na pozadí aplikace. Jindy se vyčkává na dokončení požadované funkce ve vlákně a poté dojde k zobrazení výsledku.

Každá komunikace mezi desktopovou aplikací a službou hostovanou na serveru se skládá z požadavku a odpovědi. Jedinou výjimku tvoří přihlášení a registrace uživatele. Požadavek vždy obsahuje token, který jednoznačně určuje uživatele, který požaduje data, a také slouží k zaručení bezpečnosti celé aplikace. Odpověď vždy klientskou aplikaci informuje, zda požadovaná funkce proběhla v pořádku a pokud ano, provede požadovanou funkci na serveru, případně předá požadovaná data. V případě, že nastala nějaká chyba, obsahuje odpověď kód a druh chyby, která nastala. Klientská aplikace podle toho samozřejmě zareaguje. Například pokud se nepovedlo nahrání souboru na server, dojde k jeho opětovnému vrácení do fronty odesílaných souborů. Později se pokus o odeslání bude opakovat.



Obrázek 15 – Návrh layoutu desktopové aplikace

Zdroj: vlastní

Před zahájením tvorby klientské aplikace byl navržen jednoduchý layout aplikace. Bylo předem jasné, že bude nutné využít záložek, aby bylo co nejjednodušší zobrazení poměrně velkého množství funkcí v aplikaci. Také bylo nutné zobrazovat stav aplikace a informaci, jestli se aplikace připojila k serveru. Poslední předem známou informací, kterou bylo nutné zobrazovat, se stalo uživatelské jméno a uživatelská role.

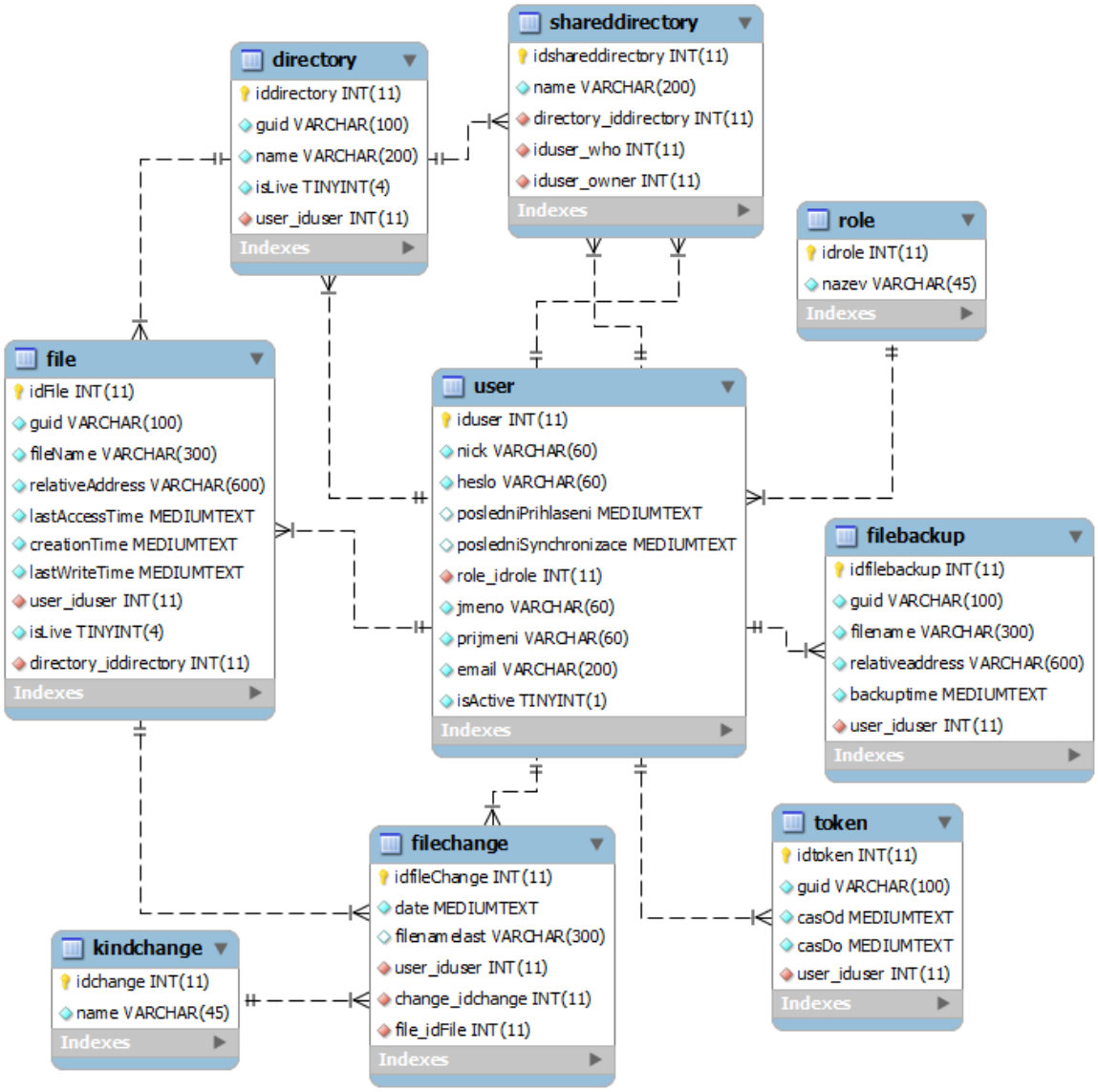
2.2.3 Databázový model

V aplikaci bylo nutné někam ukládat data o uživatelích, souborech a mnoho dalších informací. Nejprve byla využívána databáze Db4o. Tato databáze není klasická relační databáze, ale jedná se o objektovou databázi. Její hlavní výhodou je, že můžeme ukládat celé objekty bez nutných znalostí o příkazech využívaných v relačních databázích. Dalším důvodem proč byla tato databáze zvolena, je jednoduchost instalace. Není nutné instalovat databázový server a všechna data se ukládají do jediného fyzického souboru. Ve finální verzi aplikace již databáze Db4o není využita z důvodu pevného zadání této práce. V zadání jsou uvedeny databáze MySQL nebo Oracle.

Zvolena byla relační databáze MySQL. Návrh této databáze je poměrně jednoduchý a nevyžaduje žádné složité dotazy. Pokud by ovšem byla aplikace nasazena ve velké firmě, s velkým množstvím uživatelů, bylo by nutné provést optimalizace některých dotazů. Třídy komunikující s databází implementují jednotné rozhraní, proto by nebylo obtížné změnit databázi MySQL za libovolnou jinou databázi.

Návrh databáze obsahuje devět tabulek. Tabulky user a token slouží k jednoznačné identifikaci uživatele. Dále je nutné uživateli přiřadit roli, která určuje jeho práva. Další tabulky file, filechange, kindchange directory a shareddirectory slouží k uložení informací týkající se synchronizace dat. Poslední tabulka filebackup obsahuje informace o zálohovaných souborech.

- Tabulka user – Obsahuje informace získané při registraci uživatele. Slouží hlavně k autentifikaci uživatele při přihlášení.
- Tabulka role – Určuje práva, která uživatel vlastní. Prozatím existují role administrátor a uživatel. K autorizaci dochází také při přihlášení.
- Tabulka token – Obsahuje tokeny, které jsou vytvořeny při úspěšném přihlášení uživatele. Další důležité parametry jsou casOd a casDo, které informují o době platnosti tokenu.
- Tabulka filebackup – Obsahuje data o zálohovaných souborech.
- Tabulka file – Každý synchronizovaný soubor je uložen v této tabulce.
- Tabulka directory – Zde jsou uloženy všechny synchronizované složky.
- Tabulka filechange – Pokud nastane nějaká změna souboru, bude tato změna uložena v této tabulce.
- Tabulka kindchange – Obsahuje čtyři parametry změny souboru. Vytvoření, změnu, přejmenování a odstranění souboru.
- Tabulka shareddirectory – Zde jsou uvedeny sdílené složky mezi různými uživateli.



Obrázek 16 – Diagram databáze

Zdroj: vlastní

2.2.4 Zabezpečení aplikace

Zabezpečení v aplikaci bylo rozděleno do několika částí. Všechny jsou velmi důležité. Stejně jako ve většině podobných aplikací je nutné se k použití aplikace přihlásit. Provede se autentifikace. Po přihlášení dojde k autorizaci, čímž se uživateli přiřadí jeho uživatelská role. Dále má aplikace zabezpečení v oblasti přenosu dat mezi klientskou aplikací a rozhraním služby umístěné na serveru.

Po ověření uživatelského jména a hesla s databází na serveru, dojde k přihlášení uživatele. Uživatel získá ověřovací token. Ten tvoří funkce Guid, 128bitové celé číslo. Token je také uložen v databázi na serveru. Při každém požadavku na server dojde k porovnání platnosti tokenu. Token musí být platný, jinak požadavek na funkci nebude proveden a služba vrátí chybu o neplatnosti tokenu. Platnost byla v počátku tvorby aplikace nastavena na pět

minut, ale je možné tuto hodnotu z mnoha různých důvodů změnit. Delší časový interval má vliv na snížení bezpečnosti. Kratší čas bude mít vliv na vyšší nároky datových přenosů, ale za cenu vyšší bezpečnosti, protože čím častěji se bude měnit bezpečnostní token, tím bude celkové zabezpečení aplikace vyšší.

Když token vyprší a aplikace požádá server o provedení libovolné funkce, funkce nebude provedena. Dojde k vrácení chyby o neplatnosti tokenu. Poté aplikace požádá o reautorizaci. Pokud je token neplatný pouze do pěti minut, tento čas byl opět nastaven již při vývoji aplikace, dojde k vrácení nového platného tokenu. Pokud nastane opak, bude vrácena chyba a poté dojde k odpojení aplikace od serveru. Pokud má uživatel nastavené automatické přihlašování, dojde velmi brzy k novému připojení a přihlášení pomocí lokálně uložených informací o uživateli. Jinak bude zobrazen formulář s požadavkem na přihlášení uživatele.

Pomocí autorizace každý uživatel získá svou uživatelskou roli. V této aplikaci se prozatím vyskytují role administrátor a uživatel. Uživatel má práva využívat všechny běžné funkce, které aplikace poskytuje. Administrátor má stejná práva jako uživatel, plus navíc může spravovat ostatní uživatele. Správa se týká možnosti změnit uživatelské informace včetně možnosti zakázat či povolit uživateli přístup do aplikace.

O zabezpečení přenosu dat mezi aplikacemi se stará sama technologie Windows Communication Foundation, ve které při vývoji bylo vyzkoušeno několik možností tzv. bindování. V aktuální chvíli aplikace využívají bindování WSHttpBinding, které poskytuje vyšší bezpečnost než bindování BasicHttpBinding. Bezpečnost by se dala ještě dále zvýšit, ale bylo by nutné vytvořit zabezpečovací certifikáty CA. Při vývoji je možné vytvořit testovací certifikát, ale při skutečném nasazení ve firmě by bylo nutné získat certifikát ověřený certifikační autoritou. Z tohoto důvodu tato metoda zabezpečení nebyla prozatím využita, ale v případě potřeby by přechod netrval příliš dlouho. Pokud by vývoj této aplikace pokračoval na mobilní a webové platformy, bylo by pravděpodobně nutné přejít na bindování BasicHttpBinding. Hlavním důvodem je, že toto bindování využívá pro odesílání zpráv SOAP 1.1. Zabezpečení je sice ve výchozím nastavení vypnuto, ale opět lze využít certifikát a změnit nastavení v BasicHttpSecurityMode. [16]

2.3 Vybrané funkce aplikace

V této části práce jsou představeny funkce, které jsou v aplikaci využity pro zajištění správného chodu. Tyto funkce běžný uživatel vůbec nevidí a velmi často si myslí, že takové funkce nikdy nevyužívá. Zejména se jedná o způsob uložení souborů na serveru, přenosy od klienta na server i opačně. Další důležité funkce jsou například získání ustáleného stavu, reakce na nové soubory na serveru, zjištění nových souborů v klientském počítači, smazání souboru a další.

2.3.1 Uložení souboru v úložišti serveru

Každý soubor, který je nahrán na server se uloží samozřejmě do diskového úložiště serveru. Uloží se do speciálního adresáře definovaného přímo v programu serverové strany aplikace. Dále do adresáře daného uživatele, který soubor vytvořil. Název tohoto souboru se vygeneruje pomocí funkce Guid, která je přímou součástí programovacího jazyka C#. Jedná se o náhodně generované 128bitové celé číslo. Tento název se spolu se skutečným názvem, relativní adresou u uživatele v synchronizované složce a několika dalšími informacemi uloží do databáze. Hlavním důvodem uložení všech informací o souborech v databázi je především rychlost vyhledávání a získávání informací podle zadaných kritérií. Kdyby při synchronizaci vždy musela aplikace získat metadata o souboru přímo z pevného disku, docházelo by k jeho zbytečnému zatěžování.

guid	fileName
0974eee3-d2b2-45ea-be64-76f30b5cffb6	ABackups.bin
306a7578-b041-4547-adf6-679e4c89fa7b	Asettings.bin
2e276a89-de19-404a-9da5-99e615ba8429	Async.bin

Obrázek 17 – Přiřazení jména souboru v úložišti

Zdroj: vlastní

2.3.2 Přenos souboru

Přenos souborů mezi klientskou aplikací a serverovou službou probíhá pomocí stream přenosu. Takže se dá říct, že se jedná o blokový přenos dat. Z tohoto důvodu bylo upraveno rozhraní služby. Veškerá ostatní komunikace probíhá pomocí tříd, které jsou označeny tzv. DataContracty. Třídy použité k přenosu souborů pomocí streamu jsou označeny tzv. MessageContracty.

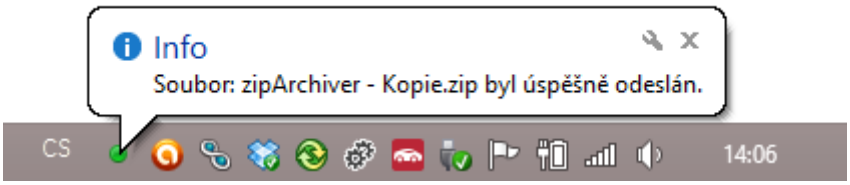
Klientská aplikace odesílá všechny soubory podle pořadí ve frontě souborů k odeslání. Do fronty jsou přidávány soubory k odeslání z mnoha dalších funkcí. Přímo se jedná o funkce sledování změn ve složce a ustálení stavu při startu aplikace. Odeslání probíhá formou streamu. Rozhraní služby předá stream serverové straně. Poté jsou zkontrolována metadata o souboru, zda je přenos nutný. Pokud ano, serverová strana spustí přenos ze zdrojového streamu od klienta do cílového streamu, který se stará o ukládání bloků dat přenášeného souboru přímo do úložiště dat na serveru.

Stažení souboru proběhne vždy pouze na žádost z klientské aplikace. Ke stažení souboru dojde pouze, pokud o něj požádá některá z funkcí klientské aplikace. Jedná se buď o funkci, která zajišťuje získání ustáleného stavu, nebo pokud pravidelné zjišťování nových souborů na serveru zjistí změněný nebo nový soubor na serveru.

2.3.3 System tray

System tray je speciální oblast plochy u systémových hodin. V této oblasti jsou zobrazovány některé běžící aplikace. Mezi těmito aplikacemi byla přidána i tato klientská aplikace.

Tato aplikace zobrazuje v oblasti system tray svou vlastní ikonu. Zobrazovány jsou celkem tři různé ikony. Každá z nich zobrazuje různý stav aktivity aplikace. První stav zobrazuje spuštěnou aplikaci, která není připojena k serveru. Tento stav nastává většinou, když počítač není připojen k síti nebo je server odpojen. Zobrazí se šedivá ikona. Dalším stavem je ikona červená. Ta zobrazuje aktivitu aplikace. Zejména pokud dochází k přenosu dat při synchronizaci nebo při zálohování. Poslední definovaný stav vyjadřuje, že se aplikace nachází v ustáleném stavu. Všechna data jsou aktuální. Zobrazuje se zelená ikona.

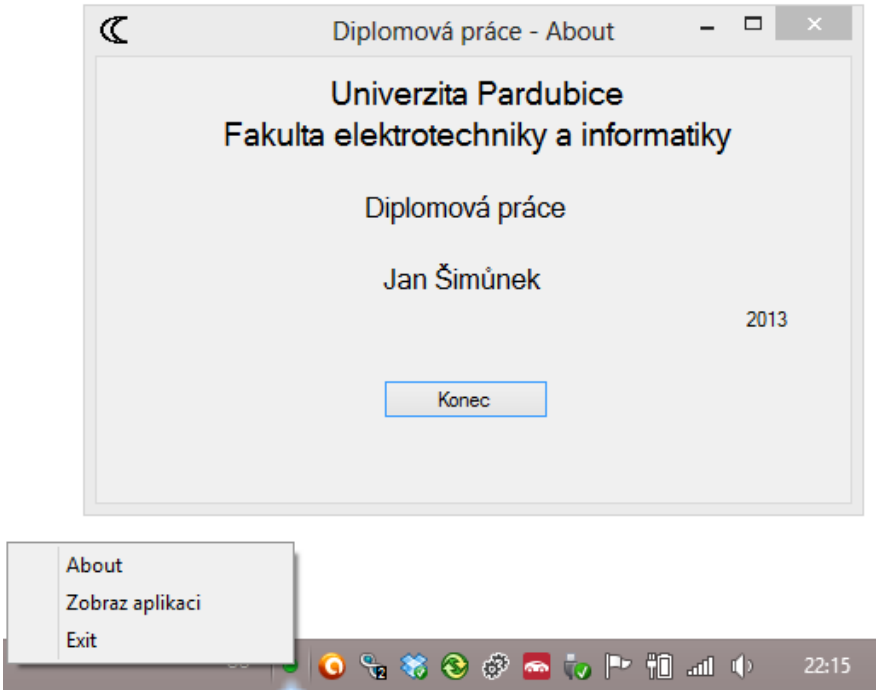


Obrázek 18 – Zobrazení v system tray

Zdroj: vlastní

Informace o všech aktivitách aplikace jsou zobrazovány ve velmi oblíbených tooltip bublinách. Ty se zobrazují nad system tray ikonou aplikace. Zobrazovány jsou informace o dokončení odeslání nebo přijetí souboru jak u synchronizace, tak u zálohování.

System tray ikona poskytuje také menu. V menu lze nalézt tlačítka pro zobrazení aplikace, když je minimalizována, ukončení celé aplikace a zobrazení informací o autorství a verzi této aplikace.



Obrázek 19 – Menu v system tray oblasti a formulář about

Zdroj: vlastní

2.3.4 Automatické přihlášení

Automatické přihlášení se stalo jednou z běžných funkcí, které využívá každý uživatel dnes a denně. Velmi často tuto funkci mnoho uživatelů přehlíží a považuje ji za samozřejmost. Tato funkce umožňuje přihlášení do klientské aplikace automaticky ihned po startu aplikace, což velmi urychlí práci každého uživatele. U této funkce nalezneme i poměrně velikou nevýhodu, protože přímo na klientově počítači musí být uloženy přihlašovací údaje. Aplikace má všechna svá místní data uložena ve složce temp v operačním systému. Data jsou uložena jako bytový soubor, aby nebyl přímo čitelný. Obsahuje uživatelské jméno a zašifrované heslo pomocí MD5. Šifrování MD5 zatím stále nikdo nedokázal zpětně dešifrovat, proto je tato funkce alespoň minimálně bezpečná. Automatické přihlášení se velmi vhodně doplňuje s funkcí automatického startu aplikace po spuštění operačního systému.

2.3.5 Smazání souboru

Nastane-li na klientském počítači smazání libovolného souboru, dojde na serveru také k fyzickému odstranění odpovídajícího souboru. Každý soubor je na serveru reprezentován příslušným záznamem v databázi. Při smazání souboru dojde v databázi pouze ke změně atributu isLive na hodnotu false. Což indikuje, že soubor již na serveru fyzicky neexistuje. Tento atribut se využívá z důvodu uchování dalších informací v databázi vztažených k tomuto souboru. Kdyby došlo k jeho odstranění, bylo by nutné odstranit i všechny ostatní záznamy o něm. Především důležité jsou záznamy o změně souboru, které jsou pravidelně stahovány do klientské aplikace.

2.3.6 Po spuštění aplikace

Při spuštění aplikace a úspěšném přihlášení musí dojít k synchronizaci, aby bylo dosaženo tzv. ustáleného stavu. Tím se myslí stav, kdy budou na serveru i u klienta aktuální soubory a složky. Toho je dosaženo pomocí několika funkcí. Nejprve se ze serveru získají synchronizované složky, které se porovnají s lokálně uloženými složkami. Nové složky budou přidány do výběru a chybějící odstraněny z lokálního úložiště. V tomto uložišti jsou uloženy informace o synchronizovaných složkách a jejich umístění v místním počítači.

Dalším krokem je získání seznamu všech souborů uložených na serveru. Jedná se však pouze o metadata o souborech. Tyto metadata jsou uložena v databázi serveru. Hlavním důvodem uložení dat o souborech v databázi je rychlost získání těchto dat. Pokud by bylo nutné všechna data získat přímo z fyzického úložiště dat na serveru, docházelo by často k velmi pomalým odezvám aplikace, případně by mohlo dojít i k zaseknutí systému z důvodu přetížení pevných disků. Metadata jsou porovnána se soubory uloženými na lokálním počítači. O nové nebo pouze novější soubory si aplikace požádá ke stažení. Pokud jsou nové nebo novější soubory na místním počítači dojde později k jejich odeslání na server přidáním do fronty souborů k odeslání. Také se na serveru uloží informace o dané změně souboru, aby mohla být změna provedena i na dalších počítačích, které tento uživatel používá. Po dokončení synchronizace získáme ustálený stav a další synchronizace probíhá pouze na základě změn souborů v místním počítači nebo zjištění změn na serveru.

2.3.7 Reakce na lokální změnu souboru

Mezi hlavní požadavky aplikace patří reakce na změnu souboru ve složce. Nabízejí se dva způsoby zjišťování změn. První se týká pravidelného průchodu celého adresáře a porovnávání lokálních souborů s metadaty souborů uloženými v databázi serveru. Tato metoda není vhodná z důvodu náročnosti, protože je nutný přímý přístup k úložišti souborů a musí se zkontrolovat každý jednotlivý soubor celé stromové struktury ve sledované složce. Mnohem vhodnější způsob je neustálé sledování vybraného adresáře. Tato metoda sice více využívá operační paměť a procesor než předchozí způsob, ale ve výsledku mnohem méně zatíží celý systém než prohledávání adresáře přímo v datovém úložišti.

Ke správné funkčnosti tohoto požadavku byla využita již existující třída přímo z programovacího jazyka C#. Jedná se o třídu FileSystemWatcher. Instance této třídy poskytuje plnou funkčnost potřebnou pro tuto aplikaci. Slouží ke sledování změn v zadaném adresáři. Lze sledovat změny souborů a podadresářů vybrané složky. Na každou změnu reaguje příslušná událost, která poté využije některou další funkci aplikace. Například při vytvoření nového souboru ve sledovaném adresáři dojde k vyvolání události onCreate() a ta přidá daný soubor do fronty souborů k odeslání. Nastane-li ve sledovaném adresáři přejmenování souboru, poskytne instance třídy název nového souboru, ale i původní název před změnou. [17]

2.3.8 Reakce na změnu souboru na serveru

Zjištění změny souboru na serveru zajišťuje klientská aplikace pravidelným dotazováním se na server a získávání případných změn. Interval dotazování je nastaven na deset vteřin, ale s největší pravděpodobností by byl po dlouhodobém testování aplikace prodloužen, aby nedocházelo ke zbytečnému síťovému provozu. Další možností řešení tohoto problému by byla další možnost v lokálním nastavení, kde by si každý uživatel mohl nastavit interval zjišťování nových souborů na serveru. Hlavní výhodou této možnosti by byla alespoň minimální úspora přenesených dat. Tato úspora by hrála roli zejména u přenosných počítačů využívající mobilní datové přenosy, které jsou v dnešní době velmi oblíbené, ale stále ještě limitované kapacitou přenesených dat za měsíc.

Dotazem na server získá aplikace seznam změn souborů od posledních synchronizace. Poté porovná jednotlivé soubory a případně požádá o stažení daného souboru ze serveru do lokálního počítače. Může také nastat stav, kdy na serveru byl soubor změněn, ale než byl přenesen do lokálního počítače, došlo k jeho změně také na lokálním počítači. Tento stav je označován za konflikt. Oba soubory jsou samozřejmě zachovány, aby uživatel nepřišel o žádná data. Jeden ze souborů se přejmenuje a poté ihned nastane další synchronizace.

2.3.9 Spuštění aplikace po spuštění operačního systému

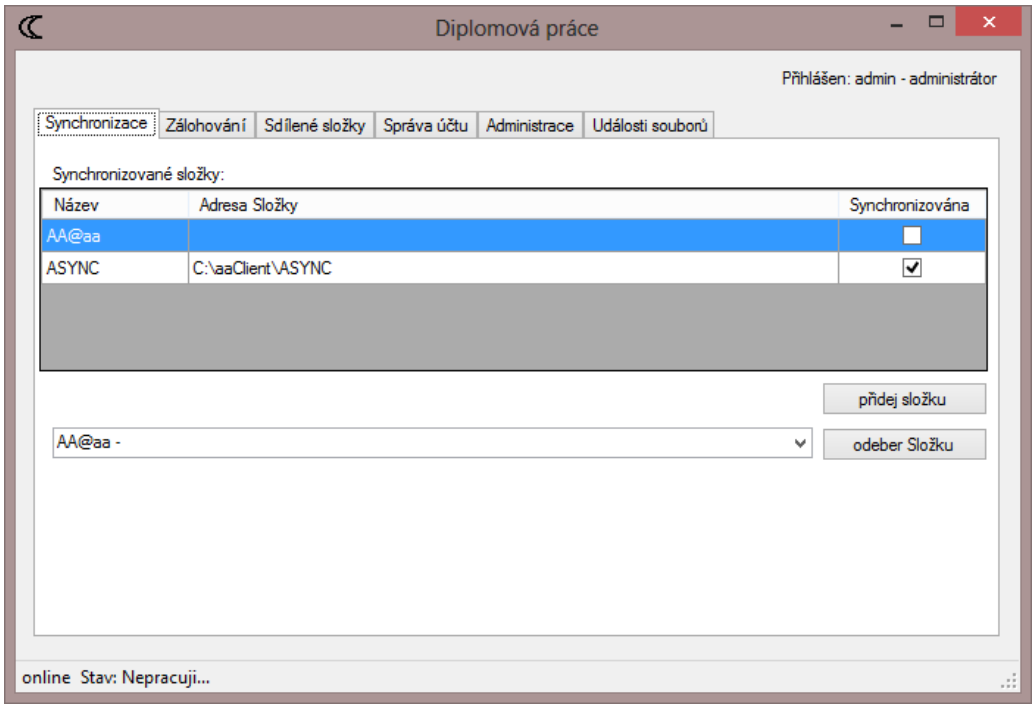
Do aplikace byla také přidána funkce, která využití této aplikace velmi usnadní. Aplikace je schopna se sama spustit ihned po spuštění operačního systému. V kombinaci s funkcí automatického přihlášení tedy uživatel nemusí udělat jediný klik a bude mít neustále aktuální data u sebe.

Tato funkce se integruje přímo do registrů operačního systému. Je možné ji vypnout v běžných místech, kde se zobrazují programy spouštěné po spuštění, ale také přímo v nastavení aplikace.

2.4 Představení aplikace

Původně navržený vzhled aplikace byl skutečně použit. Využití záložek velmi pomohlo k přehlednému zobrazení jednotlivých funkcí a jejich nastavení. V některých záložkách bylo dokonce nutné zobrazit další panel se záložkami, aby byly funkce, které spolu souvisí na jednom místě, a byla zachována přehlednost celé aplikace.

První záložka byla určena nastavení synchronizace složek. Druhá poskytuje funkce k zálohování. Zde byl využit další panel se záložkami, který obsahuje vytvoření jednoduché zálohy, vytvoření pravidelné zálohy, zobrazení pravidel pro pravidelné zálohování a zobrazení všech provedených záloh. Další záložka poskytuje uživateli možnost nastavení sdílených složek mezi uživateli. Následující dvě záložky jsou určeny pro správu a administraci aplikace. Poslední záložka zobrazí všechny události, které nastaly při synchronizaci dat.

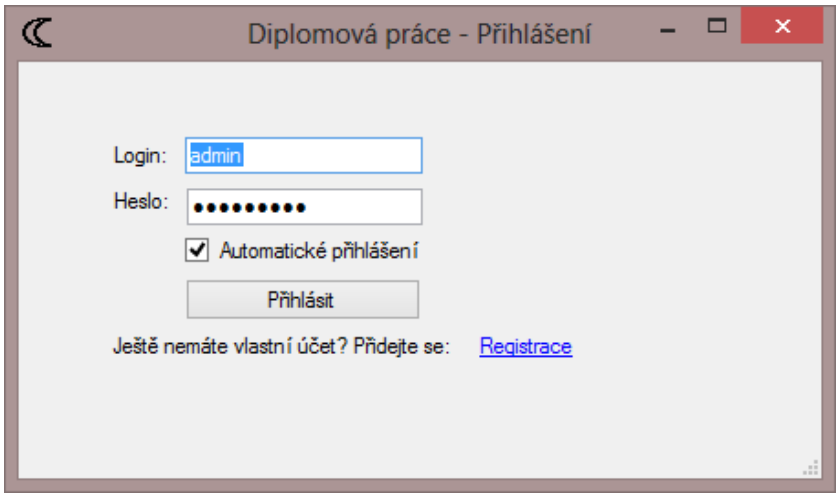


Obrázek 20 – Vytvořený layout aplikace

Zdroj: vlastní

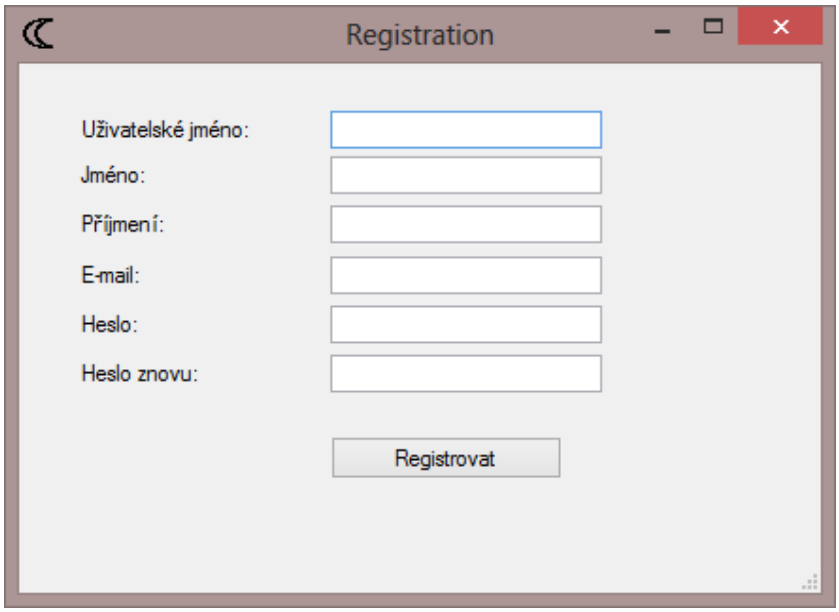
2.4.1 Registrace a přihlášení

Po spuštění klientské aplikace se zobrazí formulář k přihlášení. V něm je nutné vyplnit správné uživatelské jméno a heslo. Po kliknutí na tlačítko přihlásit dojde ke kontrole zadaných údajů. Nejprve se zkontroluje, zda zadané řetězce nejsou zcela prázdné. V opačném případě dojde k zobrazení chybového hlášení. Dále je možné zvolit automatické přihlašování. Funkce automatické přihlášení již byla představena v předchozí kapitole. Tato volba bude uložena na klientském počítači a je možné ji změnit v nastavení správy účtu. Zadané heslo se ihned zašifruje pomocí šifrování MD5. Poté se uživatelské jméno a heslo ověří v databázi na serveru, a pokud jsou údaje správné, dojde k přihlášení a zavření přihlašovacího formuláře.



Obrázek 21 – Formulář pro přihlášení do aplikace

Zdroj: vlastní



Obrázek 22 – Registrační formulář

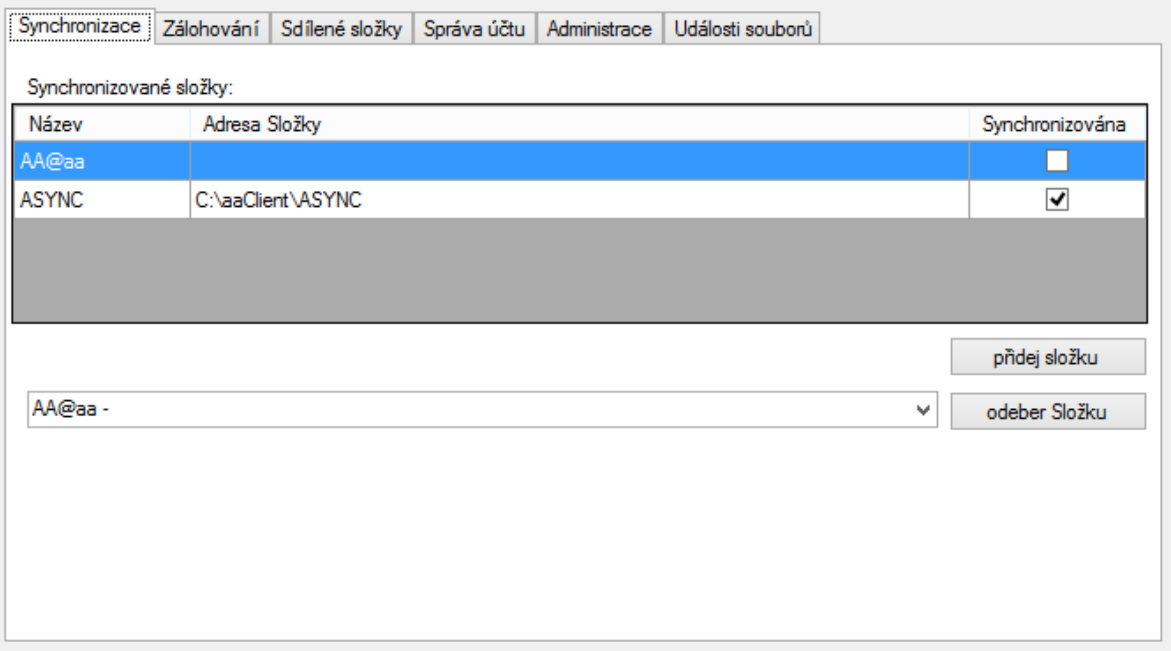
Zdroj: vlastní

Pokud uživatel ještě nemá vytvořený účet, může zvolit volbu registrace. Poté dojde k zobrazení dalšího formuláře. Tento formulář slouží k vytvoření nového uživatele a poté provede uložení v databázi na serveru. Obsahuje všechny informace, které je nutné o uživateli zadat. Jedná se o uživatelské jméno, jméno a příjmení, e-mail a heslo. Po vyplnění potřebných údajů a stisknutí tlačítka registrovat dojde ke kontrole zadaných dat. Všechna pole jsou zkontrolována, aby nebyla prázdná. E-mail je zkontrolován pomocí validace jeho tvaru. Hesla musí být shodná a jejich délka musí být větší než osm znaků. Heslo se okamžitě zašifruje pomocí šifrování MD5. Další kontrola nastane na serverové straně aplikace. Jedná se o zjištění, jestli zadané uživatelské jméno již neexistuje. Pokud celá kontrola proběhne v pořádku, dojde k přidání uživatele do databáze na serveru a poté se již uživatel může pomocí přihlašovacích údajů přihlásit.

2.4.2 Synchronizace

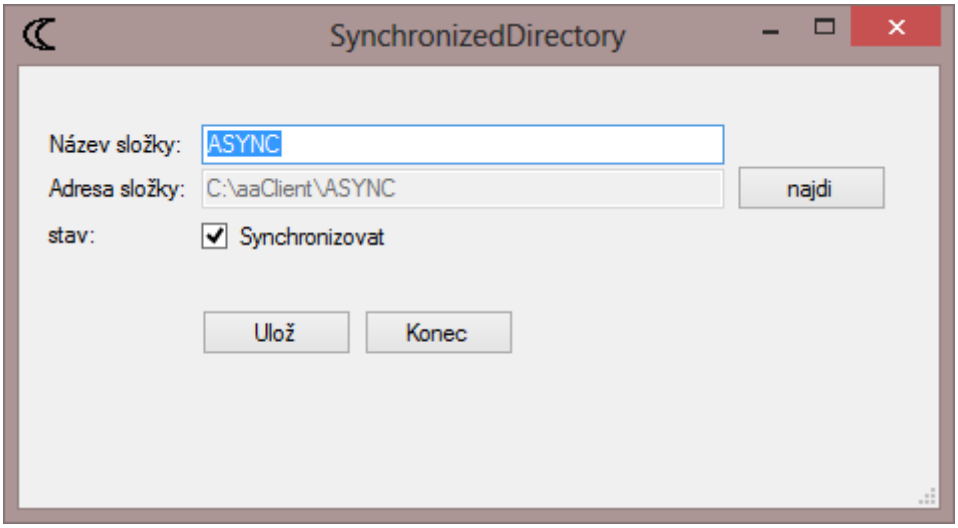
Synchronizace libovolných složek

Uživatel může synchronizovat libovolné složky v počítači. Názvy těchto složek jsou uloženy v databázi na serveru z důvodu možnosti synchronizace těchto složek v jiných počítačích nebo sdílení mezi dalšími uživateli. Každý soubor v těchto složkách bude synchronizován se serverem.



Obrázek 23 – Nastavení synchronizovaných adresářů

Zdroj: vlastní



Obrázek 24 – Změna synchronizované složky

Zdroj: vlastní

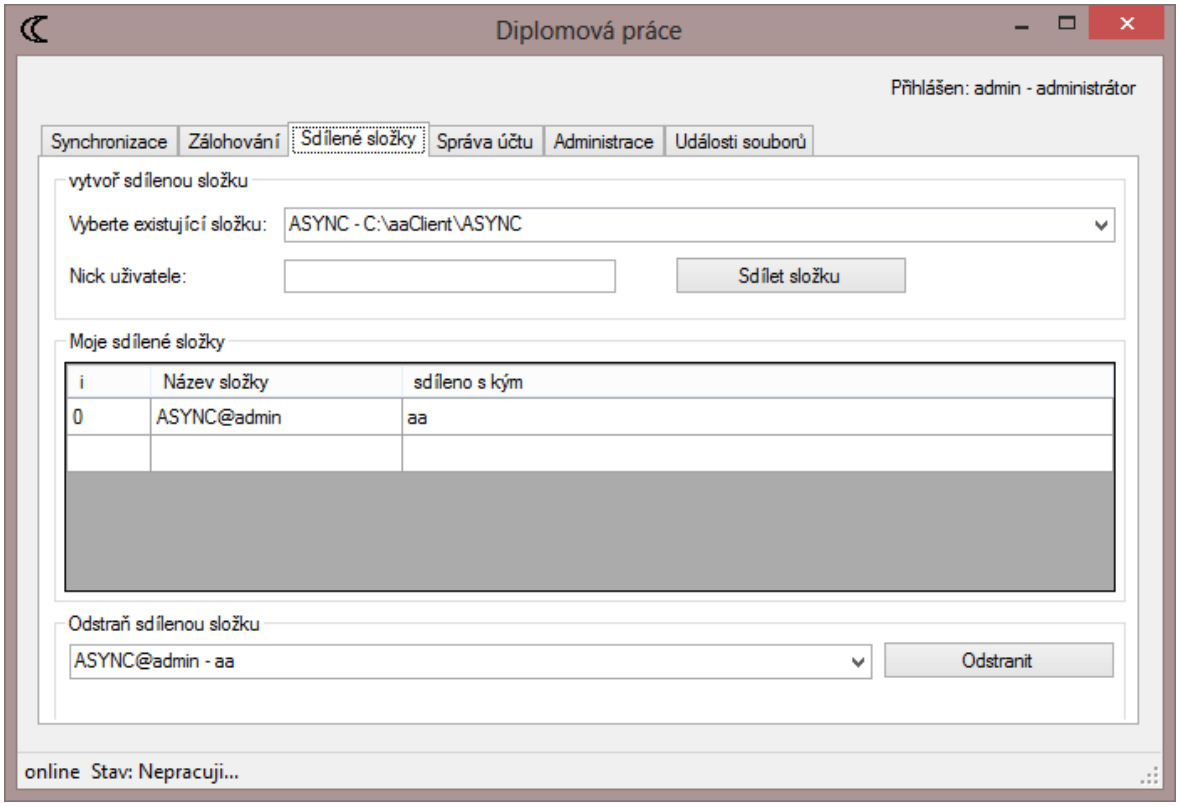
Všechny synchronizované adresáře jsou zobrazeny v aplikaci na záložce synchronizace. Pokud by uživatel chtěl nabízenou složku v tomto počítači synchronizovat, stačí na ni poklepat a zobrazí se detailní formulář dané složky. V něm je možné vybrat fyzickou složku v místním počítači, do které budou data synchronizována. Také zde může synchronizaci této složky v tomto počítači zrušit. Samozřejmě není nutné v každém počítači synchronizovat všechny složky.

Sdílené složky mezi uživateli

Jedním z hlavních požadavků na klientskou aplikaci byla možnost sdílet synchronizované složky mezi uživateli. Tuto funkci lze nalézt na záložce sdílené položky. Zde si uživatel může vybrat libovolnou ze svých synchronizovaných složek a sdílet ji s libovolným uživatelem. K tomu ovšem musí znát uživatelské jméno uživatele, se kterým chce složku sdílet. Název sdílené složky se vytvoří automaticky ze skutečného názvu synchronizované složky a uživatelského jména vlastníka sdílené složky. Ve spodní části obrazovky může uživatel vidět všechny své sdílené složky a také má možnost jejich odstranění.

Složky, které jsou danému uživateli nabízeny ke sdílení, jsou přidány na první záložce přímo do složek k synchronizaci. Zde si uživatel může zvolit, zda sdílenou složku chce v tomto počítači synchronizovat nebo ne. Sdílenou složku pozná podle jejího názvu, který obsahuje název a uživatelské jméno vlastníka.

Veškerá funkcionalita je zcela stejná, jako kdyby složka nebyla sdílena s jiným uživatelem. Jedná se hlavně o reakce na změny v adresáři, příjem a odeslání souborů a další funkce.



Obrázek 25 – Nastavení sdílených složek mezi uživateli

Zdroj: vlastní

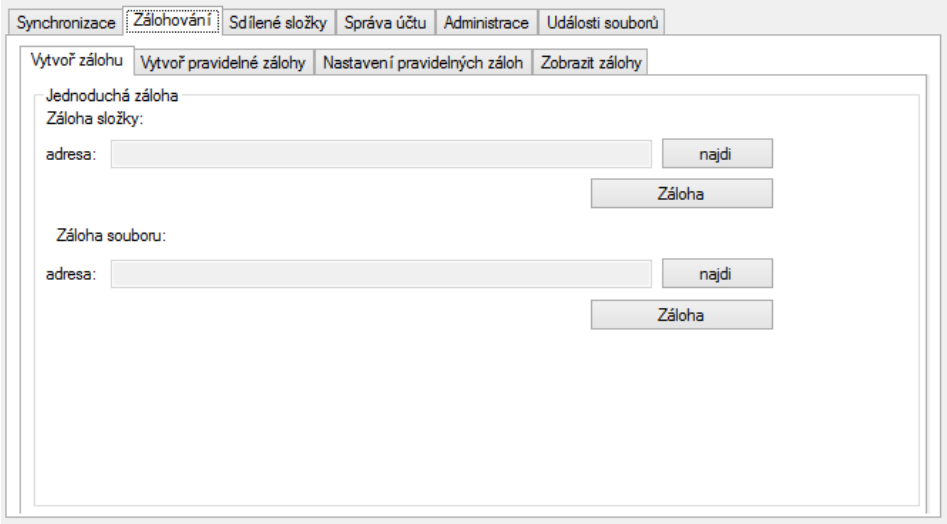
2.4.3 Zálohování

Data, která nejsou tolik důležitá, aby pro ně byla využita synchronizace, jsou také důležitá. Druhá část této aplikace se zabývá vytvářením záloh. Možnosti záloh byly rozděleny do několika částí. První je pouze jednoduchá záloha souboru nebo složky. Dále aplikace nabízí možnost nastavení pravidelného zálohování souboru či složky. Nakonec aplikace poskytuje zálohované soubory procházet a samozřejmě umožnit jejich stažení.

Zálohování souboru proběhne pouze jako odeslání do specializované složky na serveru. Při záloze celé složky nejprve dojde k archivování dané složky do souboru ve formátu *.zip. Tento archiv se dočasně uloží v adresáři temp lokálního operačního systému. Poté se teprve tento soubor odešle na server.

Jednotlivá záloha

Každý uživatel jednou za čas potřebuje provést zálohu pouze vybraného souboru nebo celé složky. Tuto možnost aplikace nabízí s velmi jednoduchým použitím. Stačí na záložce vytvoř zálohu vybrat složku nebo soubor, který uživatel chce zálohovat a kliknout na tlačítko zálohovat. Pokud chce uživatel zálohovat celý adresář, dojde nejprve k zabalení složky do archivu.



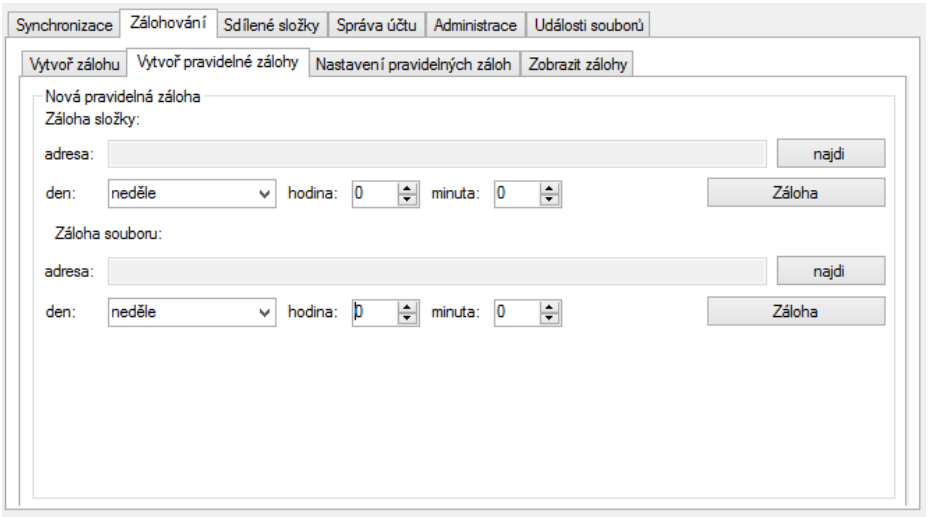
Obrázek 26 – Vytvoření zálohy

Zdroj: vlastní

Pravidelné zálohování

Pravidelná záloha probíhá vybraný den a ve vybraném čase. Záloha proběhne pravidelně každých sedm dní. Tento interval byl zvolen z důvodu nasazení této aplikace. Aplikace je přímo určena do menších podniků a zálohování by mělo mít primární úlohu archivovat nějaká data každý týden. Ideálně na konci pracovního týdne.

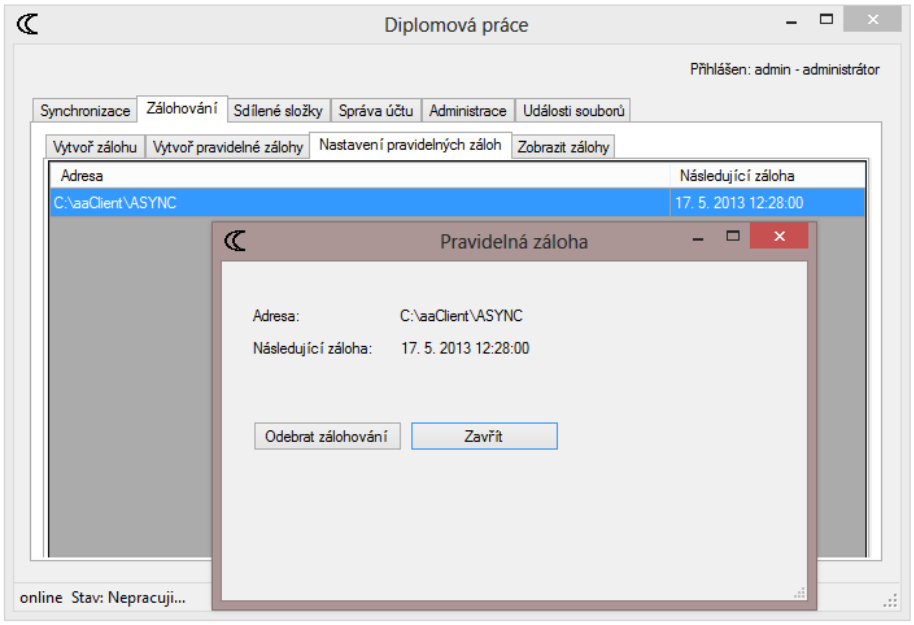
Vytvoření pravidelné zálohy zvládne i zcela běžný uživatel. Pouze vybere soubor či složku, den v týdnu a čas. Kliknutím na tlačítko dojde k přidání a lokálnímu uložení pravidelných záloh. Tato data jsou také uložena v adresáři operačního systému Windows. Zálohy probíhají pravidelně podle plánovaného času, který se kontroluje v pravidelných intervalech. Pokud by aplikace nebyla spuštěna ve chvíli zálohy, dojde k záloze po spuštění aplikace. Poté také dojde ke změně data, kdy má nastat další záloha.



Obrázek 27 – Vytvoření nastavení pro pravidelné zálohování

Zdroj: vlastní

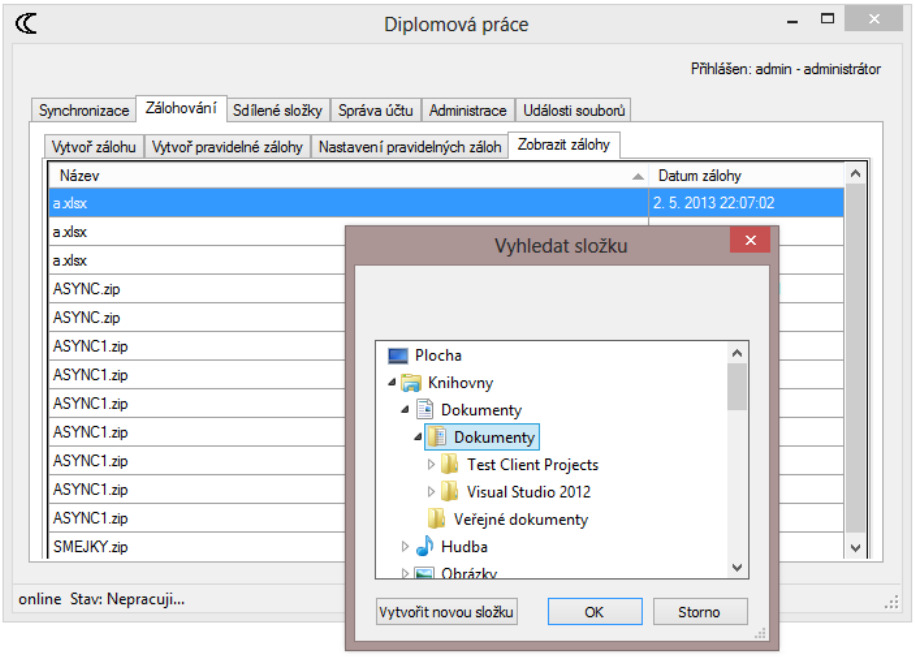
Všechna pravidla pro pravidelné zálohování jsou zobrazena na další záložce. Zde jsou zobrazeny všechny pravidelné zálohy a jejich následující datum a čas pravidelné zálohy. Každé pravidlo pro zálohování po dvojitém kliku zobrazí detailní informace o pravidle. Také formulář detailních informací umožňuje odebrat zálohovací pravidlo.



Obrázek 28 – Detailní formulář pravidelného zálohování

Zdroj: vlastní

Zobrazení záloh



Obrázek 29 – Zobrazení záloh a možnost stažení do lokálního počítače

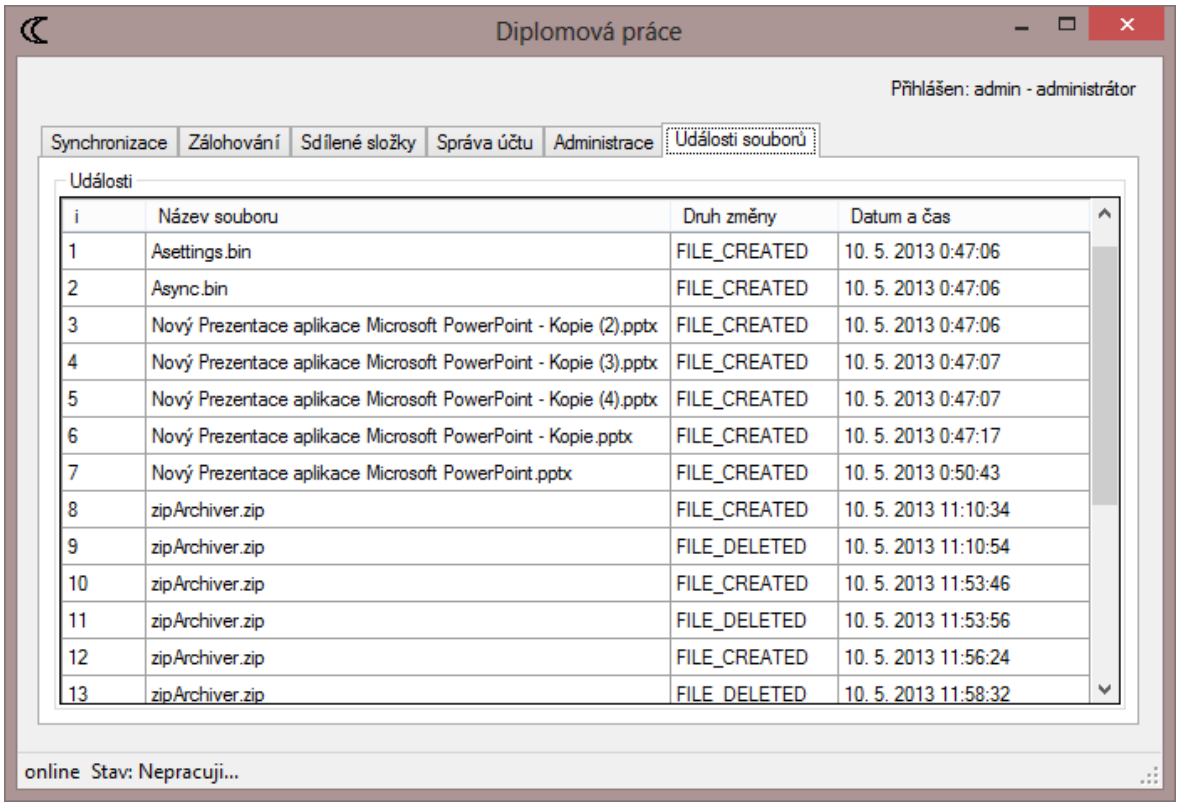
Zdroj: vlastní

Poslední záložka umožňuje procházet veškeré provedené zálohy a jejich verze. Zobrazena jsou i data provedení těchto záloh. Pokud uživatel chce některou ze záloh stáhnout zpět do lokálního počítače, stačí dvojitý klik na danou zálohu. Poté si uživatel vybere adresář, do kterého chce soubor uložit.

2.4.4 Změny souborů

Všechny změny souborů, které nastaly, jsou zobrazeny v této části aplikace. Jedná se o reakce na změny v synchronizovaném adresáři. Každá změna se definuje typem změny, názvem souboru a časem, kdy změna nastala. Typ změny určuje, co se s daným souborem stalo. Typ změny může být: vytvoření souboru, změna souboru, přejmenování souboru nebo smazání souboru.

Tyto změny jsou uloženy v databázi serveru a využívají se pro další funkce aplikace. Hlavní využití má funkce reakce na změnu na serveru, která se pravidelně dotazuje, zda nastala nějaká změna od poslední synchronizace.



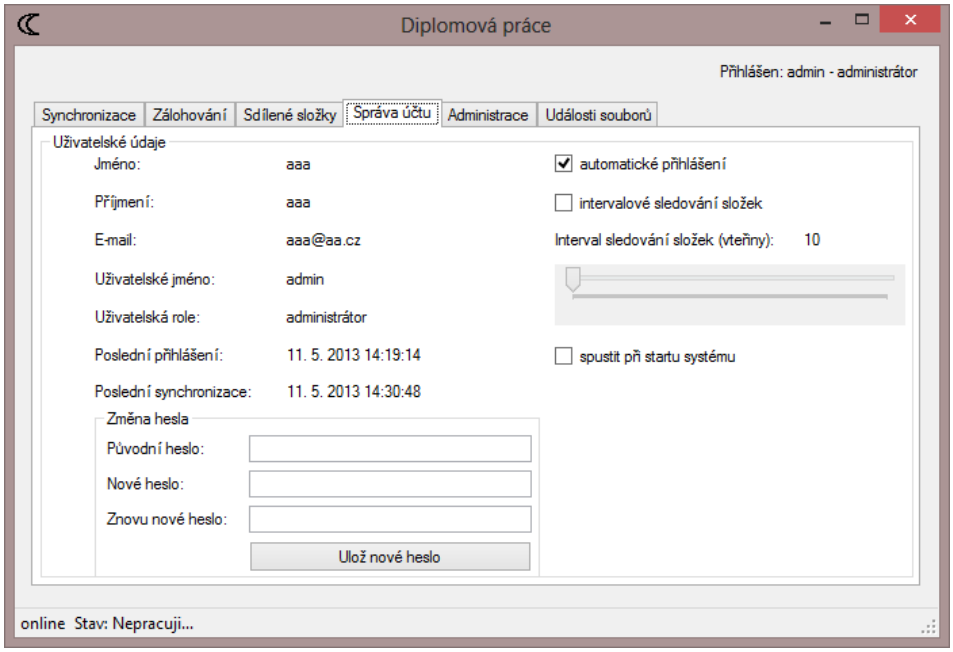
Obrázek 30 – Změny souborů při synchronizaci

Zdroj: vlastní

2.4.5 Administrace

Administrace v klientské aplikace se rozděluje na dvě části. První poskytuje informace o aktuálně přihlášeném uživateli a není nutné mít vyšší oprávnění k zobrazení. Druhá část administrace poskytuje informace o všech registrovaných uživatelích. K zobrazení těchto informací musí mít přihlášený uživatel speciální oprávnění.

Záložka správa účtu poskytuje všechny informace o přihlášeném uživateli, který je aktuálně přihlášen. Dále nabízí některá nastavení, která aplikace poskytuje a umožňuje změnit uživatelovo heslo.



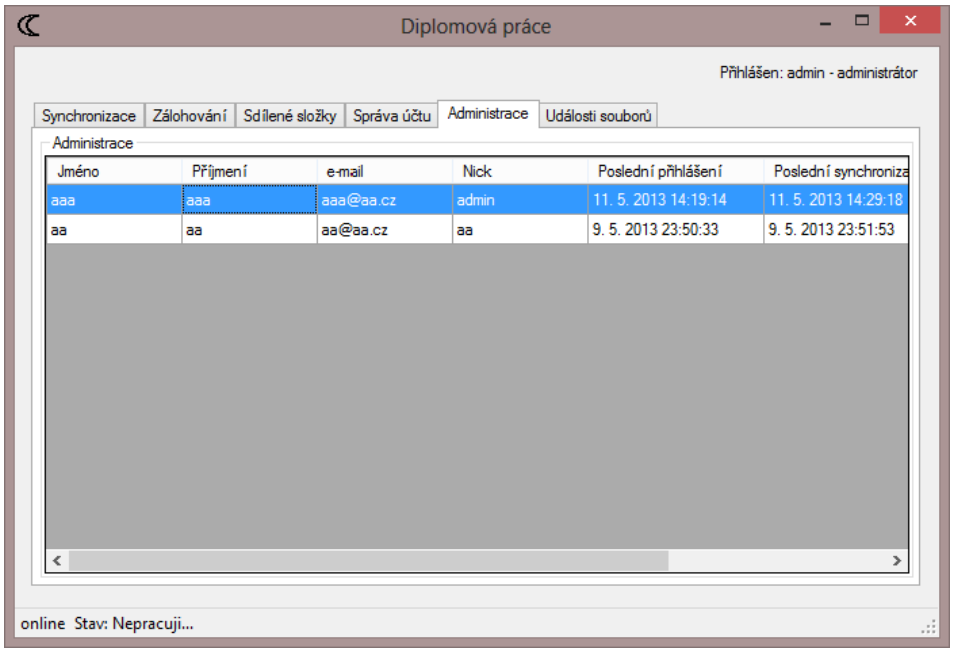
Obrázek 31 – Nastavení účtu přihlášeného uživatele

Zdroj: vlastní

Uživatel si může změnit heslo vyplněním vstupních polí na tomto formuláři. Nejprve musí být zadáno aktuální heslo. Poté dojde k jeho ověření. V případě správného hesla dojde ke kontrole nového hesla. Heslo musí splňovat stejné parametry jako při registraci nového uživatele. Délka hesla musí být minimálně osm znaků a nové heslo se musí shodovat s heslem uvedeným v políčku pro ověření správnosti zadaného hesla. V případě splnění všech požadovaných podmínek dojde k zašifrování hesla pomocí MD5 a jeho změně v databázi. Pokud nastane libovolná chyba, bude o ní uživatel informován chybovým hlášením.

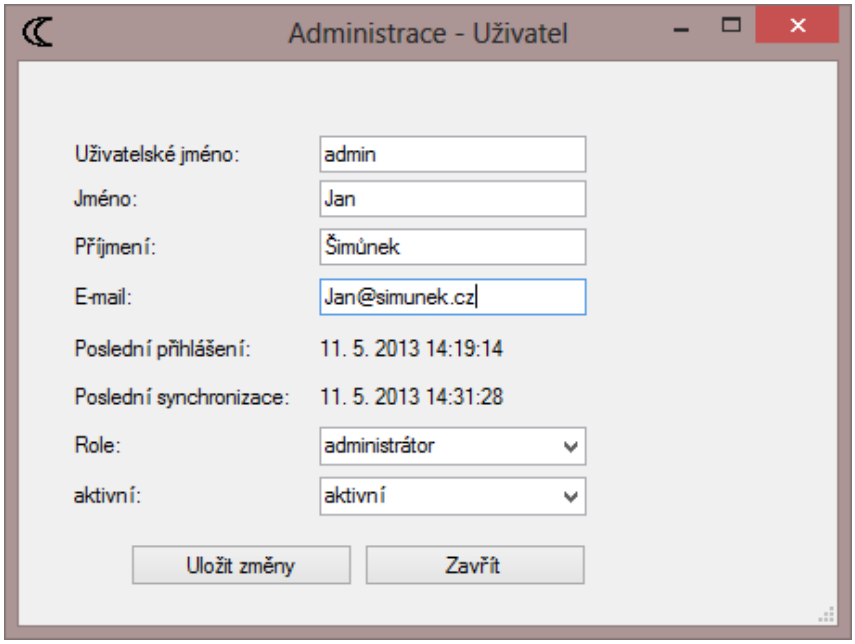
Dále zde aplikace umožňuje změnit některá nastavení. Konkrétně se jedná o automatické přihlášení, interval kontroly změny v synchronizovaných složkách a možnost spuštění aplikace ihned po spuštění operačního systému. Tato nastavení jsou uložena v místním adresáři temp operačního systému a uživatel má možnost je kdykoliv změnit.

Více možností nastavení aplikace poskytuje ve formě správy uživatelů. Všichni uživatelé jsou zobrazeni na záložce Administrace. Do této sekce klientské aplikace musí mít uživatel speciální oprávnění. Musí mít uživatelskou roli administrátor. V případě, že uživatel potřebná práva nemá, bude při vstupu na tuto záložku informován chybovým hlášením.



Obrázek 32 – Zobrazení uživatelů v administraci

Zdroj: vlastní



Obrázek 33 – Změna uživatelských údajů v administraci

Zdroj: vlastní

Má-li uživatel potřebná práva, může procházet informace o všech uživatelích. Dvojitým klikem na vybraného uživatele se zobrazí další formulář, který poskytuje detailní informace o uživateli. Tyto informace je umožněno změnit. Mohou být změněny informace o uživateli získané při registraci, práva nebo oprávnění přístupu do aplikace. Toto oprávnění dokáže zamezit uživateli přihlášení do aplikace. Při změně údajů dojde k uložení těchto informací v databázi na serveru.

2.4.6 Vylepšení aplikace

V aplikaci se nachází velké množství funkcí, které nebyly v původním zadání. Mnoho dalších funkcí však dosud nebylo implementováno. Často bychom mohli říci, že se jedná o maličkosti, ale ve výsledku každá taková maličkost dokáže velmi ušetřit člověku práci i čas, který tráví zbytečným klikáním na myš.

Implementována již například byla funkce automatického přihlašování, která pokud si to uživatel přeje, při každém spuštění aplikace rovnou daného uživatele přihlásí. K této funkci se přímo nabízela funkce, která aplikaci spustí ihned po spuštění operačního systému. Logování chyb, které nastanou na serverové straně. Tyto chyby jsou ukládány do textového souboru na serveru. V další verzi aplikace by mohly být tyto chyby zobrazeny. Samozřejmě pouze uživateli s potřebnými právy.

Mezi plánované funkce, které prozatím nebyly vytvořeny, patří automatické zálohování souborů při jejich jakékoliv změně. Tato záloha by tvořila jednotlivé verze souborů, které se mění při synchronizaci. Další velmi zajímavou funkcí by mohlo být připojení k některému cloudovému úložišti pomocí API. Tato funkce by potřebovala velmi zvážit, protože toto připojení z firemní aplikace by znamenalo pravidelné výdaje. Sice třeba ne ve výši částek, které jsou uvedeny v analýze konkurenčních aplikací, ale přesto by se o nějaké částky jednalo. Dalším problémem by bylo snížení bezpečnosti, protože by data opouštěla prostředí společnosti. Poměrně jednoduchá funkce, která by měla být později doplněna, je funkce, která po registraci zašle na e-mail, který byl zadán při registraci, informace o nově registrovaném uživateli. Souběžně s tím by mohly být stejné údaje zaslány správci sítě.

Dále by se dalo uvažovat nad webovou a mobilní aplikací. Velmi by záleželo na společnosti, zda by takovou funkci dokázala vůbec využít. Zda by se nestala pouze bezpečnostní hrozbou. Využití by měla v podnicích, kde zaměstnanci často cestují a potřebují mít neustále u sebe aktuální data. Ale ve většině běžných firem by přínos mobilní a webové aplikace měl nulový přínos.

Závěr

Tato práce se zabývala poměrně rozsáhlým tématem o synchronizaci a zálohování dat. Cílem práce bylo navrhnout a implementovat funkční aplikaci, která bude splňovat zadané požadavky synchronizace a zálohování.

V celé práci bylo několikrát zmíněno, kde by bylo vhodné vytvořenou aplikaci využít. Jedná se hlavně o menší firmy. Při návrhu i vývoji byl kladen důraz na cenu, rychlost a bezpečnost celé aplikace. I když některé tyto vlastnosti určí až přímé nasazení do konkrétního prostředí, přesto by náklady měly být nižší než konkurenční řešení.

V první části této práce byly uvedeny základní informace o dané problematice, vysvětleny základní pojmy a použité technologie k vývoji aplikace. Tyto technologie bylo nutné rozepsat, aby byl čtenář alespoň minimálně seznámen s funkčností aplikace. Poté byla provedena velmi rozsáhlá analýza konkurenčních řešení. Provedeno bylo také porovnání vlastní aplikace s těmito konkurenčními řešeními a navržena možná řešení všech nedostatků, které vlastní aplikace má v porovnání s konkurencí.

Druhá část podrobně popisuje nejprve jednotlivé funkce, které jsou v aplikaci implementovány. Jedná se o funkce, které jsou nutné ke správnému běhu programu. Většinu z nich uživatel vůbec nezaregistruje, protože běží na pozadí aplikace. Poté jsou vypsány jednotlivé funkce, které aplikace poskytuje uživateli. Nejčastěji se jedná o různá nastavení týkající se synchronizace, zálohování nebo uživatelského účtu. Dále je v aplikaci vytvořené jednoduché rozhraní ke správě uživatelů. Samozřejmě k této funkci je nutné mít vyšší oprávnění.

Při vývoji bylo nutné seznámení s technologií WCF, která poskytuje celou komunikaci mezi aplikacemi. Jedná se o obrovskou technologii, z které byla ve výsledné práci použita jen velmi malá část. Přestože tato technologie není úplně nejnovější, stále ke složitějším funkcím neexistuje mnoho článků nebo knížek. V češtině lze nalézt jen velmi málo informací, a pokud existují, tak obsahují pouze základní funkce. Zbytek informací existuje pouze v angličtině. Dále byly při vývoji použity návrhové vzory a některé datové struktury, které bylo nutné nejprve vytvořit. Ve výsledné práci bylo použito jen malé procento zdrojových kódů, které byly během celého vývoje vytvořeny.

Aplikace byla pravidelně testována. Správná funkčnost aplikace byl nejdůležitější aspekt při vývoji. Neustále probíhá optimalizace jednotlivých algoritmů, které jsou v aplikaci vytvořeny.

Všechny požadavky, které byly na počátku této práce požadovány, byly splněny. Práce poskytla plně funkční řešení vhodné pro synchronizaci a zálohování dat. Ideální použití by bylo uvnitř firmy, ale její využití je možné téměř kdekoliv, kde není potřeba cloudového úložiště. Cíle této práce byly splněny a dokonce přidány některé další funkce.

Literatura

1. **Microsoft.** Visual C#. *Microsoft Developer Network.* [Online] 2013. [Citace: 12. Květen 2013.] http://msdn.microsoft.com/en-us/library/vstudio/kx37x362.aspx.

2. **Laryš, Kryštof.** Windows Communication Foundation. *NetStudent.c.* [Online] 2. Únor 2009. [Citace: 12. Květen 2013.] http://netstudent.cz/Articles/?tag=windows-communication-foundation.

3. **Salvator, Paolo.** Paolo Salvatori's Blog. *MSDN Blogs.* [Online] 19. Listopad 2009. [Citace: 12. Květen 2013.] http://blogs.msdn.com/b/paolos/archive/2009/11/17/customizing-and-extending-the-biztalk-wcf-adapters.aspx.

4. **Löwy, Juval.** *Programming WCF Services.* Sebastopol : O'Reilly Media, Inc., 2010. 978-0-596-80548-7.

5. **Košťál, Marián.** WCF - Bindings. *Vyvojar.cz.* [Online] 24. Únor 2007. [Citace: 12. Květen 2013.] http://www.vyvojar.cz/Articles/459-wcf-bindings.aspx.

6. **Šípal, Štěpán.** Materiály pro studenty IVT. *Výuka IVT Gymnázium Čakovice.* [Online] 26. Březen 2009. [Citace: 12. Květen 2013.] http://vyuka.greendot.cz/materialy/material-4.pdf.

7. **Tvrz, Stanislav.** db4o aneb objektové databáze v .NET. *Databázový svět.* [Online] 25. Červenec 2005. [Citace: 12. Květen 2013.] http://www.dbsvet.cz/view.php?cisloclanku=2005072503.

8. **Oracle Corporation.** *MySQL :: The world's most popular open source database.* [Online] 2013. [Citace: 12. Květen 2013.] http://www.mysql.com/.

9. **Horváth, Tomáš.** Teoretický úvod do relačních databází. *Programujte.com.* [Online] 8. Listopad 2007. [Citace: 12. Květen 2013.] http://programujte.com/clanek/2007110801-teoreticky-uvod-do-relacnich-databazi/.

10. **Dropbox.** Dropbox for Business. *Dropbox.* [Online] 2013. [Citace: 12. Květen 2013.] https://www.dropbox.com/business.

11. **Google Inc.** Služby – Google Apps pro firmy. *Google.* [Online] 2012. [Citace: 12. Květen 2013.] https://www.google.com/intl/cs/enterprise/apps/business/products.html#drive.

12. **Box.** Box for Business: What Can Box Do For Your Business? *Box.* [Online] 2013. [Citace: 12. Květen 2013.] https://www.box.com/business/.

13. **Bitto, Ondřej.** Cobian Backup: Ideální zálohování vašich dat. *JNP.cz.* [Online] 04. Prosinec 2012. [Citace: 12. Květen 2013.] http://jnp.zive.cz/cobian-backup-idealni-zalohovani-vasich-dat.

14. **MK SOLUTIONS s.r.o., FOX COMPUTERS s.r.o.** True Image 2013. *Acronis.* [Online] Zebra systems s.r.o., 2007. [Citace: 12. Květen 2013.] http://www.acronis.cz/domacnosti-a-kancelare/produkty/true-image-home/.

15. **Bernard, Borek.** Úvod do architektury MVC. *Zdroják.* [Online] 07. Květen 2009. [Citace: 12. Květen 2013.] http://www.zdrojak.cz/clanky/uvod-do-architektury-mvc/.

16. **Microsoft.** BasicHttpBinding – třída. *Microsoft Developer Network.* [Online] 2013. [Citace: 12. Květen 2013.] http://msdn.microsoft.com/cs-cz/library/system.servicemodel.basichttpbinding.aspx.

17. —. FileSystemWatcher – třída. *Microsoft Developer Network.* [Online] 2013. [Citace: 12. Květen 2013.] http://msdn.microsoft.com/cs-cz/library/system.io.filesystemwatcher.aspx.

18. **Sharp, John.** *Microsoft® Visual C# 2010 Step by Step.* Redmond : Microsoft Press, 2010. 2009939912.

19. **Pirkl, Josef.** *5ešené příklady v C# aneb C# skutečně prakticky.* České Budějovice : Kopp, 2005. 80-7232-265-6.

Příloha A – Obsah přiloženého CD

Přílohu tvoří instalační CD se zdrojovými kódy obou projektů, které tvoří celou aplikaci. Serverový projekt je nutné nainstalovat jako službu na serveru a poté je možné klientské aplikace rozšířit mezi uživatele. Dále musí být exitující databáze MySQL. Příkazy nutné k vytvoření tabulek a základnímu naplnění daty jsou také přiloženy na instalačním CD. Instalaci by měl provádět zkušený správce sítě v daném podniku.