VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

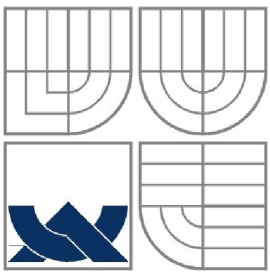
E-MAILOVÁ BRÁNA INFORMAČNÍHO SYSTÉMU JAKO WEBOVÁ SLUŽBA

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

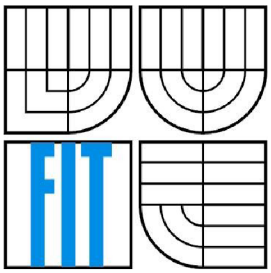
AUTOR PRÁCE
AUTHOR

MICHAL IMLAUF

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

E-MAILOVÁ BRÁNA INFORMAČNÍHO SYSTÉMU JAKO WEBOVÁ SLUŽBA

AN INFORMATION SYSTEM'S E-MAIL GATEWAY AS A WEB SERVICE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

MICHAL IMLAUF

VEDOUCÍ PRÁCE
SUPERVISOR

MGR. MAREK RYCHLÝ, PH.D.

BRNO 2010

Abstrakt

Cílem této bakalářské práce je implementovat webovou službu, která by umožňovala uživatelům realizovat oboustrannou komunikaci s jejich klienty pomocí e-mailových zpráv. Tato webová služba by mohla být využívána jako součást informačního systému. V této práci jsou vysvětleny principy webových služeb a e-mailové komunikace i s jejím zabezpečením.

Abstract

The goal of this bachelor thesis is to implement web service which allows to its users two way communication with their clients using e-mail. This web service could be used as part of information system. In this thesis are explained principles of web services and e-mail communication with its security.

Klíčová slova

C#, .NET, Webová služba, SOAP, E-mail, SMTP, E-mailová brána, Informační systém

Keywords

C#, .NET, Web service, SOAP, E-mail, SMTP, E-mail gate, Information System

Citace

Michal Imlauf: E-mailová brána informačního systému jako webová služba, bakalářská práce, Brno, FIT VUT v Brně, 2010

E-mailová brána informačního systému jako webová služba

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Mgr. Marka Rychlého, Ph.D.

.....
Michal Imlauf
19.5.2010

Poděkování

Tímto bych chtěl poděkovat vedoucímu této práce Mgr. Marku Rychlému, Ph.D. za poskytnutou pomoc při řešení této bakalářské práce.

© Michal Imlauf 2010
Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.

Obsah

1 Úvod

Cílem této práce je implementovat webovou službu informačního systému, která umožňuje uživatelům oboustrannou komunikaci s jejich adresáty pomocí e-mailových zpráv. Tato služba umožňuje tuto komunikaci tím, že uživatel zaregistruje filtr, který je porovnávám s příchozí odpovědí od adresáta na zaslaný e-mail a pokud je nalezena shoda, webová služba vykoná akci specifikovanou při vytváření filtru. Takováto webová služba by se uplatnila v informačním systému, ve kterém by se e-mail odeslal po registrace uživatele do systému, a následně by bylo potřeba automaticky zpracovat uživatelovu odpověď na daný e-mail.

Součástí této práce je i SMTP server, tuto část řešení nebudu realizovat sám, ale upravím server od jiného autora, jelikož implementace tohoto softwaru není součástí zadání a má sloužit pouze k demonstraci funkcionality mého řešení.

Pro implementaci této práce jsem si mohl vybrat libovolnou platformu a libovolný jazyk. Vybral jsem si platformu Windows, a jazyk C#. Celou práci hodlám realizovat za pomocí programu Microsoft Visual Studio 2008, který nabízí poměrně širokou škálu nástrojů a knihoven uzpůsobených pro vytváření webových služeb.

V druhé kapitole se věnuji všem technologiím a protokolům obsažených v této práci, tato část je důležitá pro pochopení funkcionality webových služeb a e-mailové komunikace. Kapitola třetí obsahuje návrh a analýzu mého řešení. V této části se snažím vysvětlit propojení všech jednotlivých částí, které budou dohromady tvořit řešení této práce. Ve čtvrté kapitole se věnuji samotné implementaci jednotlivých částí řešení. V páté kapitole se věnuji instalaci a správě webové služby. V předposlední tedy šesté kapitole popisuji možná rozšíření společně s návrhem na umístění této webové služby na jiné platformy než pouze na platformu Windows. V poslední části se pokusím shrnout dosavadní výsledky a budoucnost tohoto řešení.

2 Teoretická část

Cílem této části je objasnit principiální fungování všech aplikací a protokolů, které se týkají webových služeb a e-mailové komunikace a budou obsaženy v mojí práci.

2.1 Webová služba

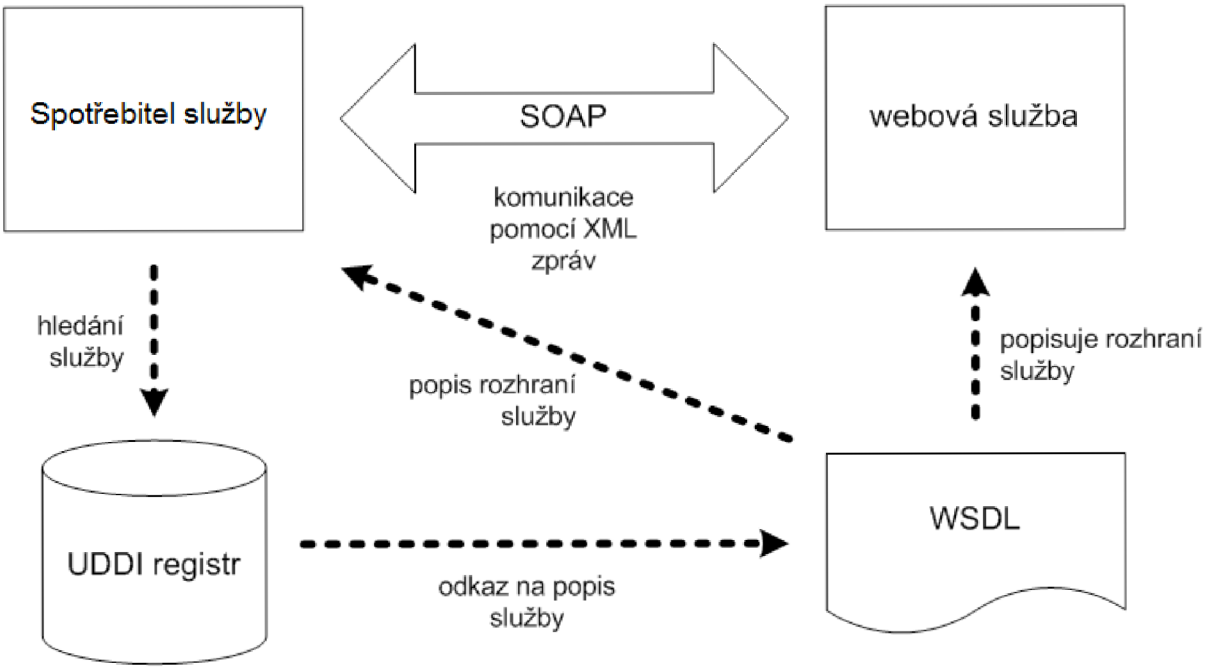
Webová služba funguje na principu síťové klient-server aplikace, v případě webových služeb na principu spotřebitel-poskytovatel. Hlavní výhodou webových služeb je fakt, že konzument i poskytovatel služby mohou běžet na různých platformách, dokonce je možné, aby byly napsány v různých jazycích, a i tak bude jejich komunikace bezproblémová.

Webová služba je softwarový systém navržený pro podporu interakce dvou počítačů. Její rozhraní je popsané v strojově zpracovatelném formátu (konkrétně WSDL). Ostatní systémy pracují s webovou službou způsobem popsaným v její definici, tedy pomocí zpráv SOAP, typicky přenášených pomocí protokolu HTTP v XML formátu ve spojení s ostatními webovými standardy [1].

To, že spolu konzument a poskytovatel mohou bezproblémově komunikovat, je způsobené tím, že zprávy se předávají pomocí známých a často používaných protokolů - SMTP a HTTP. Jelikož jsou to protokoly používané pro e-mail a pro přístup k webovým stránkách, nebývají jejich porty zablokované firewallem a tudíž mohou bez problému fungovat. Většina webových služeb funguje pod protokolem HTTP, jelikož je rozšířenější.

2.1.1 Jak webová služba funguje

I když se jedná o klient-server síťovou aplikaci je ve většině případů pro bezchybnou komunikaci mezi klientem a serverem ještě potřeba třetí strana. Tou je registr webový služeb neboli UDDI. Tímto registrem se budu podrobněji zabývat v kapitole 2.1.4. Celý tento proces funguje tak, že poskytovatel služby uloží do registru WSDL popis své webové služby. Spotřebitel následně si z registrů může tento popis stáhnout a na základě tohoto popisu vytvořit aplikaci, která bude s danou webovou službou komunikovat pomocí SOAP zpráv zapsaných ve formátu XML. Celý postup je názorně zobrazen na obrázku 2.1.1 .



Obrázek 2.1.2 - Popis architektury webových služeb, Převzato z [2].

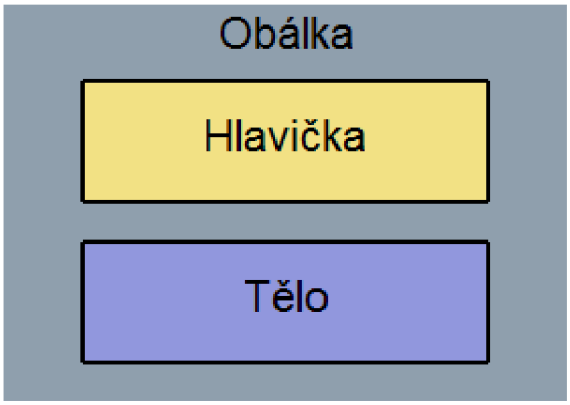
2.1.2 SOAP

V této kapitole budeme vycházet z [3], kde se nachází přehled tohoto protokolu. SOAP je zkratka pro simple object access protocol. SOAP je komunikační protokol, který popisuje syntaxi zpráv používaných ve webových službách.

Co je SOAP:
SOAP znamená jednoduchý protokol objektového přístupu
SOAP je komunikační protokol
SOAP byl vytvořen pro komunikaci mezi aplikacemi
SOAP je formát pro odesílání zpráv
SOAP je navržen pro komunikaci po internetu
SOAP je nezávislý na platformě
SOAP je nezávislý na jazyku
SOAP je založen na XML
SOAP je jednoduchý a rozšiřitelný
SOAP umožňuje obejít firewally
SOAP bude vyvíjen jako standard W3C

SOAP tedy umožňuje komunikaci i v prostředích s různými operačními systémy a jazyky jelikož používá pro zápis XML a pro přenos protokol HTTP, jenž je široce rozšířený. V základu funguje v konceptu jednosměrné zprávy, v praxi se ale využívá pro složitější modely komunikace.

Struktura SOAP zprávy



Obrázek 2.1.2 - Struktura SOAP zprávy. Převzato z [4].

- **Obálka (Envelope)** - Kořenový element XML dokumentu, který ji definuje jako SOAP zprávu. Do obálky je možno přidat jak deklaraci namespace tak i ostatní atributy.

- **Hlavička (Header)** - Pokud je ve zprávě přítomen, musí být umístěn ihned za obálku, většinou jsou v ní umístěny informace o tom, jak by měla být SOAP zpráva zpracována. Tato část zprávy není povinná.

- **Tělo (Body)** - Pokud zpráva neobsahuje hlavičku, tělo následuje okamžitě za obálkou. Obsahuje zprávu, kterou chceme předat.

2.1.3 WSDL

WSDL čili Web Service Description Language, volně přeloženo jazyk popisující webové služby. Informace v této kapitole jsou volně převzaty z [5]. Dokument WSDL je textový soubor zapsaný ve formátu XML obsahující umístění a popis funkcí dané webové služby, například vstupy a výstupy či názvy proměnných.

WSDL používá 4 hlavní elementy XML k popisu webové služby. Tyto elementy jsou uzavřeny do lomených závorek < >. Tyto elementy jsou vypsány v následujícím odstavci.

types - Definice datových typů, které jsou použity v dané webové službě.

messages - Popisuje data, která jsou vyměňovány mezi službou a uživatelem, mohou se skládat z více částí.

binding - Specifikuje, jak budou operace definované v <portType> přenášeny. Je možné specifikovat více možností přenosu pro jeden druh operace.

portType, Interfaces - Výčet operací prováděných službou. Definuje webovou službu, operace, které je možné provádět, a zprávy zasílané v rámci těchto operací.

Jsou podporovány 4 druhy operací :

1. Jednosměrné - klient zašle zprávu webové službě.
2. Požadavek-odpověď - zaslání zprávy webové službě a čekání na odpověď.
3. Vyžádání odpovědi - webová služba zašle žádost a čeká na odpověď od klienta.
4. Upozornění - webová služba zašle zprávu.

2.1.4 UDDI

Co je to UDDI (Universal Description, Discovery and Integration)

- UDDI znamená univerzální popis, objevení a integraci
- UDDI je místo pro uložení informací o webových službách
- UDDI je místo kde jsou popsána rozhraní jednotlivých webových služeb
- UDDI komunikuje pomocí protokolu SOAP
- UDDI je postavena na platformě Microsoft .NET

UDDI je tedy místo, které slouží firmám i jednotlivcům pro registraci jejich webových služeb, případně pro vyhledávání webových služeb, vhodných pro jejich záměry. Před UDDI zde nebyl žádný sjednocený systém pro tento typ služeb. V současné době se tohoto projektu účastí i giganti softwarové a hardwarové výroby. Například Dell, Fujitsu, HP, Hitachi, IBM, Intel, Microsoft, Oracle, SAP a Sun [6].

2.2 XML

V této kapitole se budeme věnovat XML a budeme čerpat ze zdroje [7]. XML je zkratka pro Extensible markup language, tedy rozšiřitelný značkovací jazyk. Patří mezi jazyky popisné, což znamená, že jeho syntaxe obsahuje popis toho co znamenají jednotlivé data v dokumentu. Tento jazyk je primárně určený pro uchování a výměnu dat. Hlavní výhodou jazyka XML je jeho otevřenost a snadná zpracovatelnost, tedy kdokoliv má přístup k jeho specifikaci a je možné ho bez problémů strojově zpracovat, případně i otevřít v textovém editoru a jednotlivé prvky upravit.

2.2.1 Syntaxe XML

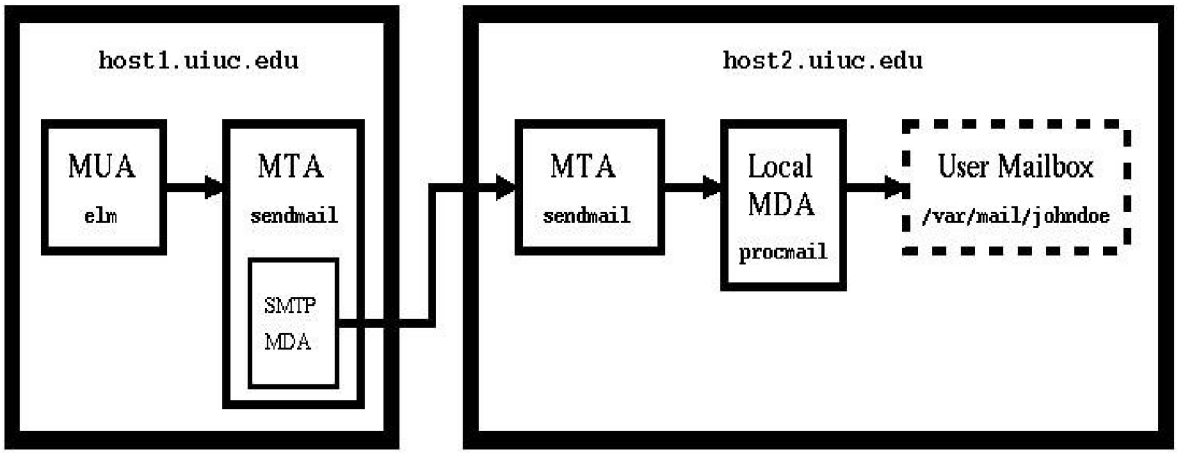
Dokument v XML je vždy textový s použitím znaků unicode. Další důležitý znak dokumentu je to, že má stromovou strukturu s právě jedním kořenem. Všechny prvky v dokumentu XML jsou definovány pomocí XML tagů, které jsou párové a case-sensitive. Párový v tomto případě znamená že je zde tag, který prvek uvozuje, a tag který ho ukončuje. Tagem začíná a končí každý jednotlivý prvek dokumentu. Následně si ukážeme strukturu XML dokumentu [7].

```
<kořen>
        <potomek>
         <<Zde je místo pro informace>>
        </potomek>
</kořen>
```

2.3 Protokol SMTP

Smyslem protokolu SMTP je spolehlivé a efektivní zasílání elektronické pošty. SMTP vyžaduje pouze spolehlivý datový přenos, a v drtivé většině případů je implementován pomocí protokolu TCP, i když je možné přenos implementovat i pod jiným protokolem. Protokol SMTP funguje pod portem 25.

2.3.1 Struktura SMTP



Obrázek 2.3.1 Model SMTP. Převzato z [8].

SMTP funguje takto [8]:

1. Uživatel pomocí MUA (mail user agent), například thunderbird, vytvoří a odešle e-mail přes SMTP klient (MTA - mail tranfer agent).
2. SMTP klient ověří jestli se adresát zprávy nenalézá na lokálním počítači, pokud ano tak jí předá lokálnímu MDA (mail delivery agent). Pokud ne tak pomocí DNS záznamu nalezne adresu MTA pro danou doménu a předá e-mail.
3. MTA na cílové doméně zkontroluje jestli je e-mail skutečně určený pro tuto doménu, pokud ano předá zprávu MDA.
4. MDA uloží zprávu do uživatelské schránky adresáta.

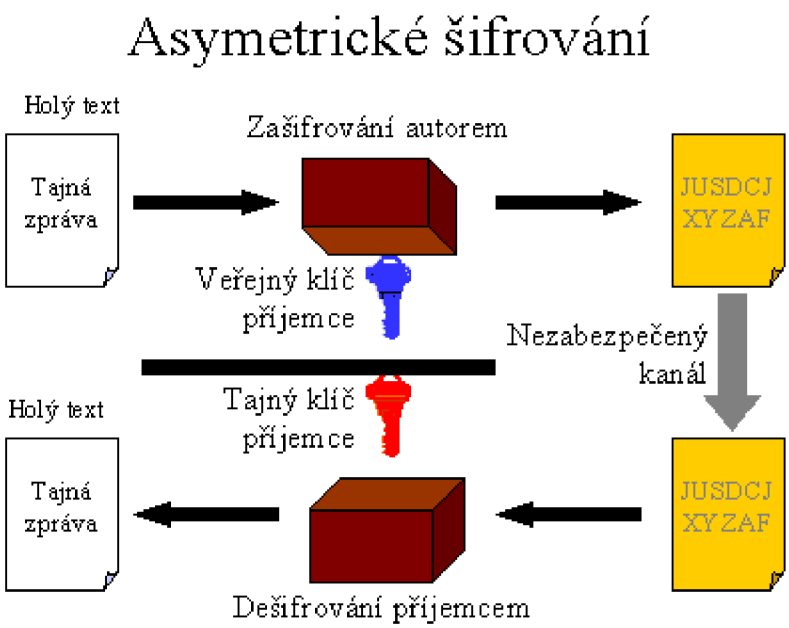
Programy pro jednotlivé části SMTP systému:
MUA (mail user agent) - Mozilla Thunderbird, Microsoft Outlook.
MTA (mail transfer agent) - Sendmail, Qmail
MDA (mail delivery agent) - Procmail, Maildrop

2.4 Bezpečnost v e-mailové komunikaci

Bezpečnost v e-mailové komunikace může mít dvě formy. První formou jsou digitální podpisy zpráv, které slouží k tomu abychom mohli ověřit to, že zprávu nikdo nemodifikoval, a že odesilatelem je skutečně ten, kdo se za něj vydává. Druhou formu by bylo zašifrování celého e-mailu tak, aby si ho nikdo nepovolaný nemohl přečíst.

2.4.1 Asymetrická kryptografie

Tato kapitola je volně převzata z [9]. Principem asymetrického šifrování je kódování dat pomocí šifrovacích klíčů. Tyto klíče jsou vždy minimálně dva, jeden soukromý klíč pro dešifrování, a jeden či více veřejných klíčů pro šifrování. V klasické asymetrické kryptografii probíhá výměna šifrovaných zpráv s tím, že ten kdo má veřejný klíč, má možnost s jeho pomocí zašifrovat zprávu tak, že ani on sám ji nebude schopen znovu dešifrovat pokud nebude mít k dispozici soukromý klíč. Jednoduchý příklad je vidět v následujícím obrázku.



obrázek 2.4.1 Asymetrické šifrování. Převzato z [9]

2.4.2 Digitální podpis

Při vytváření této kapitoly jsem se inspiroval na stránkách [10]. Jeden z prvků bezpečnosti realizovaných pomocí asymetrické kryptografie je digitální podpis. K vytvoření digitálního podpisu e-mailové zprávy je nejdříve potřeba její hash. Pomocí některé z hashovacích funkcí (například MD5) vytvoříme otisk zprávy který má pevnou délku (128 bitů). Pokud bychom změnili například i jediný znak v celé zprávě, tento otisk by vypadal naprosto jinak. Následně tento hash zašifrujeme pomocí soukromého klíče a přiložíme tento zašifrovaný otisk k e-mailové zprávě.

Příjemce následně tento zašifrovaný otisk rozšifruje pomocí veřejného klíče odesilatele zprávy, a jako výstup dostane hash původní zprávy. Pomocí funkce MD5 potom porovná samotný e-mail s jeho otiskem a pokud se shodují, tak je jisté, že tento e-mail skutečně odeslal majitel daného veřejného klíče a nikdo ho při přenosu nepozměnil.

2.4.3 Certifikáty X.509

Certifikáty standardu X.509 definované v RFC 3280 jsou v současné době jedny z nejpoužívanějších certifikátů. Jsou to certifikáty založené na principu asymetrické kryptografie. Ve většině případů je možno obstarat tyto certifikáty pomocí certifikačních autorit jako jsou Thawte nebo VeriSign. Existují i jiné možnosti získání certifikátů, ale pokud certifikát nebude vydán certifikační autoritou, tak bude obsah zašifrovaný tímto certifikátem nedůvěryhodný.

Struktura certifikátu je následující:

- Číslo verze (version)
- Sériové číslo (serialNumber)
- ID podpisového algoritmu (signature)
- Vydavatel (issuer)
- Platnost (validity)
  - Od (notBefore)
  - Do (notAfter)
- Subjekt (subject)
- Informace o veřejném klíči subjektu (subjectPublicKeyInfo)
  - Algoritmus veřejného klíče subjektu (algorithm)
  - Veřejný klíč subjektu (subjectPublicKey)
- Jednoznačná identifikace vydavatele (issuerUniqueID)
- Jednoznačná identifikace subjektu (subjectUniqueID)
- Rozšíření certifikátu (extensions)

Informace o certifikátech typu X.509 jsou převzaty z [11].

2.5 Protokol HTTP

Protokol HTTP neboli Hypertext Transfer Protocol je protokol aplikační vrstvy pro přenos jakýchkoliv dat. Původní verze umožňovali pouze přenos hypertextových dokumentů, ale od verze 1.0 je možné pomocí rozšíření MIME posílat jakákoliv data. Jedná se o obecný, bezstavový protokol, který může být použit pro mnoho úkonů, nejenom pro přenos hypertextových dokumentů. Základem tohoto protokolu jsou dotazy a odpovědi na ně.

2.5.1 Metoda POST

Tato metoda je základem komunikace webových služeb a jejich konzumentů. Na rozdíl od metody HTTP get, umožňuje tato metoda přenášet data v hlavičce zprávy, nikoliv v adrese URL jak je tomu v případě metody HTTP get, tudíž je HTTP post vhodnější pro přenášení SOAP zpráv.

3 Analýza zadání a návrh aplikace

3.1 Specifikace zadání

3.1.1 Specifikace webové služby

Zadáním této bakalářské práce bylo vytvořit webovou službu umožňující komunikaci uživatele dané webové služby se svými adresáty. Tato komunikace měla být realizována pomocí zasílání e-mailů adresátům a následnou reakcí na jejich došlé odpovědi. Tato reakce má být umožněna možností zaregistrování filtrů, které budou testovat vlastnosti daného příchozí e-mailu tím, že porovnají odesilatele, předmět a tělo e-mailu s hodnotami zadanými ve filtru . V případě že e-mail projde filtrem, musí webová služba reagovat tak, že kontaktuje webovou službu specifikovanou v daném filtru a předat jí informace o filtru a e-mailu, který filtr spustil. Dalším rozšířením této webové služby má být autentizace uživatele a možnost elektronického podpisu u odesílaných e-mailů.

3.1.2 Specifikace funkcí webové služby

V následující podkapitole se budu věnovat pouze veřejným funkcím, tedy funkcím, které je možno po zalogování do systému zavolat. Pomocné funkce webové služby, které jsou implementovány pro zpřehlednění kódu vynechám. Všechny funkce kromě funkce login používají jako jeden z argumentů vstupu proměnnou sessionID. Tato proměnná slouží k identifikaci uživatele.

- **Login**

  Uživatel odešle svoje heslo a login, služba tyto hodnoty porovná s hodnotami v lokálním souboru access.txt a na základě toho buďto odešle pětimístné číslo které bude použito jako sessionID pro další operace, nebo zápornou jedno-číselnou hodnotu označující kde nastala chyba v procesu autentizace.
- **Logout**

  Uživatel odešle sessionID, kterou dostal při zalogování do systému, služba smaže tento sessionID a odešle uživateli zprávu že zalogování proběhlo úspěšně.
- **NewTemplate**

  Tato funkce umožňuje uložení šablony e-mailu kterou je možno použít pro pozdější odeslání.
- **ReturnTemplates**

  Načte všechny šablony uživatele do listu šablon a vrátí je.

- **DeleteTemplates**

  Smazání šablony daného uživatele na základě předaného ID.
- **SendEmail**

  Prosté odesílání e-mailů, jako argumenty přijímá Recipients, Subject a Body a bool hodnotu digital signature, které značí jestli má být e-mail digitálně podepsán před odesláním.
- **SendEmailByTemplate**

  Odeslání e-mailu na základě uložené šablony. Jako argument požaduje ID šablony k odeslání.
- **RegisterEventHandler**

  Zaregistrování filtru pro příchozí odpovědi. Umožňuje specifikovat 3 prohledávané oblasti - odesílatele, předmět a vlastní tělo e-mailu. Dále je možno tento filtr modifikovat tím, že tyto oblasti budeme porovnávat s textem, který tam musí nebo nesmí být. Dále má tato funkce jako vstupní parametr adresu URI, kde se specifikuje adresa další webové služby, která se má zavolat při úspěšném vykonání filtru. Posledním vstupním argumentem je název proměnné, webové služby specifikované ve filtru, do které se bude ukládat ID filtru.
- **DeleteEventHandler**

  Smaže filtr na základě předaného ID.
- **ReturnEventHandlers**

  Načte všechny filtry uživatele a vrátí je.
- **ReceivedEmail**

  Tato funkce byla vytvořena pouze pro spolupráci s lokálním SMTP serverem, který bude předávat adresáty příchozích e-mailů webové službě za pomocí této funkce.
- **ReturnUsedEmails**

  Vrátí všechny e-mailové adresy, které kdy uživatel použil.
- **SaveNewTemplateVariable**

  Vytvoření nové proměnné pro šablony. Předávají se tři hodnoty, první je sessionID, druhý je název proměnné ve tvaru *&proměnná* a hodnota proměnné, tedy textová hodnota, které bude při odesílání e-mailu nahrazena za proměnnou.
- **ReturnTemplateVariables**

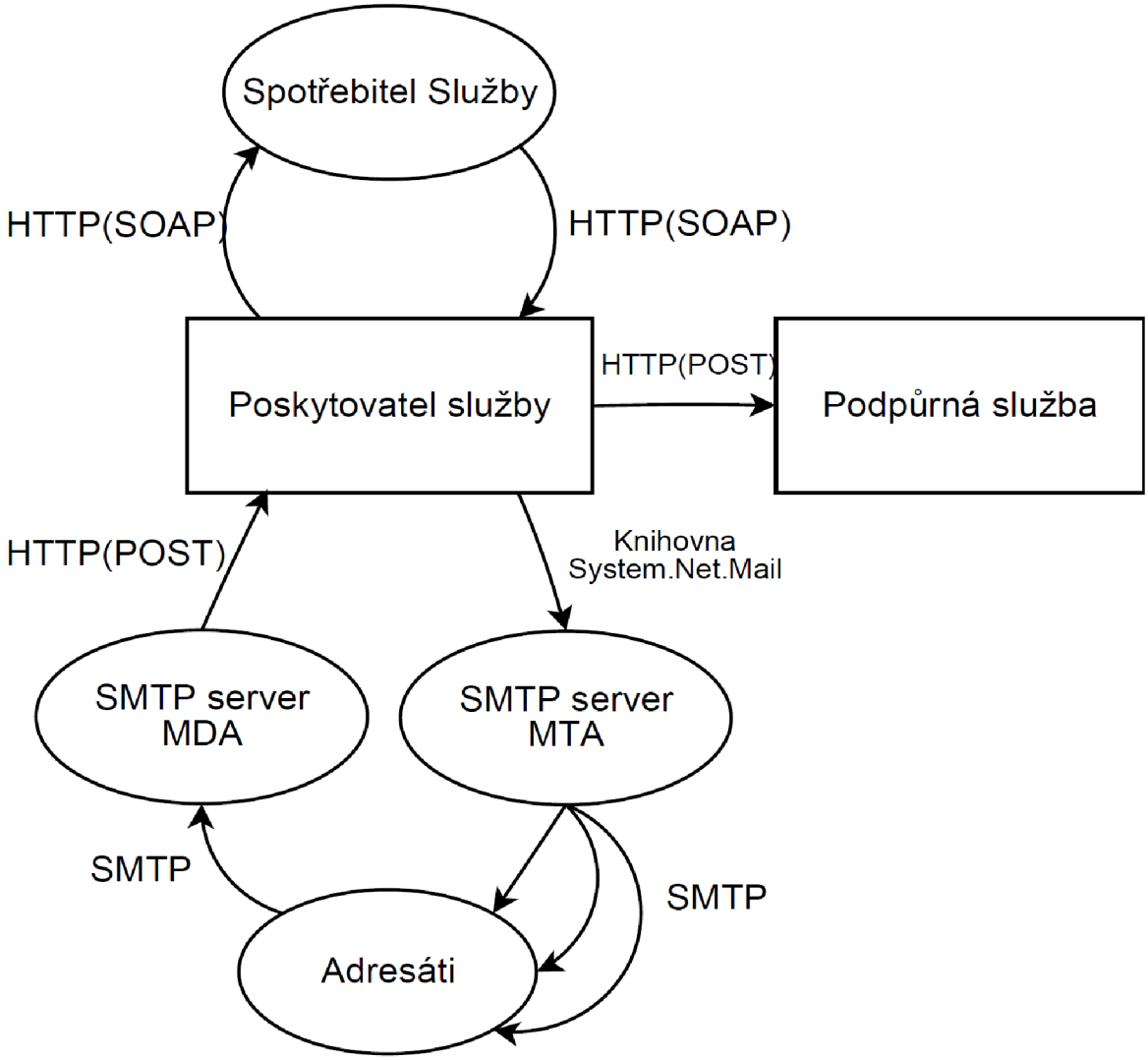
  Navrátí všechny proměnné pro šablony definované uživatelem.
- **DeleteTemplateVariable**

  Na základě předaného ID smaže proměnnou definovanou uživatelem.
- **ReturnReceivedOrSentEmails**

  Funkce vrací všechny odeslané nebo obdržené e-maily na základě sessionID a definice typu e-mailu. Tyto typy jsou "ReceivedEmail" a "SentEmail".

3.2 Návrh řešení

V této kapitole se budu věnovat jednotlivým částem systému, který má tvořit e-mailovou bránu daného informačního systému. Tyto části a způsob jakým spolu komunikují, je možno vidět na následujícím obrázku.



Obrázek 3.2 Návrh řešení

3.2.1 Poskytovatel služby

Jádro celého systému umožňuje uživateli za pomocí klienta ovládat všechny prvky systému. Webová služba bude uživateli poskytovat možnost sestavovat e-mailové zprávy za pomocí šablon a jejich proměnných, připravovat filtry jako přípravu na možné reakce od adresátů těchto zpráv. Při příchozí zprávě tuto zprávu porovná s filtry a v případě shody odešle informace o této shodě pomocné webové službě, která je implementována specificky pro tuto událost.

3.2.2 Spotřebitel služby

Jelikož k funkcím webové služby není možno přistupovat přímo, je potřeba pro komunikaci mezi uživatelem a webovou službou implementovat klienta. Jelikož se jedná o klienta webové služby, je možno ho napsat na libovolné platformě a v libovolném jazyce. V tomto případě se jedná pouze o prostředek pro demonstraci funkcionality webové služby a není součástí zadání, vybral jsem tedy platformu a jazyk s jejichž pomocí jsem již implementoval danou webovou službu, tedy platforma Windows a jazyk C#.

3.2.3 Podpůrná služba

Tato služba slouží pouze k demonstraci možnosti předávání informací mezi dvěma webovými službami. V tomto případě poskytovatel původní služby kontaktuje tuto webovou službu, pokud je přijat e-mail, které vyhovoval nastaveným filtrům, tato komunikace probíhá pouze pomocí metody HTTP POST, nepředává se tedy klasická SOAP zpráva.

3.2.4 SMTP server MTA

Přenos e-mailu je v mém systému řešen za pomocí dvou SMTP serverů. MTA server, tedy mail transfer agent slouží pro doručení e-mailů k adresátům informačních zpráv. Pro aplikaci tohoto serveru máme dvě možnosti: buďto definujeme vzdálený SMTP server na jiném serveru, který se bude starat o doručování této pošty, nebo spustíme na serveru další SMTP server na jiném portu než na portu 25, který bude obsazen SMTP serverem MDA.

3.2.5 SMTP server MDA

MDA server, tedy mail delivery agent bude v mém systému sloužit pro ukládání e-mailových odpovědí od adresátů původních informačních zpráv. Do tohoto serveru také hodlám přidat funkci, které bude v případě obdrženého e-mailu kontaktovat Informační službu. I zde bude zasílání zpráv založeno pouze na metodě HTTP POST, kde se bude zasílat pouze string, nikoliv celá SOAP zpráva.

4 Implementace

Popis Implementace této webové služby rozdělím do 3 podkategorií. Implementaci webové služby ovládající SMTP server, klienta ovládající tuto webovou službu, a nakonec úpravu SMTP server pro komunikaci s webovou službou.

4.1 Implementace webové služby

Pro implementaci jsem vybral jazyk C# a platformu .NET jelikož ve spojení s Microsoft Visual Studiem se toto jevilo jako nejlepší volba. Platforma .NET poskytuje mnoho užitečných knihoven, které usnadňují implementaci webových služeb.

4.1.1 Autentizace uživatele

Základním prvkem této webové služby je autentizace uživatele. Bez autentizace uživatele nelze plnohodnotně využívat tuto službu, jelikož téměř všechny funkce jako jeden ze vstupů vyžadují sessionID, které lze získat právě při autentizaci. Po zadání autentizačních údajů se tyto údaje ověří vůči souboru access v kořenovém adresáři služby. Po ověření správnosti údajů vrátí funkce login pětimístné číslo sessionID, které bude uživatel používat pro ovládání této webové služby. Jak jsem již zmínil na začátku tohoto odstavce, bez sessionID není možné zavolat většinu funkcí. Při autentizačním procesu se tento sessionID společně s loginem uživatele uloží do listu přihlášených uživatelů.

Během autentizace uživatele také na pozadí proběhne inicializace, která spočívá ve vytvoření složky uživatele pokud je toto jeho první přihlášení. Jestliže byl již uživatel přihlášen a při svém posledním přihlášení vytvořil proměnné pro šablony, případně odeslal e-maily, autentizační proces otevře uložené soubory a načte jejich obsah do listů vytvořených specificky pro tyto účely z důvodu snadnějšího přístupu k těmto datům.

V případě, že se uživatel přihlásí znovu i když je ještě zalogovaný, webová služba pouze smaže původní sessionID a nahradí ho novým. Tím pádem je znemožněno, aby více lidí používalo stejný účet v jednom okamžiku.

4.1.2 Práce se šablonami

Jedním z důležitých funkcí této služby je práce se šablonami. Uživatel má možnost pracovat se šablonami, které si sám vytvoří. Tyto šablony jsou uloženy ve složce uživatele v souboru Templates.txt, a každá z nich má svoje unikátní ID, podle kterého jsou rozlišeny. Na základě tohoto ID je možné i šablonu buďto odeslat, nebo smazat. Do šablony je klasicky možné uložit všechny prvky jednoduchého e-mailu, tedy adresáty, předmět a samotné tělo e-mailu. Do e-mailu je také možno přidávat proměnné. O těch bude pojednávat další kapitola.

4.1.3 Proměnné v šablonách

Součástí pokročilého zpracování šablon je možnost do šablon přidávat proměnné. Tyto proměnné mohou být buďto systémové nebo definované uživatelem. Proměnná je vždy uvozena znakem & a změna se aplikuje až při odesílání e-mailu. Systémové proměnné jsou:

1. &time - Čas odeslání e-mailu.
2. &date - Datum odeslání e-mailu.
3. &allRecipients - Vloží všechny e-maily, které kdy uživatel použil.
4. &recipientsCount - Vloží počet adresátů, které kdy uživatel použil.

Dále má možnost uživatel uložit svoje vlastní proměnné, tyto proměnné také musejí začínat znakem & a je možné uložit jakoukoliv textovou informaci jako jejich hodnotu..

4.1.4 Odesílání e-mailů

Tato část je implementována tak, že uživatel buďto přímo nebo přes šablonu sestaví e-mailovou zprávu k odeslání. Služba nejprve prověří, jestli tento e-mail neobsahuje proměnné. Pokud ano, tak služba všechny tyto proměnné nahradí jejich danými hodnotami. Adresa odesílatele je sestavena z loginu a jako doménové jméno slouží údaj v souboru settings.txt, kde je nastavena doména,na které bude webová služba eventuálně fungovat.

Jednou z možností, kterou má spotřebitel služby k dispozici, je možnost odeslat tento e-mail digitálně podepsaný. Pokud se rozhodne pro tuto možnost, načte webová služba certifikát ze souboru, vytvoří pomocí MD5 hashe otisk e-mailu, a následně tento otisk zašifruje pomocí soukromého klíče uživatele. Takto zašifrovaný otisk je přidán k e-mailu jako příloha.

Veškeré odeslané e-maily jsou uskladněny v souboru SentEmails.txt ve složce uživatele. Obsahují všechny atributy, které měli při odesílání a navíc má každý z nich unikátní ID aby bylo možné jednotlivé e-maily identifikovat.

4.1.5 Práce s certifikáty

Certifikáty jsou klíčovým prvkem pro realizaci digitálního podpisu e-mailu. Jsou dvě možnosti jak si obstarat certifikát.

1. Získat certifikát od Certifikační autority.
   Certifikát je možno obdržet od velkých certifikačních autorit. Jejich certifikáty jsou již vloženy do webových prohlížečů či poštovních klientů, takže již není potřeba distribuovat veřejné klíče k jednotlivým uživatelům. Problémem ale je, že certifikáty vydány těmito autoritami nejsou zadarmo pro komerční použití. I vydání osobního certifikátu pro e-mail nabízí zdarma pouze několik z nich.

2. Vygenerování vlastního certifikátu.
   Existuje několik programů které umožňují vytvoření vlastního certifikátu. Takto vytvořeným certifikátům ale webové prohlížeče či poštovní klienti nedůvěřují. Je tedy potřeba uživatelům předat veřejný klíč z tohoto certifikátů aby mohli ověřit, že tvůrcem digitálního podpisu daného e-mailu je skutečně ten, kdo se za něj vydává.

Já jsem v implementaci zvolil druhou možnost jelikož je zdarma, pokud by tato webová služba měla někdy sloužit jako brána informačního systému společnosti, rozhodně bych tuto část realizoval pomocí prvního způsobu, jelikož by odpadly starosti s ověřováním validity certifikátů.

K výrobě certifikátu a extrakci veřejného a privátního klíče je zapotřebí několik nástrojů.

1. makecert.exe - Nástroj pro samotné vytváření certifikátů. Je vydaný Microsoftem a slouží pro vytváření testovacích certifikátů. V naší aplikaci je ale budeme používat jako plnohodnotné certifikáty. Více o tomto nástroji je možno přečíst na [12].
2. cert2spc.exe - Potřebný pro převod certifikátu na soubor spc, ze kterého lze následně vytvořit PFX soubor, který budeme používat ve webové službě.
3. pvkimprt.exe - Tato aplikace vytváří ze souborů pkv a psc, tedy privátního a veřejného klíče soubor s koncovkou .pfx, tedy Personal information exchange. Tento soubor lze již použít v aplikaci pro vytváření digitálních podpisů.

Celý proces proběhne tak, že s nástrojem makecert vytvoříme certifikát pro daný e-mail, zároveň ale zadáme, aby se privátní klíč vygeneroval do jiného souboru s koncovkou pvk. Nástrojem sert2spc převedeme certifikát, který obsahuje pouze veřejný klíč do souboru spc. Následně použijeme soubory spc a pkv k vytvoření souboru pfx, který je už připraven k používání v aplikaci.

4.1.6 Filtry pro příchozí e-maily

Uživatel má možnost sestavit si filtry pro eventuální e-mailové odpovědi od adresátů jeho e-mailů. Vložení filtru probíhá tak, že uživatel má možnost hledat výskyt specifického textu v jedné ze 3 částí e-mailu, tedy odesílatele, předmětu a těla e-mailu. Při základním nastavení je možno nastavit, jestli daný text musí obsahovat tuto část e-mailu nebo naopak nesmí.

Filtry v této webové službě mají při spuštění jedinou funkci. A to předat ID filtru, E-mailu a jméno uživatele, které spustili událost, další webové službě. Weobvou službu kam se mají tyto informace předávat musí uživatel specifikovat společně s vytvářením filtru. Specifikací webové služby se v tomto případě rozumí vložit do filtru její URI a zároveň i proměnnou, do které se bude ID filtru předávat.

Pro vytvoření složitějšího filtru je možno kombinovat textové výrazy s logickými operátory OR a AND. Webová služba umožňuje tyto operátory i kombinovat, takže je možno vytvořit i poměrně složitý filtr. Jádrem vyhodnocování těchto výrazů je funkce *EvaluateExpressions*, která nejdříve všechny textové části výrazu ohodnotí buďto jedničkou nebo nulou v závislosti na tom, jestli výrazy jsou v e-mailu obsaženy nebo ne. V této fázi již máme připravený sled jedniček a nul oddělených logickými operátory. Tento výraz předáme funkci *Compute* knihovny System.data, která pro nás daný výraz vyhodnotí.

Pokud jsou hodnoty ve filtru vyhodnoceny jako pravdivé, zavoláme funkci *ExecuteFilter, která* načte URI a proměnnou webové služby určenou pro předání informací. Z těchto informací sestaví příkaz HTTP POST kterou odešle dané webové službě a tím předá požadované informace.

4.1.7 Struktura souborů uživatele

Webová služba ukládá všechny šablony, filtry, odeslané a přijaté e-maily do souborů ve složce uživatele. Jejich obsah je následující.

1. Templates.txt - Tento soubor obsahuje šablony a všechny e-maily, které kdy uživatel použil. Každá šablona má svoji unikátní ID.
2. Filters.txt - Soubor s filtry. Každý filtr má svoji unikátní ID.
3. SentEmails.txt - Soubor s odeslanými e-maily. Společně s e-mailem se ukládá i datum a čas odeslání. Každý e-mail má svoji unikátní ID.
4. ReceivedEmails.txt - Soubor s e-maily, které uživatel obdržel. I tyto e-maily se ukládají s informací o datu a času a jsou rozlišovány pomocí ID.
5. user@domainname.pfx - Soubor obsahující klíče potřebné pro odesílání digitálně podepsaných e-mailů.

Každý uživatel má svoji vlastní složku se soubory a zároveň není možné aby více uživatelů používalo jeden účet najednou. Takto je systém nastaven kvůli tomu, aby nedocházelo ke konfliktům při zápisu do stejného souboru. Dvě vlákna webové služby by se mohli najednou pokusit zapsat do stejného souboru, případně by jeden z uživatelů pracoval z daty, která by již byla ze souborů smazána a to by vyvolalo konflikty.

V případě použití ve větším systému by bylo vhodnější použití databázového systému. Prozatím tato webová služba pracuje se systémem souborů, jelikož je to podle mého názoru dostačující.

4.1.8 Odhlášení uživatele

Při odhlášení uživatele se smaže jeho záznam z listu Credentials, kde jsou uschovány sessionID a loginy všech přihlášených uživatelů, dále se uloží všechny e-mailové adresy, které byly použity do souboru se šablonami. Po odhlášení již není možno používat sessionID předanou při přihlášení.

4.2 SMTP server

Jelikož v zadání této práce nebyla implementace SMTP serveru stěžejní částí, rozhodl jsem se upravit stávající řešení autora Marufa Maniruzzamana, které jsem převzal z [13].

Tento SMTP server slouží pouze jako mail delivery agent, tedy stará se pouze o doručování zpráv k lokálnímu uživateli. Pro odesílání zpráv musíme využít SMTP server třetí strany, případně použít další SMTP server, který bude běžet také na lokálním počítači, ale na jiném portu.

Cílem mé úpravy bylo, aby při příchozím mailu SMTP server zároveň kontaktoval webovou službu a předal jí adresáta e-mailu. Webový služba čeká na tento typ příchozí komunikace, a po obdržení adresáta daného e-mailu vykoná danou funkci.

Tuto úpravu jsem implementoval do funkce *ProcessDATAEnd,* pokud program zavolá tuto funkci, je již ve fázi kdy je jisté, že se uživatel nachází na lokálním počítači a ukládá e-mail do složky, s mojí úpravou tato funkce vytvoří spojení s lokálním WWW serverem na portu 80 a předá příkaz HTTP post, sestavený danou funkcí. V těle příkazu HTTP POST se předá string,* který obsahuje adresu příjemce e-mailu.

4.3 Pomocná webová služba

Tato webová služba slouží v tomto systému pouze pro demonstraci možnosti předávání informací mezi dvěma webovými službami, tedy situaci která by ve složitějším informačním systému jistě nastala. Na vstupu funkce očekává string obsahující informace o spuštěném filtru v hlavní webové službě, jmenovitě ID spuštěného filtru, ID e-mailu, jenž prošel filtrem a jméno uživatele, kterému tento e-mail došel. Všechny tyto informace uloží společně s časem a datem do souboru pro pozdější zpracování.

4.4 Spotřebitel služby

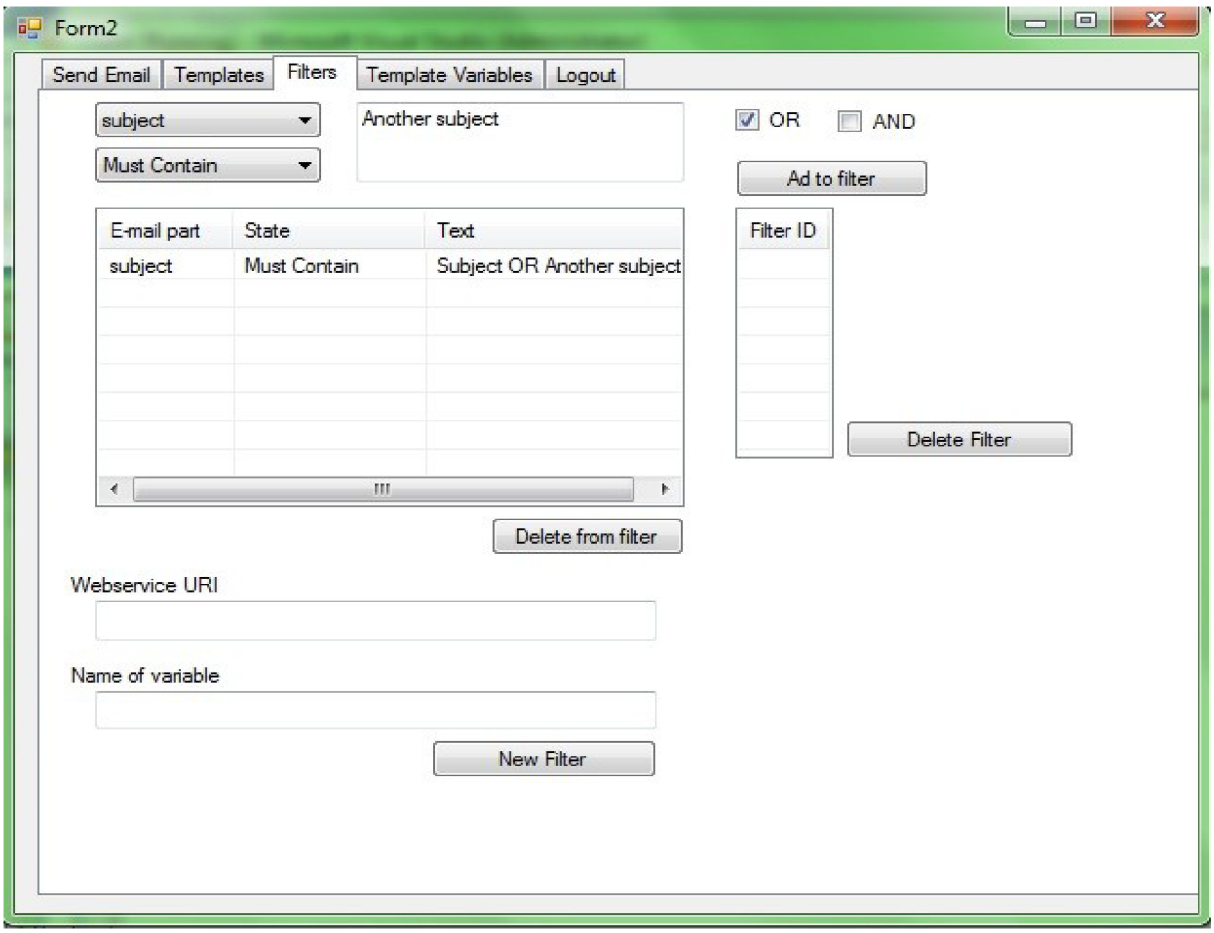
Spotřebitel služby je implementován pouze z důvodu otestování funkčnosti webové služby a pro lepší prezentaci jejich funkcí.

Rozhodl jsem se spotřebitele služby implementovat ve stejném prostředí jako webovou službu, tedy ve vývojářském programu Visual studio, a jazyku C#. Zvolil jsem typ aplikace Windows, která bude implementována za pomocí knihovny system.windows.forms.

Základem aplikace jsou dva formuláře. V prvním formuláři jsou umístěny pouze dvě textové pole a tlačítko pro přihlášení. Tyto dvě pole slouží pro vyplnění loginu a hesla. Po odeslání autentizačních údajů čeká formulář na odpověď. Pokud odpověď obsahuje pětimístné číslo, čili kladnou odpověď spustí se druhý formulář, který obsahuje prvky umožňující ovládání webové služby. Pokud je odpověď ve tvaru záporného čísla, zobrazí se na formuláři zpráva oznamující chybné zadání autentizačních údajů.

Druhý formulář obsahuje objekt TabControl, jde v podstatě o objekt, kde je možnost přepínat mezi plochami. V tomto Tabcontrolu jsou již obsaženy všechny ovládací prvky pro tuto webovou službu. Během inicializace druhého formuláře jsou pomocí funkcí webové služby načteny všechny šablony, filtry, e-maily a e-mailové adresy uživatele.

Spotřebitel služby je v této práci zahrnut pouze pro demonstraci funkcionality funkcí webové služby, z tohoto důvodu jsem mu nevěnoval takovou pozornost, jako v případě webové služby. I z toho důvodu je jeho design poměrně jednoduchý. Následující obrázek zobrazuje část spotřebitele služby, konkrétně část, která se zabývá Filtry.



Obrázek 4.3 Spotřebitel služby

5 Instalace webové služby

Tato kapitola se bude zabývat postupem pro instalaci a spravování webové služby, přidáváním nových uživatelů a instalací SMTP serveru potřebného pro běh aplikace.

5.1 Systémové požadavky

Platforma: Windows 7 Home Premium a vyšší verze. Windows Vista Home Premium a vyšší verze, Windows XP proffesional, Windows Server 2003 a vyšší.

V systému je nutné nainstalovat .NET Framework 3.5 a Microsoft IIS server. Ke správné funkcionalitě je též potřebná veřejná IP adresa kvůli tomu aby bylo možné přímo zpracovávat příchozí e-maily.

5.2 Instalace služby

Na přiloženém CD jsou ve složce *Instalattion* dvě podsložky. Jsou to složky x86 a x64. V těchto složkách se nacházejí instalační soubory pro danou platformu. Instalátor vytvoří všechny soubory a složky potřebné pro funkci této webové služby. Jako cíl instalace je nastavena složka Microsoft IIS serveru. Tedy *c:\inetpub\wwwroot.*

Výsledný systém souborů bude vypadat takto:

/wwwroot

/access	Tento soubor obsahuje loginy a hesla uživatelů.
/settings.txt	Tento soubor obsahuje nastavení serveru. Doménové jméno a případně adresu SMTP serveru pro odchozí poštu.
/informationsystem.asmx	Soubor webové služby, obsahuje informace o jazyku ,ve kterém je webová služba napsána, umístění zdrojového kódu a třídu.
/errors.txt	Soubor do kterého jsou zapisovány chybové stavy webové služby.
/bin/	Složka s dll a asmx.cs soubory.
/bakalarka.dll	Knihovna obsahující kompletní webovou službu.
/bakalarka.asmx.cs	Zdrojový kód třídy implementující webovou službu.
/users/	Složka obsahující jednotlivé složky uživatelů

Toto je systém souborů webové služby, instalační balíček bude dále obsahovat spotřebitele služby v podobě aplikace, a SMTP MDA server, který bude sloužit pro lokální doručování pošty. V případě, že uživatel nebude chtít používat SMTP server v té samé složce jako webovou službu, je potřeba do souboru settings.txt doplnit cestu ke složce kde bude SMTP server umístěn.

5.3 Přidání nového uživatele

Tato kapitola vznikla z informacích na této stránce [14]. Přidat nového uživatele lze editováním souboru *access*. Je potřeba přidat jak uživatele, tak i jeho heslo. Po zalogování do systému se uživateli vytvoří všechny potřebné složky, soubory pro šablony, filtry, odeslané a přijaté e-maily se vytvoří automaticky, jakmile do nich bude potřeba něco zapsat. Jediný soubor, který je potřeba přidat manuálně je pfx soubor s veřejným a privátním klíčem potřebným pro generování digitální podpisu nového uživatele.

Vytváření certifikátu probíhá za pomocí příkazové řádky windows, a k tomuto procesu je zapotřebí aplikací které se nacházejí na přiloženém CD ve složce Certifikates. Řádky podbarvené stupněm šedi jsou příkazy, které je možno zadat pomocí příkazové řádky.

Vytváření certifikátů

1. Pomocí programu makecret vytvoříme certifikát

```
makecert -r -n "CN=user, E=user@mydomain.com" -b 01/01/2000 -e 01/01/2099 -eku 1.3.6.1.5.5.7.3.4 -sv user@mydomain.com.pvk user@mydomain.com.cer
```

Při provedení příkazu se ukáže obrazovka, která umožňuje soubor s privátním klíček zaheslovat. V této verzi programu jsem od zaheslování upustil. Program tedy vytvoří dva soubory. Certifikační soubor .cer který obsahuje veřejný klíč, a soubor .pkv který obsahuje privátní klíč.

2. Pomocí programu certtospc vytvoříme soubor spc obsahující privátní klíč.

```
cert2spc user@mydomain.com.cer user@mydomain.com.spc
```

Tento krok je důležitý pro umožnění posledního kroku a to vytvoření souboru PFX, který už můžeme použít v aplikaci.

3. Pomocí prorgramu pvkimprt vytvoříme soubor pfx obsahující jak privátní tak veřejný klíč.

```
pvkimprt -pfx user@mydomain.com.spc user@mydomain.com.pvk
```

Po provedení příkazu se objeví obrazovka, kde zvolíme Yes, export private key. V další obrazovce zaškrtneme možnost delete private key if the export is sucessful a pokračujeme dále. Webová služba

bude jako heslo k certifikátu používat login uživatele, V tomto případě user. Jako jméno souboru zadáme user@mydomain.com. Jakmile je soubor pfx vytvořen, můžeme ho zkopírovat do složky daného uživatele. Jakmile soubor přesuneme, je možno odesílat digitálně podepsané e-maily.

5.4 Zprovoznění na doméně

Pro plnou funkcionalitu této webové služby je potřebná vlastní veřejná IP adresa, na které by mohl běžet webový server společně se SMTP serverem. Důležitým prvek pro správnou funkcionalitu je to, aby tuto adresu odkazoval DNS záznam typu A a MX.

DNS záznam Typu A je potřeba kvůli tomu, aby bylo možno URI webové služby používat ve tvaru http://myWebservice.com/InformationService.asmx. Pokud by nebyl přítomen A záznam, museli bychom místo doménového jména myWebservice.com publikovat přímo IP adresu, na které se nachází WWW server.

DNS záznam typu MX je nezbytný k tomu, aby vzdálené SMTP servery dokázali získat IP adresu SMTP serveru, kterému mají předávat poštu určenou pro danou doménu. Bez tohoto záznamu by bylo nemožné obdržet odpovědi od adresátů informačních e-mailů.

6 Možnosti rozšíření webové služby

V tomto odstavci se budu věnovat možným budoucím rozšířením této webové služby pokud by mělo dojít k jejímu využití.

6.1 E-mail

Tato webová služba implementuje pouze základní operace s e-mailem, jako je odesílání čistě textových e-mailových zpráv, a jednoduché digitální podepisování e-mailů. V současné době je ale vhodné aby aplikace, které se zabývají odesílám e-mailů měli více funkcí než pouze tyto dvě.

6.1.1 Protokol MIME

Tedy Multipurpose Internet Mail Extensions je jedním z prvních rozšíření, které bych rád přidal do webové služby. Implementováním toho protokolu bych umožnil případným uživatelům připojovat k odesílaným e-mailům různé typy souborů. Bylo možné odesílat e-maily formátované v HTML s obrázky a s možností připojit libovolný typ souborů.

6.1.2 Šifrování e-mailů

Dalším prvkem ke zlepšení celkové bezpečnosti zasílání zpráv by bylo šifrování odchozích e-mailů. Na rozdíl od digitálního podpisu by toto řešení bylo poměrně komplikovanější. K tomu, aby webová služba byla schopna odeslat šifrovaný e-mail tak, aby si ho mohl přečíst pouze její příjemce, bychom potřebovali od tohoto příjemce veřejný klíč k zašifrování dané zprávy. Příjemce by musel být zároveň i držitelem privátního klíče z tohoto páru klíčů, aby byl schopen následně zprávu rozšifrovat.

6.2 Multiplatformní webová služba

V této verzi je možné mojí webovou službu instalovat pouze na platformu Windows. Jelikož v současné době mají webové servery Apache mnohem větší zastoupení celosvětově, bylo by vhodné mít možnost přesunout tuto webovou službu právě na server Apache. Takovýto přesun by byl možný s použitím Mono project. Tento projekt byl vytvořen se záměrem usnadnit vývojářům vytváření multiplatformních aplikací. Specificky pro mojí potřebu bych použil Mono mód pro server Apache, který by umožnil hostování webové služby napsané v pod .NET přímo na serveru Apache.

6.3 Bezpečná autentizace

Současná implementace používá pro autentizaci pouze ověření vůči souboru s hesly. Myslím, že pro tuto službu je to dostačující bezpečnost, ale pokud bychom chtěli začít posílat důvěrný obsah v šifrovaných e-mailech, bylo by potřeba vylepšit i zabezpečení autentizace. Existuje několik možností jak zabezpečit nejenom autentizaci, ale i samotnou komunikaci mezi webovou službou a klientem. Tyto možnosti jsou popsané v [15]. Jako první možnost bych vybral zabezpečení pomocí SSL. Tato technologie zabezpečení je poměrně jednoduchá na implementaci, a poskytuje dostatečnou úroveň bezpečnosti.

6.4 Filtry

Pokud webová služba pomocí filtrů zachytí odpovídající e-mail, pouze přepošle informace o tomto e-mailu a filtru, který ho zachytil další webové službě. Toto řešení by se dalo rozšířit o možnost výběru, co má webová služba po zachycení e-mailu udělat. Například namísto kontaktování webové služby, by mohla odeslat parametry e-mailu a filtru na zadanou e-mailovou adresu.

Jako další úpravu by bylo možné implementovat možnosti, co všechno má webová služba po obdržení e-mailu předat za informace. V této implementaci webová služba pouze předá login uživatele, a ID e-mailu a filtru s tím, že si cílová webová služba tyto informace zpracuje. V rozšíření by mohlo být implementováno předání všech dostupných informacích, jak o e-mailu, tak i o filtru.

6.5 Systém fungující bez MDA SMTP serveru

V tomto rozšíření by se jednalo o možnost tohoto systému fungovat i bez MDA SMTP server, který by upozorňoval webovou službu na příchozí e-maily. Webová služba by tedy musela periodicky kontrolovat přítomnost nových zpráv na vzdáleném serveru. Ideální by bylo toto řešení implementovat pomocí POP3 protokolu.

7 Závěr

V této práci jsem rozebral princip fungování webových služeb, a všechny technologie, které se na jejich funkčnosti podílí. Dalším krokem bylo seznámit se s principem e-mailové komunikace a jejím zabezpečením. Po seznámení se všemi potřebnými protokoly a technologiemi jsem navrhl systém webové služby, která by fungovala jako e-mailová brána informačního systému. Tato služba umožňuje svým uživatelům dávat dohromady šablony, které je možné následně hromadně odesílat uživatelům jako e-mailové zprávy, V očekávání na jejich reakci může uživatel připravit filtry, a v případě, že adresát odpoví na původní e-mail, je tato odpověď srovnána s nastaveným filtrem a pokud je nalezena shoda, webová služba předá informace o této shodě další webové službě definované uživatelem. Jedním z prvků bezpečnosti v této službě je možnost odesílání digitálně podepsaných e-mailů, což by u informačního systému, který by pracoval s důležitými informacemi, jako například výpisy z bankovních účtů, nemělo chybět.

Webová služba byla navrhována pro použití na platformě windows ve vývojovém nástroji Visual Studio 2008 za použití jazyka C#. Jako jedno z možných rozšíření jsem navrhl možnost instalovat tuto službu i na server apache za pomocí nástrojů vyvíjených v projektu MONO. Myslím, že by bylo vhodné, aby si pořizovatel takovéhoto softwaru mohl vybrat, na jaké platformě by chtěl tento nástroj spustit, jelikož pokud by musel kvůli jedné službě měnit celý webový server, nejspíš by se porozhlédl po jiném řešení. V současné verzi je ovšem k jejímu provozu potřeba veřejná IP adresa, jelikož je potřeba přijímat odpovědi od adresátů přímo. Verzi, která by fungovala i bez veřejné IP adresy, jsem doporučil v rozšíření webové služby.

V současném stavu je služba plně funkční a po menších úpravách by ji bylo možné použít jako součást složitějšího informačního systému, který by ke své funkcionalitě využíval e-mailovou komunikaci.

Literatura

[1] W3C. Web Services Architecture Part 1.4 What is a Web service? [online]. 2002-2004 [cit. 2010-04-19]. Dostupný z WWW: <http://www.w3.org/TR/ws-arch/#whatis>

[2] KOSEK, Jiří. Inteligentní podpora navigace na WWW s využitím XML : Využití webových služeb a protokolu SOAP při komunikaci [online]. Vysoka škola ekonomická v Praze, 2002. 72 s. Diplomová práce. Vysoká škola ekonomická v Praze, Fakulta informatiky a statistiky. [cit. 2010-04-20] Dostupné z WWW: <http://www.kosek.cz/diplomka/html/websluzby.html>

[3] W3school. SOAP Tutorial [online]. 1999-2010 [cit. 2010-04-20]. Dostupný z WWW: <http://www.w3schools.com/soap/default.asp>

[4] W3C.SOAP Version 1.2 Part 0: Primer (Second Edition) Part 2.1 SOAP Messages [online]. 2002-2004 [cit. 2010-04-20]. Dostupný z WWW: <http://www.w3.org/TR/2007/REC-soap12-part0-20070427/#L1165>

[5] W3school. WSDL Tutorial [online]. 1999-2010 [cit. 2010-04-20]. Dostupný z WWW: <http://www.w3schools.com/wsdl/default.asp>

[6] W3school. WSDL Tutorial, WSDL and UDDI [online]. 1999-2010 [cit. 2010-04-21]. Dostupný z WWW: <http://www.w3schools.com/WSDL/wsdl_uddi.asp>

[7] W3school. XML Tutorial [online]. 1999-2010 [cit. 2010-04-25]. Dostupný z WWW: <http://www.w3schools.com/xml/default.asp>

[8] Roth D. Mark : sendmail Tutorial Part 1.1 MUAs, MTAs, and MDAs [cit. 2010-4-27] Dostupný z WWW :<http://www.feep.net/sendmail/tutorial/intro/MUA-MTA-MDA.html>

[9] Jankovič Jakub, Kolman Pavel: Šifrování na Internetu a PGP Kapitola 4. Asymetrická kryptografie [cit. 2010-4-29] Dostupný z WWW: <http://agrospolvb.cz/paja/download/site/asymetr.html>

[10] Jankovič Jakub, Kolman Pavel: Šifrování na Internetu a PGP Kapitola 6. Digitální podpis [cit. 2010-4-29] Dostupný z WWW: <http://agrospolvb.cz/paja/download/site/digital.html>

[11] BITTO, Ondřej. LUPA.CZ [online]. 10. 6. 2005 [cit. 2010-05-08]. Ukryto pod rouškou X.509. Dostupné z WWW: <http://www.lupa.cz/clanky/ukryto-pod-rouskou-x-509/>.

[12] Microsoft. MSDN [online]. c2010 [cit. 2010-05-07]. Certificate Creation Tool (Makecert.exe) . Dostupné z WWW: <http://msdn.microsoft.com/en-us/library/bfsktky3(VS.80).aspx>.

[13] MANIRUZZAMAN, Maruf. THE CODE PROJECT [online]. 23-9-2007 [cit. 2010-05-08]. SMTP Server. Dostupné z WWW: <http://www.codeproject.com/KB/IP/smtp-server.aspx>

[14] D'HEUREUSE, Christian. SOURCE-CODE.BIZ [online].[cit. 2010-05-15]. How to create a self-signed certificate that can be used to sign MS-Office VBA projects (Excel/Word macros) on multiple computers. Dostupné z WWW: <http://www.source-code.biz/snippets/vbasic/3.htm>

[15] Microsoft. MSDN [online]. c2010 [cit. 2010-05-15]. Securing XML Web Services Created Using ASP.NET . Dostupné z WWW: <http://msdn.microsoft.com/en- us/library/bfsktky3 (VS.80).aspx>.

Seznam příloh

Příloha 1. CD Obsahující zdrojové kódy, technickou zprávu, instalační soubory pro instalaci na platformě Windows a aplikace potřebné pro vytváření nových certifikátů pro uživatele.