Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Katedra kybernetiky

DIPLOMOVÁ PRÁCE

Hlasové rozhraní pro on-line kalendář

PLZEŇ, 2012 MARTIN SKÁLA

(originál zadání)

(originál zadání)

P R O H L Á Š E N Í

Předkládám tímto k posouzení a obhajobě diplomovou práci zpracovanou na závěr studia na Fakultě aplikovaných věd Západočeské univerzity v Plzni.

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a výhradně s použitím odborné literatury a pramenů, jejichž úplný seznam je její součástí.

V Plzni dne 18. 5. 2012

…….
vlastnoruční podpis

P O D Ě K O V Á N Í

Chtěl bych poděkovat panu Ing. Luboši Šmídlovi, Ph.D. za cenné rady a odborné vedení mé diplomové práce, které mi velmi pomohly a bez kterých by vypracování této práce bylo velmi obtížné.

A B S T R A K T

Tato diplomová práce se zabývá vytvořením hlasového rozhraní pro on-line kalendář. Jako on-line kalendář, byl použit kalendář firmy Google. V mojí práci se zabývám gramatikami, které využívá tento systém a parsováním promluvy. Vytvořil jsem webovou část, pro registraci uživatele a získání více informací o systému. Dále jsem vytvořil hlasové rozhraní, které obsahuje uzel pro snadné připojení dílčích aplikací a aplikaci na vytvoření událost a aplikaci na čtení události, pomocí níž je možné události také modifikovat a mazat. K realizaci dialogu je využit jazyk VoiceXML a k přístupu do kalendáře byla použita knihovna Zend Gdata.

K L Í Č O V Á S L O V A

Hlasové rozhraní, on-line kalendář, Google kalendář, Zend Gdata, VoiceXML, Google API, gramatiky, čtení událostí, vytváření událostí, mazání událostí, modifikace událostí.

A B S T R A C T

This thesis deals with the creation of voice interface for an online calendar. Google calendar is used as the online calendar. In this work I deal with grammars which are used by this system and also with speech parsing. I created a web interface for user registration and for getting more information about the system. I also created a voice interface which contains a node for easy conection to other applications such as an application for the creation of a new event or an application for reading stored events which can also modify and delete them. The code of the dialog is written in VoiceXML and Zend Gdata is used for the calendar acces.

K E Y W O R D S

Voice interface, on-line calendar, Google calendar, Zend Gdata, VoiceXML, Google API, grammars, read events, create events, delete events, update events.

Obsah:

1. Úvod

V dnešní době hrají v našich životech velkou roli informace a jejich rychlost předávání. Člověk, který má více informací má jistou výhodu před tím, který jich má málo, má tak lepší schopnost se rozhodovat. V předávání informací hraje velkou roli internet, který je v dnešní době značně rozšířený.

Způsob a rychlost šíření informace se od dřívějších dob velmi změnila. V dřívějších dobách, kdy neexistovali počítače, mobilní telefony ani nic podobného, bylo předání informace na dálku velmi složité. Vše se ale vyvinulo a v dnešní době s nástupem internetu není problém předávat informace napříč celým světem za zlomek sekundy.

Internet má také další výhodu v tom, že pomocí něj dokážeme shromáždit data, která jsou přístupná ze všech míst, kde je zaveden nebo také své informace můžeme sdílet s ostatními uživateli. Příkladem toho může být Google kalendář, který je uložen na serveru a uživatel má pomocí internetu do něj přístup odkudkoliv.

Co ale dělat v případě, že uživatel nemá momentálně k dispozici počítač s internetem nebo jiný přístup k internetu a má pouze obyčejnou telefonní linku? Jedinou jeho možností je někam zavolat a na potřebné informace se zeptat.

Pokud by uživatel ale někam zavolal a chtěl by se dozvědět nějaké informace z osobního kalendáře, určitě by nebyl rád, pokud by to zvedl člověk a on by musel říkat své uživatelské jméno a heslo do kalendáře a ptát se ho na nějaké osobní informace, které má v kalendáři.

Řešením takového problému je nahradit člověka strojem. Stroj má oproti člověku výhodu, že je dostupný 24 hodin denně a může na něj zároveň zavolat více uživatelů. Pokud by s uživatelem mluvil počítač, je nutné nějak zařídit, aby počítač komunikoval přirozenou řečí. Pokud by systém obsahoval omezené množství promluv, bylo by možné tyto promluvy namluvit dopředu. V mém systému pro on-line kalendář a mnoha dalších systémech ovšem omezené množství promluv není.

Tyto problémy byly důvodem, že se začaly vyvíjet systémy na automatické rozpoznávání a syntézu řeči. Díky tomu mohly vzniknout dialogové systémy, které například čtou výsledky přijímacího řízení, zajišťují přihlašování na zkoušky nebo čtou informace o počasí a dopravní informace. Všechny zmiňované aplikace byly vyvíjeny na ZČU na Katedře kybernetiky.

2. Cíl práce

Cílem této práce bylo vytvořit hlasové rozhraní pro on-line kalendář. Toto hlasové rozhraní by mělo přístup do on-line kalendáře registrovaných uživatelů. Registrovaný uživatel by pak mohl přistupovat do svého kalendáře pomocí telefonní linky. Uživatel by mohl přidávat do kalendáře události, číst je, mazat a také modifikovat. Důležité také je navrhnout dialog tak, aby bylo možné co nejsnadněji přidávat další funkcionality.

Je také potřeba vytvořit webovou část, kde by se uživatelé mohli registrovat, popřípadě měnit své údaje a kde mi mohl administrátor jednotlivé uživatele spravovat. V této webové aplikaci by se také uživatelé mohli dozvědět více o systému.

Celý tento systém je třeba napsat v jazyce VoiceXML a pomocí VoiceXML interpretu připojit systém k telefonní lince, přes kterou by se systémem mohl komunikovat uživatel. Je také nutné vybrat vhodný nástroj pro přístup do kalendáře. Rozhodnout se, zda je vhodné použít Google API nebo vybrat nějakou alternativu.

Je potřeba vytvořit vhodné gramatiky, aby uživatel mohl snadno komunikovat se systémem, pokud možno přirozeným jazykem a aby se daná promluva poté dala parsovat a mohli jsme získat potřebné informace.

Protože VoiceXML je textový jazyk a každý uživatel potřebuje jinak upravený dialog, je nutné využít jazyk PHP, který zpracuje získané informace a upraví dialog podle požadavků uživatele.

Jednotlivé cíle:

1. Webová aplikace
   - 1.1. Registrace uživatele
   - 1.2. Změna údajů registrovaného uživatele
   - 1.3. Administrační část

2. Hlasová aplikace
   - 2.1. Vybrání vhodného přístupu do kalendáře
   - 2.2. Navržení vhodných gramatik
   - 2.3. Vytvoření uzlu pro připojení aplikací
   - 2.4. Aplikace na vytvoření události
     - 2.4.1. Získání dalších informací
     - 2.4.2. Vytvoření události
   - 2.5. Aplikace na čtení událostí
     - 2.5.1. Čtení událostí
     - 2.5.2. Smazání událostí
     - 2.5.3. Modifikace události

3. Dialogové systémy

Hlasové dialogové systémy slouží ke komunikaci uživatelů s počítačovými či internetovými aplikace a to pomocí hlasu. Tyto aplikace mohou být například databázové a expertní systémy, systémy automatického řízení, ovládání, monitorování a podobně. Dialogové systémy tedy zprostředkovávají techniky pro komunikaci s počítačem přirozeným hlasem, tvoří interface mezi člověkem a aplikací.

Takovýchto systému může být celá řada a to od velmi jednoduchých, kdy člověk může vyslovit jenom určité povely, až k daleko složitějším, kdy člověk komunikuje se systémem souvislou přirozenou řečí. Nicméně v současnosti se pro vědní disciplíny i komerční aplikace předpokládá, že hlasový dialog, je omezen pouze na určitý účel (například čtení počasí a zpráv, přihlašování na zkoušky a podobně). Nejedná se tedy o systém, kdy by mohl člověk komunikovat s počítačem na libovolné téma.

Pokud má být úspěšná interakce uživatele s hlasovým dialogovým systémem, je nutné zajistit kromě modulu, který řídí dialog, také moduly na automatické rozpoznávání řeči, modul na syntézu řeči a také modul na porozumění mluvenému jazyku (přesněji modul pro porozumění posloupnosti slov, kterou poskytuje rozpoznávač řeči).

Lidé dokážou správně interpretovat informaci i v případě ztížených komunikačních podmínek a to proto, že mají určitou obecnou apriorní znalost o tématu konverzace. Jestli chce tvůrce systému dosáhnout podobných podmínek, je nutné provést přesnou analýzu úlohy a zajistit modelování zdrojů znalostí. Otázkou tedy je, jak zajistit potřebné informace a jak stanovit odpovídající architekturu. Zajistit tak bezproblémový a uživatelsky přívětivý provoz.

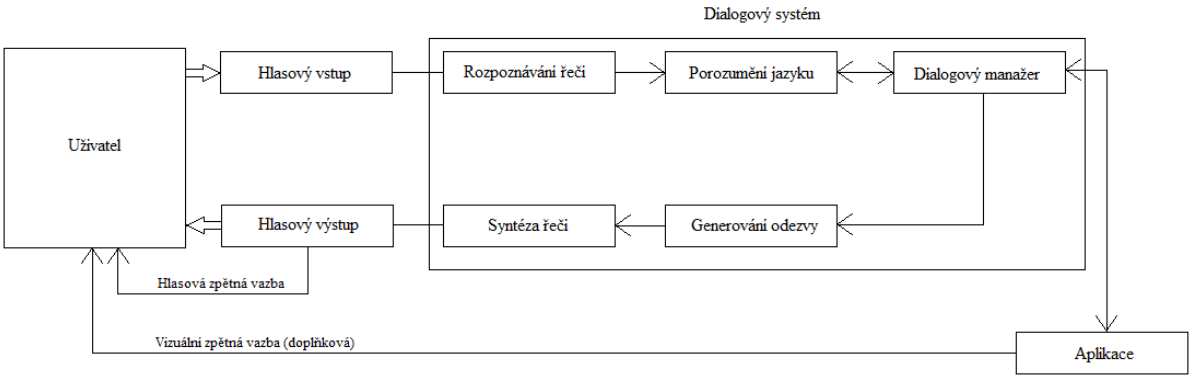
Využití řečového vstupu sebou nese klady ale určitě i nějaké zápory, proto než navrhneme nějaký dialogový systém, je třeba se rozhodnout, zda je použití řečového vstupu (výstupu) vhodné. Mezi nevýhody patří menší přenosová rychlost informace, než u vizuálního přenosu. U vizuálního přenosu, můžeme rychle pohlédnout na scénu a rychle vyhledat požadované informace. Jelikož je řeč sériová, o celkový pohled na informace jsme ochuzeni. Pokud je promluva velmi dlouhá, může se stát, že informace, které byly vysloveny na začátku promluvy, postupně zapomínáme. Proto je nutné promluvu někdy několikrát zopakovat a to je časově náročné. Oproti tomu je vizuální obnova informace daleko rychlejší.

Další nevýhodou řeči, je její povaha, kterou se nedá snadno ignorovat. Pokud chceme ignorovat obraz, který nás nezajímá, tak od něj odvrátíme zrak, kdežto se zvukem to nejde tak lehce a může to například rušit ostatní spolupracovníky a vytvářet tak nepříjemné pracovní podmínky. Řešením tedy je použití sluchátek nebo zvukově oddělené pracovní prostředí. Syntetická řeč také není tak kvalitní jako přirozená, proto jsou snadněji zřetelné její nedostatky.

I přes tyto nevýhody, existuje mnoho oblastí, kde je využití dialogových systému, tedy komunikace člověka s počítačem přirozenou řečí, výhodná. Pokud třeba uživatel má oči a ruce plně zaměstnány jinými úkoly, je vhodné použít ke komunikaci hlas. Nebo pokud uživatel potřebuje být pohyblivý a jiná vstupně výstupní zařízení nejsou efektivní. Dále pokud je uživatel vzdálen od systému a jedinou jeho možností je použít běžné telefonní spojení. Další využití může být pro zrakově hendikepované osoby nebo pro osoby s pohybovými

problémy. Pro zrakově postižené, mohou dialogové systémy zprostředkovat informace, které by jinou cestou nemohli získat. Pro pohybově postižené mohou být tyto aplikace prospěšné pro ovládání zařízení pomocí hlasu (kolečkových křesel, televize, světel a podobně).

Aby se dialogový systém stal skutečnou pomůckou, je nutné dodržovat několik zásad. Měl by být robustní vůči interakci s uživatelem a zároveň by měl být uživatelsky přívětivý. Systém by také měl být aktuálně prospěšný pro uživatele a neměla by být pouze novinkou, která zvyšuje prodej. Komunikace by měla být co nejvíce přirozená. V ideálním případě, by systém poskytoval uživateli možnost zeptat na danou informaci jakýmkoliv způsobem. Systém by měl být navržený tak, aby uživatele zbytečně nezdržoval, vhodně rozpoznal jeho promluvu a efektivně poskytnul uživateli potřebné informace. Uživatel by měl mít pocit, že se skutečně podílí na případných řídících akcích systému. Uplatněním lidského faktoru při návrhu systému by měla být potencionální spokojenost uživatele se snadnou a pohodlnou obsluhou [1].



Obr. 1: Dialogový systém.

Obrázek znázorňuje schéma dialogového systému. Uživatel vytvoří hlasový vstup. Poté se zpracuje rozpoznávačem řeči a porozuměním promluvy a vstoupí do dialogového manažeru, kde se řídí dialog. Manažer komunikuje s aplikací, kam zadává a odkud získává data, ze kterých pak generuje odezvu, která se syntetizuje a přes hlasový výstup se předá uživateli [1].

4. VoiceXML

Jako mnoho jazyků, jako například HTML (HyperText Markup Language), tak i VoiceXML (Voice eXtensible Markup Language) vzniklo z jazyka XML. Byl vytvořen fórem, které založili firmy IBM, Motorola, Lucent a AT&T. VoiceXML slouží ke zpřístupnění obsahu z internetu přes telefonní linku a využil jsem ho při tvorbě našeho dialogového systému [2].

4.1. Popis VoiceXML

VoiceXML byl založený na principu hlasové komunikace mezi počítačem a člověkem. Tento jazyk zajišťuje vstup pomocí tónové volby telefonu (vstup přes tlačítka) a vstup pomocí rozpoznávání mluvené řeči. Umožňuje také například nahrávání hovoru, který se pak může uložit, přesměrování hovoru, popř. jeho ukončení. Výstup je realizován buď do audio souboru nebo syntézou řeči (text-to-speech) do výstupního zařízení (telefonní přístroj).

VoiceXML je vysokoúrovňový jazyk (skrývá před vývojáři obecné věci, jako např. nastavení telefonní karty), pro vývojáře velmi příjemný, ve kterém se dají snadno napsat aplikace jakékoliv složitosti. Nároky mezi uživatelem a serverem, ze kterého získává následné informace, se zmenšují, protože nejprve od uživatele získá všechny potřebné informace a až poté odesílá dotaz na server.

Spojení mezi uživatelem, který volá z telefonního přístroje, dále pak následuje počítač s telefonní kartou a VoiceXML interpretem, který zajišťuje ASR (zpracování řeči) a TTS (syntézu řeči). To je připojeno na Dokument server pomocí TCP-IP protokolu, na kterém jsou uloženy VoiceXML dialogy a je spojeno také přes TCP-IP se serverem, kde jsou uloženy informace, které nás zajímají.

Velmi zjednodušeně by se dalo říct, že VoiceXML je jako HTML, s tím rozdílem, že HTML slouží k vizuální prezentaci dat a VoiceXML slouží ke zvukové prezentaci dat [18].

4.2. Historie

V březnu roku 1999 založili firmy IBM, Motorola, Lucent a AT&T forum, které položilo základ jakyku VoiceXML. V září vydali verzi VoiceXML 0.9, která sloužila k okomentování od jednotlivých členů fóra.

První oficiální verze byla VoiceXML a vyšla v dubnu roku 2000. Krátce potom vzniklo konsorcium W3C, které pokračovalo ve vývoji jazyka. W3C vydalo v březnu 2004 další verzi, a to VoiceXML 2.0. Tato verze obsahovalo kromě popisu dialogů a telefonních přístrojů, značek pro syntézu řeči a gramatik, také rozšíření do několika příbuzných jazyků, kde každý z nich se zaměřuje na určitou oblast hlasové komunikace [18].

4.3. Základy jazyka

Každý VoiceXML dokument obsahuje kořenový element, který začíná tagem <vxml> a končí tagem </vxml> [3].

4.3.1. Formuláře

Jsou základním stavebním kamenem VoiceXML dokumentu, slouží k rozčlenění dialogu a obsahují prvky pole (field items) a kontrolní prvky (control items).

Prvky pole – definují proměnnou a získávají vstup od uživatele. Obsahují tyto elementy:

- <field> - Obsahuje hodnotu získanou pomocí tónové volby nebo rozpoznáním řeči.
- <subdialog> - Zavolá jiný dialog a poté se vrátí zpět.
- <object> - Spustí objekt s parametry.
- <record> - Nahrává audio vstup od uživatele.
- <transfer> - Přesměruje uživatele na jiné číslo.
- Obecné elementy (<filled>, <prompt>, <grammar>, <dtmf> a <catch>).

Kontrolní prvky – Používají se nejčastěji s elementem <prompt> a informují uživatele o vstupu. Elementy kontrolních prvků:

- <block> - Slouží pro výpočty a následné přehrání výstupu.
- <initial> - řídí dialog mezi počítačem a uživatelem, kde se počítač s uživatelem mezi sebou střídají [18].

Formulář je uzavřen mezi tagy <form> a </form>. Příklad takových to formulářů může vypadat například takto:

```
<?xml version="1.0" encoding="Windows-1250"?>
  <vxml version="1.0">
    <form id="node">
        <property name="timeout" value="20" />
        <property name="inputmodes" value="voice"/>
        <property name="maxspeechtimeout" value="10" />
        <field name='vstup'>
          <grammar src="grammars/calendar.php"/>
                <prompt bargein="false">
                  Vyslovte prosím další příkaz.
                </prompt>
                <filled>
                   <submit expr="'engine.php?'" method="post"
                           namelist="vstup"/>
                </filled>
                <catch event='filled noinput nomatch spoke-too-soon'>
                        <prompt>
                          Zkuste to prosím znovu.
                        </prompt>
                        <reprompt/>
                </catch>
        </field>
```

```
    </form>
  </vxml>
```

Tento dialog slouží k vyslovení příkazu z gramatiky calendar.php.

4.3.2. Menu

Menu je dalším základním kamenem vedle formuláře. Využívá se tam, kde je třeba vytvořit pole, ve kterém se vybírá přesměrování dokumentu na další dokument (nebo jeho část) a rozhoduje se podle vstupu z tónové volby nebo rozpoznání hlasového vstupu.

Element <choice>: Obsahuje hodnotu, kam je uživatel přesměrován, při zadání určitého vstupu. Obsahují aribut dtmf, který odchytává vstup z tónové volby. Dále obsahuje atribut next, jehož hodnota je adresa, na kterou se dialog přesměruje nebo atribut event, který obsahuje událost, která při dané volbě provede. [18]

Menu je uzavřeno mezi tagy <menu> a </menu>. Příklad menu:

```
<?xml version="1.0" encoding="utf-8"?>
  <vxml version="1.0">
    <help>
      Po přečtení záznamu máte následující možnosti. Stiskutím jedničky
      přejdete na další záznam, stisknutím dvojky záznam zopakujete,
      stisknutím trojky přejdete na předchozí záznam.
     <reprompt/>
    </help>
    <menu id="zprava">
        <property name="inputmodes" value="dtmf"/>
        <property name="maxdigits" value="1"/>
        <property name="timeout" value="1"/>
        <prompt>

            09.05. od 12:00 do 09.05. 13:00 Schůzka s Karlem. V Plzni.
            Důležitá schůzka o novém projektu.

        </prompt>
        <choice dtmf="*" event="help"/>
        <choice dtmf="1" next="index.php?n=-1"/>
        <choice dtmf="2" next="index.php?n=0"/>
        <choice dtmf="3" next="index.php?n=1"/>
        <noinput>
           <goto next="index.php?n=1"/>
        </noinput>
        <nomatch>
           <goto next="index.php?n=1"/>
        </nomatch>

    </menu>
</vxml>
```

Toto je příklad menu, kde se přečte událost a uživatel má volbu přeskočit na čtení jiné události.

4.3.3. Gramatiky

Každý dialog je spojený s nějakou řečovou nebo dtmf gramatikou. Ve strojově řízených aplikacích je aktivní pouze ta gramatika dialogu, který je právě čten. Pokud je aktivní dialog, řízený uživatelem a počítačem dohromady, je vždy aktivní ta gramatika, která odpovídá promluvě uživatele a promluva je předána do dialogu s danou gramatikou [18].

4.3.4. Linky

Specifikují gramatiku, když je člověk v dosahu linku. Když souhlasí vstup s danou gramatikou, tak se dialog přesune do dalšího dialogu určeného atributem next nebo vykoná určitou událost určenou atributem event [18].

4.3.5. Události

Události (events) jsou vyvolávány uživatelem nebo vykonávaným programem za různých podmínek. Definují, co se stane, když například uživatel zadá špatný vstup, nebo když nezadá žádný vstup. [18]

- <noinput> - Uživatel nezadal žádný vstup.
- <nomatch> - Uživatel zadal špatný vstup.
- <help> - Obsahuje nápovědu k dokumentu.
- <error> - V dokumentu se objevila chyba.

4.3.6. Vlastnosti

Neboli properties definují vlastnosti čtení dokumentu, například míru podobnosti hlasového vstupu nebo například maximální možný počet vstupních znaků. Vlastnosti se zadávají do nepárového elementu <property> a obsahují atribut name a value [18].

Příklad vlastností:

```
<property name="inputmodes" value="dtmf"/>
<property name="maxdigits" value="1"/>
<property name="timeout" value="1"/>
```

4.3.7. Aplikace

Aplikace je sada dokumentů, které jsou připojeny na jeden kořenový dokument. Každý dokument z aplikace má do tohoto kořenového dokumentu přístup. Každá aplikace má svůj kořenový dokument [18].

5. Google a jeho API

Hlasové rozhraní, je vytvořeno pro kalendář, který vytvořila společnost Google, proto si nejprve přiblížíme co to je Google, o působení společnosti a některých jejich službách, které jsem využil v této práci.

5.1. Co to je Google

Google je americká společnost sídlící v kalifornském Silicon Valley, známém tím, že zde sídlí největší firmy počítačového průmyslu. Google se proslavil díky svému internetovému vyhledávači, který je mezi uživateli velmi oblíbený a jeho obliba stále stoupá. Ohodnocení webových stránek podle důležitosti probíhá pomocí algoritmu zvaného PageRank. Další výhodou tohoto vyhledávače je vyhledávání frází na webových stránkách, pokud jsou slova z těchto frází pohromadě a ne rozmístěna různě v textu. Popularita vyhledávače je tak veliká, že synonymem pro vyhledávání na internetu se stalo slovo „googlit“, které se dokonce v roce 2006 dostalo do Oxford English Dictionary [4].

Obr. 2: Logo Google.

5.1.1. O společnosti

Začátek vzniku sahá do roku 1996, kdy dva studenti informatiky, ze Stanfordovy univerzity, vytvořili vyhledávač s názvem Backrub, který důležitost webových stránek, vyhodnocoval na základě odkazů. V roce 1998 vše dostalo formální podobu, když založili firmu Google. Od této doby společnost začala růst obrovskou rychlostí. Svůj vyhledávač rozšířili o další jazyky a vytvořili mnoho dalších služeb a produktů [5].

Společnost rozšířila své pracoviště do mnoha zemí po celém světě a počet zaměstnanců vyrostl na několik desítek tisíc (v roce 2011 to bylo 26 316 lidí).

5.1.2. Co znamená slovo Google?

Název Google vznik od slova „googol“, což je matematický výraz pro jedničku následovanou sto nulami. Toto slovo poprvé použil v roce 1938 devítiletý synovec amerického matematika Edwarda Kasnera. Název měl poukazovat na obrovský objem

informací, které jsou na světě a vizi společnosti, která si kladla za cíl toto velké množství informací organizovat a učinit je dostupnými [5].

5.1.3. Filozofie společnosti

Spoluzakladatel Larry Page řekl: „Dokonalý vyhledávač by přesně pochopil, co hledáte, a také vám to nabídl.“ Úspěšnost prohlížeče byla ze začátku tak velká, právě pro to, že dokázal poskytnout odpověď rychleji a lépe než ostatní. V dnešní době se společnost snaží řídit určitého „desatera“ a možná i díky němu, je i dnes tak úspěšná [6].

Desatero, kterým se společnost řídí:

1. **Soustřeďte se na uživatele a ostatní přijde samo**
   Společnost při vývoji klade důraz na to, aby produkty budily co nejlepší uživatelský dojem. Aby vše bylo jednoduché a rychlé. Reklamy mají relevantní obsah a nepůsobí rušivě. Nové nástroje by také měli fungovat tak, aby uživatele ani nenapadlo, že by to mohlo vypadat jinak.

2. **Nejlepší je dělat jen jednu věc, ale dělat ji opravdu, opravdu dobře**
   Google se věnuje hlavně vyhledávání. Mají největší výzkumný tým na světě, který se zabývá právě jenom vyhledáváním. Důkladně zkoumá složité problémy a jejich řešením zlepšuje své služby. Snaží se vyhledávání zavést do nových oblastí a lidem tak umožnit využití ještě více informací.

3. **Raději rychle než pomalu**
   Čas uživatele je velmi cenný a každý kdo hledá informace na internetu, nechce dlouho čekat. Google se snaží z vyhledávače a jeho domovské stránky odstranit zbytečné bity a bajty, aby zvýšili efektivitu pracovního prostředí. Díky tomu vyhledávání trvá zlomek sekundy. Co nejvyšší rychlost se snaží docílit u všech svých produktů.

4. **Demokracie na webu funguje**
   Vyhledávání závisí na milionech jednotlivců, kteří přidáváním odkazů na různé stránky, určují hodnotu obsahu. Každé nové stránky znamenají určitý hlas do tohoto hodnocení, aby pak při vyhledávání mohla být zvolena nejhodnotnější stránka pro hledaný výraz. Algoritmus, který toto vyhodnocuje je patentovaný, nese název PageRank a hodnotí weby na základě více než 200 faktorů.

5. **Někdy potřebujete odpověď, i když právě nesedíte u počítače**
   Společnost Google vyvíjí a tedy také nabízí nové řešení mobilních služeb, která pomáhají lidem na celém světě provádět mnoho úkonů od čtení e-mailů až po sledování videí. Vyvinuli například bezplatnou mobilní platformu Android, která má otevřený zdrojový kód a nabízí nové možnosti operátorů, výrobcům a vývojářům.

6. **Můžete vydělávat peníze i bez toho, abyste dělali něco špatného**
   Příjmy společnosti pocházení z poskytování vyhledávacích služeb a z prodeje reklamního prostoru. Této propagace svých produktů využívají stovky tisíc inzerentu pomocí služby AdWords a naopak vydavatelé využívají služby AdSence. Reklamy na Googlu jsou textové a relevantní, nejsou povolena žádná vyskakovací okna a není možné si zaplatit zvýšení svého PageRanku. Uživatele tak oceňují nestrannost a tu nemůže žádný krátkodobý zisk nahradit.

7. **Vždycky existuje ještě více informací**
   Když začal mít vyhledávač Google naindexováno více stránek než konkurence, začali se jeho výzkumníci zaměřovat i na vyhledávání jinde, než jenom v textu na webových stránkách. Například vyhledávání obrázků, vyhledávání v knihách a odborných časopisech a podobně.

8. **Potřeba informací překračuje všechny hranice**
   Společnost vznikla ve spojených státech, ale více než polovina výsledků vyhledávání je poskytována lidem žijícím mimo USA. Díky tomu vznikly pobočky ve více než 60 zemích, provozují přes 180 domén ve více než 130 jazycích. Pomocí překladatelských nástrojů, mohou výsledkům hledání porozumět i lidé z opačné strany Země.

9. **I bez obleku to můžeme myslet vážně**
   V Googlu věří, že velké myšlenky mohou vzniknout díky správné firemní kultuře. Kladou důraz na týmovou práci a na hrdost individuálních úspěchů. Atmosféra ve firmě je nenucená, ale nápady vznikají například i ve frontě na oběd, na setkání týmu nebo v tělocvičně.

10. **Když je něco dobré, může to být ještě lepší**
    Dokonalost je pro ně začátek, nikoliv konec. Stanovují si nesplnitelné cíle a usilují o jejich splnění a tím dosáhnou nečekaně vysokých výsledků. A vše co správně funguje, se snaží vylepšit, aby to fungovalo ještě lépe. Když jeden z techniků zjistil, že vyhledávání správně napsaných slov funguje tak jak má, zkoušel, jak to bude fungovat s překlepy a díky tomu vyvinul kontrolu pravopisu.

5.1.4. Google služby

Google nenabízí pouze vyhledávač, ale poskytuje mnoho dalších služeb. Jsou rozděleny do osmi kategorií a v krátkosti si je představíme [7].

Web:

Vyhledávání – Hlavní služba Googlu. Rychlé prohledávání webu s relevantními výsledky viz text dříve.

Toolbar – Díky Toolbaru si můžete vyhledávací pole Googlu přidat na vlastní stránky.

Google Chrome – Moderní webový prohlížeč, který klade důraz na rychlost.

iGoogle – Domovská stránka Googlu, kterou si může uživatel přizpůsobit podle svých představ přidáváním a organizováním gadgetů jako jsou například Zprávy, Počasí, Gmail a podobně.

Pro váš mobil:
Mobil – Na adrese http://m.google.com naleznete stránky uzpůsobené po mobilní zařízení, abyste mohli snadno používat ostatní služby.

Vyhledávání pro mobily – Vyhledávat na internetu můžete také pomocí mobilního telefonu.

Mapy pro mobily – Také mapy můžete prohlížet v mobilu.

Geo produkty:
Mapy – Pomocí map můžete hledat trasy, měřit vzdálenost, přidat na mapu sídlo vaší firmy nebo také vytvořit vlastní mapu.

Earth – Pomocí virtuálního globusu můžete cestovat po celém světě, prohlížet si mapy, satelitní snímky, budovy ve 3D a další. Nyní můžete prozkoumávat i například Měsíc nebo Mars, ne jenom Zemi [8].

Panoramio – Díky této službě můžete na mapě prohlížet fotografie z celého světa. Fotky přidávají sami uživatelé.

Latitude – Můžete sdílet svou pozici a podívat se na pozici svých přátel, abyste věděli, kde se právě nachází.

Média:
YouTube – Webová stránka, která slouží ke sdílení videí. Tato stránka byla zakoupena v roce 2006, necelé dva roky po jejím vzniku, společností Google za 1,65 miliardy dolarů [9].

Knihy – Můžete vyhledávat ty knihy, které jsou vhodné pro Vaše účely a na dané téma objevovat nové.

Obrázky – Hledat také můžete v obrovském množství obrázků.

Domácnost a kancelář:
Gmail – Rychlý, intuitivní a zábavný e-mail, který filtruje většinu nevyžádané pošty, pro větší pohodlí uživatele.

Kalendář – Kalendář, díky kterému si můžete plánovat svůj denní rozvrh a sdílet své události s přátel. Právě na kalendář se zaměříme v naší práci.

Picasa – Nástroj pro úpravu fotek, který nabízí mnoho efektů. Umožňuje vytvářet také koláže a filmy.

SketchUp – Služba SketchUp umožňuje rychlé a snadné vytváření 3D modelů. Díky snadné tvorbě modelů jej využívá mnoho lidí.

Dokumenty – Vytvářejte dokumenty, tabulky a prezentace přímo ve svém prohlížeči bez nutnosti desktopového softwaru.

Překladač – Můžete překládat texty a webové stránky do mnoha světových jazyků.

Weby – Jednoduše a rychle můžete vytvořit webové stránky.

Zprávy a publikování:
Blogger – Pomocí tohoto nástroje, můžete vytvořit svůj vlastní blog a publikovat své myšlenky.

Zprávy – Procházení zpráv z různých zpravodajských zdrojů.

Vyhledávání blogů – Vyhledávat můžete také mezi blogy.

Skupiny – Spojte se s lidmi a komunikujte s nimi podle skupiny z určité kategorie pomocí e-mailu.

Reader – Můžete prohlížet své oblíbené stránky všechny na jednom místě.

Specializované vyhledávání:
Scholar – Specializované vyhledávání mezi vědeckými pracemi a články.

Vlastní vyhledávání – Umožňuje vytvoření vyhledávače podle svých představ.

Upozornění – E-mailem můžete odebírat novinky s tématy dle Vašeho výběru.

Reklamní programy:
AdWords – Pomocí této služby, můžete propagovat svou firmu nebo produkt prostřednictvím reklamy.

AdSence – Díky AdSence vyděláte na svých webových stránkách, zobrazením banneru s reklamou.

5.1.5. Google API

Google nabízí ke svým službám také API (Aplicattion Programming Interface) neboli rozhraní pro programování aplikací, díky němuž můžeme přistupovat do jejich služeb ze své aplikace. Pokud máte své stránky, můžete je rozšířit o nový obsah a nové funkce. Například pokud chcete umístit na své stránky video, tak ho nemusíte nahrávat přímo na svůj web, ale využít k tomu server http://www.youtube.com a poté na své stránky vložit pouze odkaz. API je vytvořené například pro mapy – Google Maps API, pro službu AdWords – Google AdWords API a podobně. API pro kalendář, které využiju v mojí aplikaci se jmenuje Google Calendars API and Tools. Všechny dostupné API naleznete na stránce http://code.google.com/intl/cs-CZ/more/ [10].

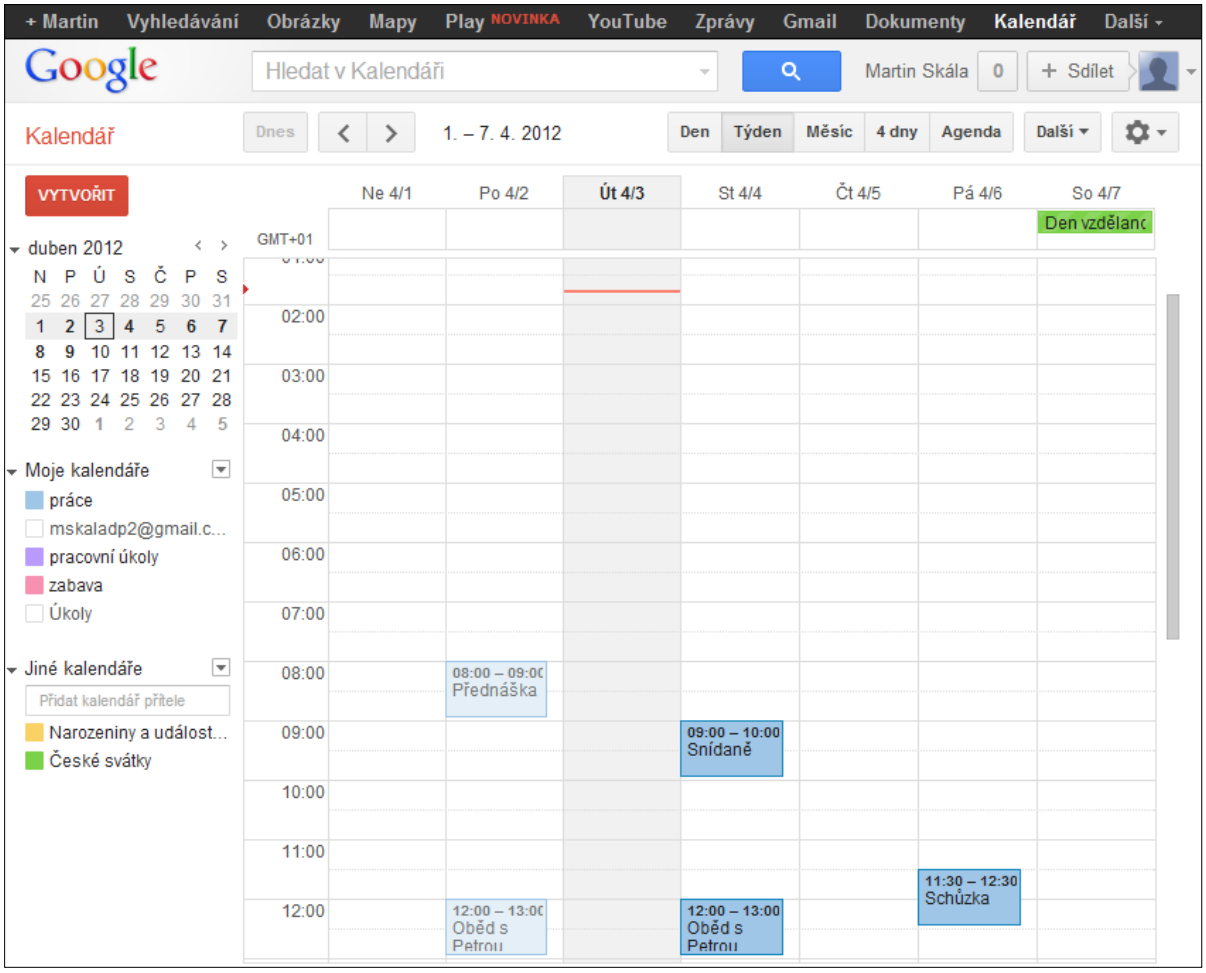
5.2. Google API pro kalendář

Jak již bylo řečeno, pro mojí aplikaci použiju API pro kalendář - Google Calendars API and Tools. Kompletní dokumentace k tomuto rozhraní, o jeho třídách a funkcích, je k dispozici na webové stránce https://developers.google.com/google-apps/calendar/. Zde se uživatel dozví o datové struktuře rozhraní, jak napsat první aplikaci a ukázky použití.

5.2.1. O kalendáři

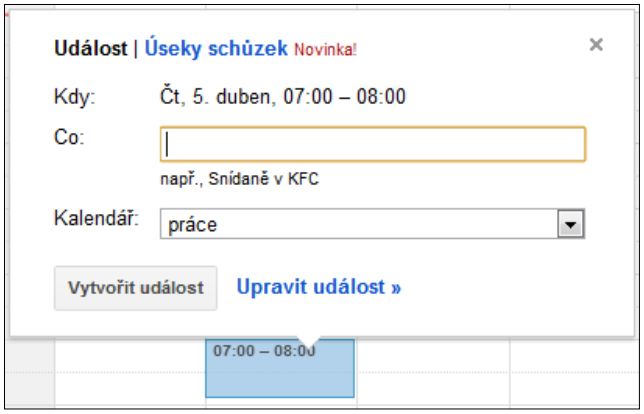
Nejprve si řekneme něco o samotném kalendáři, jak se s ním pracuje, jaké má funkce a jak se vytváří události.

Kalendář naleznete na adrese https://www.google.com/calendar. Pokud chcete používat kalendář, musíte být zaregistrováni a k tomu potřebujete Google účet. Pokud ho již máte, můžete vstoupit rovnou do kalendáře. Pokud účet nemáte, na domovské stránce kalendáře se můžete registrovat. Po přihlášení už vás čeká samotný kalendář a nic nebrání jeho používání [11].



Obr. 3: Ukázka z kalendáře.

Základní činností je přidání události. K vytvoření události stačí kliknout kamkoliv do časového pole kalendáře a do vyskakovacího okna vyplnit název události. Pokud událost trvá déle, stačí kliknout a přetáhnout okraj až na konec. Kliknutím na vytvořenou událost ji můžeme upravit a nastavit další vlastnosti. Například místo konání, popis a třeba také můžeme událost barevně označit, abychom ji rozeznali od ostatních [11].



Obr. 4: Vytvoření události.

Jiné možnosti práce s kalendářem jsou mazání událostí, tisknutí kalendáře, nebo vytvoření více kalendářů. To je vhodné například k tomu, pokud chceme oddělit kalendář, kde máme pracovní záležitosti a kalendář, kde máme osobní schůzky a zájmy.

Další důležitou funkcí, je sdílení kalendáře. Toto také využiju v mojí práci, kde je sdílení důležité kvůli možnosti ovládání kalendáře ostatními uživateli. Kalendář můžeme sdílet buď s určitými uživateli a nebo ho udělat veřejný a tudíž ho sdílet se všemi [11].

Pokud chceme kalendář sdílet, klikneme na vybraný kalendář a dáme „Sdílet tento kalendář“. Zde můžeme kalendář udělat veřejným a nebo pouze přidat uživatele, pro které je tento kalendář viditelný. Také můžeme nastavit práva uživatele, kterého jsme přidali. Na výběr máme tyto možnosti nastavení oprávnění:

- Provádět změny a spravovat sdílení
- Provádět změny událostí
- Zobrazit všechny podrobnosti událostí
- Zobrazit informace o dostupnosti (Skrýt podrobnosti)

K mému účelu poslouží první dvě možnosti. Resp. aby můj systém mohl číst události uživatele, přidávat je a upravovat, musí přidat účet mého systému (voice.calendar.zcu@gmail.com) a nastavit mu správná oprávnění.

Kalendář také můžeme ukázat svým přátelům ve formátu HTML nebo ho vložit na své webové stránky.

5.2.2. O API pro kalendář

Nyní si přiblížíme Google API pro kalendář. Google Calendar API umožňuje vytvořit klientské aplikace, které umí vytvářet události, upravovat a odstranit existující události a hledat je. Aktuální verze kalendáře je 3.0. Od této verze Google Calendar API se využívá datových objektů JSON (JavaScript Object Notation), což je platformou nezávislý způsob zapisování dat, který je určený pro přenos dat [12].

Pro zápis/editaci/mazání událostí se využívá se metody REST (Representational State Transfer). Jedná se o architekturu rozhraní, která je navržena pro distribuované prostředí. Pojem REST poprvé představil ve své disertační práci Roy Fielding, což je spoluautor protokolu HTTP. Proto také REST a HTTP má hodně společného. REST je datově orientován a určuje, jak se přistupuje k datům. Toto rozhraní je použitelné pro snadný a jednotný přístup ke zdrojům (resources). To mohou být data nebo stav aplikace a mají vlastní identifikátor URI. REST definuje čtyři základní metody pro přístup k nim. GET – základní metoda pro získání zdroje. POST – pro vytvoření dat. DELETE pro mazání dat. PUT – obdobné POST, ale známe předem URI [13].

API si můžeme stáhnout na stránce https://developers.google.com/google-apps/calendar/downloads. Na vývěr máte z několika programovacích jazyků:

.NET (beta)	Objective C (alfa)
Go (alfa)	PHP (beta)
Google Web Toolkit (alfa)	Python (beta)

Java (beta) Ruby (alfa)
JavaScript (alfa)

V závorce je uvedeno, zda se jedná o alfa nebo beta verzi. Pro jazyky Java, Python, PHP, .NET a Ruby, Google poskytuje příručku pro nastavení těchto knihoven, abychom je mohli používat. My využijeme balíček knihoven pro PHP [14].

5.2.3. Konfigurace API

Před používáním Google API je nutno nejprve projekt zaregistrovat a získat potřebné přístupové údaje. Každý kdo chce API používat, musí mít nejprve vytvořen účet od Googlu a poté může přistupovat do API konzole, kde nastaví název aplikace a vygeneruje potřebné klíče. Konzole je na adrese https://code.google.com/apis/console/ a máte zde přehled o svých aplikacích, můžete zde přepínat mezi aktivováním a deaktivováním přístupu, přidávání spolupracovníků, resp. přidat jim přístup do konzole a mimo jiné také sledovat reporty svých aplikací (počet odesílaných dotazů a podobně). Denní limit odeslaných dotazů pro kalendář je 10 tisíc. O navýšení limitu můžete požádat.

Potřebné údaje a klíče vygenerujeme v záložce API Access. Pro mojí aplikaci mám tyto údaje:

Product name:	Voice Calendar
Google account:	voice.calendar.zcu@gmail.com
Client ID:	1017441203665.apps.googleusercontent.com
Client secret:	snZ7Afm_2vjqNKxSirWIXdF8
Redirect URIs:	http://localhost/DP/voice/index.php
API key:	Abcdefghijklmn1234567890

API key nebo také někdy developer key lze vygenerovat zvláště pro serverovou aplikaci a pro aplikaci běžící v prohlížeči. V tabulce viz výše, byl z bezpečnostních důvodů nahrazen jinými znaky.

Další důležitou věcí, která je potřeba udělat, je sdílet kalendář uživatele, aby do něj měl přístup uživatel voice.calendar.zcu@gmail.com. Poté můžeme přistupovat ke kalendáři uživatele právě přes tento náš účet.

5.2.4. Ukázka API

Nakonec si ukážeme, jak takový skript, s přístupem do kalendáře může vypadat. K testování na lokálním počítači byl využit software XAMPP, kde bylo nutno povolit knihovnu cURL, které je potřebná ke komunikaci skriptu s kalendářem. To se provádí v souboru php.ini. V ukázkovém zdrojovém kódu si ukážeme, jak proběhne vytvoření klienta, identifikace, a vypsání událostí určitého uživatele (ID jeho kalendáře - mskaladp@gmail.com, který má nastaveno sdílení kalendáře s uživatelem voice.calendar.zcu@gmail.com.

```php
<?php
// připojení potřebných souborů API
require_once 'google-api-php-client/src/apiClient.php';
require_once 'google-api-php-client/src/contrib/apiCalendarService.php';
session_start();

// vytvoření nového klienta
$client = new apiClient();
$client->setApplicationName("Voice Calendar");

// nastavení identifikačních údajů klienta
$client->setClientId('1017441203665.apps.googleusercontent.com');
$client->setClientSecret('snZ7Afm_2vjqNKxSirWIXdF8');
$client->setRedirectUri('http://localhost/DP/voice/index.php');
$client->setDeveloperKey('Abcdefghijklmn1234567890');

if (isset($_GET['logout'])) {
  unset($_SESSION['token']);
}

if (isset($_GET['code'])) {
  $client->authenticate();
  $_SESSION['token'] = $client->getAccessToken();
  header('Location: http://'.$_SERVER['HTTP_HOST'].$_SERVER['PHP_SELF']);
}

if (isset($_SESSION['token'])) {
  $client->setAccessToken($_SESSION['token']);
}

// vytvoření připojení
// pokud přípojení není ověřeno, připojení proběhne přes odkaz
if ($client->getAccessToken()) {
  $_SESSION['token'] = $client->getAccessToken();
} else {
  $authUrl = $client->createAuthUrl();
  print "<a class='login' href='$authUrl'>Connect Me!</a>";
}

// vytvoření nové služby kalendáře
$service =  new apiCalendarService($client);

// získání událostí s kalendáře mskaladp@gmail.com
$calendar = $service->events->listEvents('mskaladp@gmail.com');

// vypsání událostí
print "<h1>Event List</h1><pre>".print_r($calendar, true)."</pre>";
?>
```

5.3. Alternativní způsob přístupu

Další možností jak psát aplikace s využitím služeb Google jsou Google Data Client Libraries, které umožňují přístup k oblíbeným API. Pro každý jazyk tyto knihovny poskytují nástroje a abstraktní vrstvu, díky níž můžete vytvářet dotazy a používat data z odezvy, bez nutnosti vytváření HTTP požadavků. Každá klientská knihovna poskytuje třídy, které odpovídají prvkům a datovým typům, které využívá API. Google podporuje klientské knihovny pro jazyky Java, JavaScript, .NET, PHP, Python a Objective C [15].

5.3.1. O Zend Frameworku a knihovně Zend Gdata

Pro PHP je tato knihovna dostupná jako knihovna Zend Frameworku, proto si nejprve něco povíme o něm. Zend Framework je webový aplikační framework, implementovaný v PHP 5 a licencovaný pod New BSD licence. Je objektově orientovaný a má otevřený zdrojový kód. Je vyvíjen s ohledem na snadný vývoj webových aplikací. Využívá modulární architektury, která umožní při psaní aplikací používat jen ty komponenty, které jsou potřeba. Částečné závislosti však mezi některými komponentami existují. Zend Framework v sobě zahrnuje komponenty pro MVC (Model-view-controller) architekturu, což je třívrstvá architektura, která umožňuje od sebe oddělit prezentační, doménovou a datovou vrstvu. Dále v sobě tento framework zahrnuje komponenty pro autorizaci, autentifikaci, cache, validátory pro uživatelská data a mnoho dalšího [16].

Jedna z knihoven Zend Frameworku se jmenuje Zend Gdata a slouží právě k přístupu ke službám Google. Tuto knihovnu můžeme stáhnout na adrese http://code.google.com/intl/cs/apis/gdata/docs/client-libraries.html, kde nalezneme i kompletní dokumentaci. Stažením této knihovny už nepotřebujeme samotný Zend Framework, protože všechny potřebné soubory jsou přiložené.

5.3.2. Ukázka Zend Gdata

Nyní si ukážeme zdrojový kód, pro přidání události do kalendáře s použitím Zend Gdata. Přístup do kalendáře je zajištěn pomocí Google uživatelského jména a hesla.

```
<?php
// Připojení potřebných souborů a načtení tříd
require_once 'Zend/Loader.php';
Zend_Loader::loadClass('Zend_Gdata');
Zend_Loader::loadClass('Zend_Gdata_AuthSub');
Zend_Loader::loadClass('Zend_Gdata_ClientLogin');
Zend_Loader::loadClass('Zend_Gdata_HttpClient');
Zend_Loader::loadClass('Zend_Gdata_Calendar');


// funkce pro přidání události
function createEvent ($client, $title, $desc, $where,
```

```
    $startDate, $startTime, $endDate, $endTime, $tzOffset)
{
  $gc = new Zend_Gdata_Calendar($client);
  $newEntry = $gc->newEventEntry();
  $newEntry->title = $gc->newTitle(trim($title));
  $newEntry->where  = array($gc->newWhere($where));

  $newEntry->content = $gc->newContent($desc);
  $newEntry->content->type = 'text';

  $when = $gc->newWhen();
  $when->startTime = "{$startDate}T{$startTime}:00.000{$tzOffset}:00";
  $when->endTime = "{$endDate}T{$endTime}:00.000{$tzOffset}:00";
  $newEntry->when = array($when);

  $createdEntry = $gc->insertEvent($newEntry);
  return $createdEntry->id->text;
}


// vytvoření klienta
$service = Zend_Gdata_Calendar::AUTH_SERVICE_NAME;
$client = Zend_Gdata_ClientLogin::getHttpClient($user, $pass, $service);

// vytvoření událost
$id = createEvent($client, "Oběd", "Obědvám s Petrem", "V Plzni", "2012-04-
04",
    "12:00", "2012-04-04", "13:00", "+01");
print "Event created with ID: $id\n";
?>
```

5.4. Moje volba

Při tvorbě mojí aplikace jsem dlouho pracoval s klasickým Google API. Bohužel se ale vyskytovaly problémy s připojením a autorizací uživatele. Jelikož moje aplikace je hlasová, byl by poté problém s některými nedostatky API. Proto jsem se nakonec rozhodl využít Zend Gdata.

Díky využití Zend Gdata odpadlo nastavování Google API, registrace projektu a podobně, jak bylo popsáno výše. Abych mohl využít Zend Gdata v našem kalendáři, napsal jsem skript, který obsahuje všechny důležité funkce, které jsou potřeba k užívání mého systému.

5.4.1. Skript vgc_library.php

Skript vgc_library.php obsahuje všechny funkce, které jsou potřeba v mém systému. Jelikož uživatel může číst události, vkládat nové a při čtení události modifikovat nebo mazat, vytvořil jsem tyto funkce na:

- Získání událostí od určitého času
- Vytvoření nové události
- Smazání událost
- Získání události podle ID události
- Modifikace události

Nyní si funkce blíže přiblížíme:

```
function vgc_read($client, $id_calendar, $from, $to)
```

$client – Proměnná, která obsahuje informace o klientovi. Vytvoříme ji pomocí uživatelského jména a hesla, s kterým se přihlašujeme k námi vytvořenému účtu voice.calendar.zcu@gmail.com.

$id_calendar – Zde je uloženo ID kalendáře uživatele, který je registrován do systému a právě systém používá.

$from – Čas od kterého začne čtení. Pokud není uvedeno, čas se rovná aktuálnímu času.

$to – Čas, do kterého se mají číst události. Pokud není vyplněno, rovná se aktuálnímu času plus jednomu roku.

Tato funkce slouží k získání událostí. Vrací pole s událostmi.

```
function vgc_create($client, $id_calendar, $title, $where, $description,
                                $from, $to)
```

$client a **$id_calendar** – Proměnné, které obsahují informace o klientovi a o kalendáři.

$title – Zde se předává nový název události.

$where – Tato proměnná předává nové místo konání události.

$description – Zde můžeme vyplnit nový popis události.

$from a **$to** – slouží k určení začátku a konce konání akce. Pokud začátek není uveden, vezme se aktuální čas plus jedna hodina. Pokud není uveden koncový čas, vezme se počáteční plus jedna hodina.

Funkce vgc_create() slouží k novému vytvoření události.

```
function vgc_delete($client, $event_id)
```

$client – Proměnná, která obsahuje informace o klientovi.

$event_id – Zde předáváme ID události, kterou chceme smazat.

Tato funkce slouží k mazání událostí.

```
function vgc_getEvent($client,  $eventId)
```

$client a **$event_id** – Proměnné, které obsahují informace o klientovi a o události.

Tuto funkci využiju k získání určité události podle jejího ID. Funkce vrací pole s jednou událostí. Pokud událost neexistuje, vrátí NULL a vypíše výjimku.

```
function vgc_updateEvent($client, $eventId, $q, $newData)
```

$client a **$event_id** – Proměnné, které obsahují informace o klientovi a o události.

$q – Tato proměnná obsahuje číslo od 1 do 5. Podle toho, jaké je zde číslo, systém vybírá, která hodnota události se bude měnit. Pro $q = 1 se mění název události, pro $q = 2 se mění místo konání, $q = 3 se mění popis, $q = 4 začátek konání události a pro $q = 5 se mění konec události.

$newData – Zde se předává nová hodnota dat, která se mění.

Pomocí této funkce se mění existující události.

Díky těmto pěti funkcím systém může pracovat s kalendářem přesně, jak je potřeba k tomu, abych mohl vytvářet události, modifikovat, číst a také mazat.

6. Gramatiky a parsování

Jednou z nejdůležitějších součástí mojí aplikace, jsou gramatiky a parsování vstupních dat. Pomocí gramatiky totiž systém určuje, co je možné vyslovit a jakým způsobem by měl uživatel komunikovat se systémem, aby systém mohl správně rozpoznat požadavky uživatele a vykonat je. Díky gramatice uživatel předá systému řetězec požadavků a systém musí tento vstup dát rozpoznat, správně „pochopit“ informaci. K tomu slouží parsování vstupní informace, kdy systém určí, která část promluvy obsahuje jaké data, se kterými dál pracuje.

6.1. O gramatikách

V mém systému je potřeba, aby uživatel mohl říci prakticky cokoliv a systém tuto informaci zpracoval správně. Moje aplikace slouží k ovládání kalendáře a od toho se také odvíjí navržení gramatiky (gramatik).

Je nutné, aby ovládání bylo co nejjednodušší a co nejintuitivnější pro uživatele. Možná promluva by také měla být co nejpřirozenější a odpovídat požadavkům toho co by uživatel mohl chtít vykonat. Dále je třeba se zamyslet nad tím, že uživatel nemusí najednou vyslovit všechny důležité informace (například pro vložení nové události) a potom by se tedy systém měl na další důležité informace zeptat.

Jak již bylo zmíněno, moje aplikace slouží k ovládání kalendáře, proto je třeba v promluvě vyslovit akci, která se bude vykonávat, popřípadě ji doplnit o časový údaj, například na kdy se má vložit nová událost nebo do kdy se bude konat. Také se promluva může doplnit o další informace jako je název akce, která se vkládá do kalendáře.

Použil jsem ESGF gramatiky. Více o těchto gramatikách na http://voice.zcu.cz/VoiceXML/ .

6.2. Co je možné říci?

Nyní si ukážeme možné promluvy, které uživatel může vyslovit, aby systém zareagoval správně. V mém systému jsou implementovány dvě aplikace, jedna na přidávání záznamu do kalendáře a jedna na jejich čtení. Při čtení je možné také záznamy modifikovat nebo je smazat. Pokud je ale chce modifikovat nebo smazat, musí je nejdříve číst. Je dobré, aby uživatel mohl komunikovat přirozeným jazykem, proto také pro čtení může vyslovit více slov, který mají stejný význam (číst, čti, přečíst).

Možné promluvy pro čtení událostí:

„Číst.“
„Přečti záznamy.“
„Čti události.“
„Přečíst záznamy od zítra.“
„Čti zejtra od desíti.“
„Chtěl bych prosím přečíst události od 25. května od 12 hodin 35 minut.“

Možné promluvy pro vytvoření událostí:

„Vytvoř.“
„Vlož novou událost.“
„Zapiš na zítra na 10 hodin 30 minut. Schůzka s Karlem.“
„Přidat záznam na 30. května 13:30.“
„Vytvořit událost. Oběd s Marcelou.“
„Chtěl bych prosím přidat událost od 25. května od 12 hodin 35 minut do 25. května do 14 hodin 35 minut. Oběd se šéfem.“

6.3. Jednotlivé gramatiky

Aby bylo možné vyslovit dané promluvy (a mnohé další), bylo třeba vyslovit několik dílčích gramatik, které obsahují určitá slova (a jejich tagy).

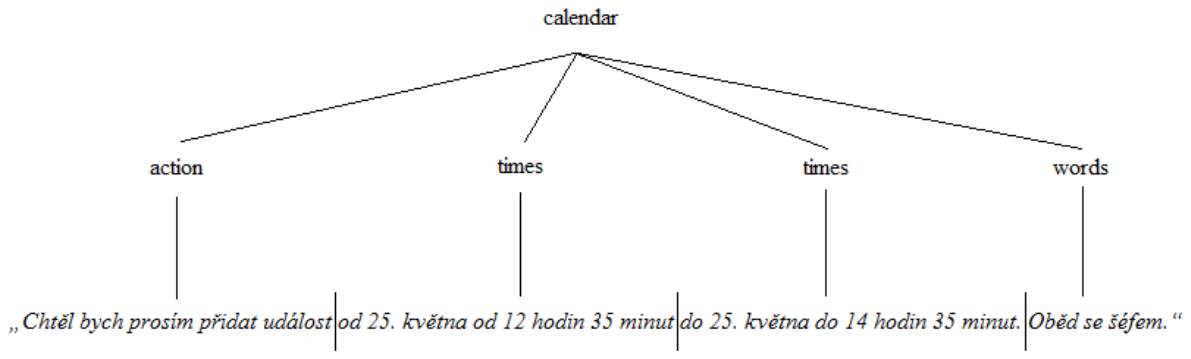
Promluva by se dala rozdělit na několik významových částí. Na vyslovení druhu požadované akce, na vyslovení časového údaje a na obecnou gramatiku, která by měla obsahovat co největší množinu slov, aby vyhověla názvům událostí uživatele.

6.3.1. Gramatika Calendar

Gramatika Calendar byla vytvořena ke sdružení všech dílčích gramatik, aby odpovídala uživatelskému vstupu na začátku dialogu. V této gramatice je tedy možné vyslovit promluvy, které byly uvedeny, viz výše. Potřebujeme tedy vyslovit akci, dále můžeme vyslovit dva časové údaje a obecnou promluvu. Všechny tyto údaje jsou v promluvě nepovinné. Nepovinnost údajů určují hranaté závorky (tedy daná část může nebo nemusí být vyslovena). Tag <VIRTUAL> vkládá do gramatiky uzel, což při načtení gramatiky sníží počet hran, ovšem rozpoznání vyslovených slov poté může být nepřesnější.

Zapsání gramatiky:

```
public <calendar> = (<BEGIN_GRAMMAR> [<action>] <VIRTUAL> [<times>]
            <VIRTUAL> [<times>] <VIRTUAL> [<words>] <END_GRAMMAR>);
```



Obr. 5: Strom gramatiky Calendar.

6.3.2. Gramatika Action

Dále si probereme jednotlivé gramatiky, ze kterých se skládá gramatika Calendar. První takovou je gramatika Action.

Gramatika Action slouží k předávání slov, které slouží k rozpoznání akce, která se bude vykonávat. Jak již bylo uvedeno, můžeme na začátku vybrat mezi čtením a vytvořením události. Tato gramatika obsahuje tagy, díky nimž pak může systém akci rozpoznat.

Kromě slova, které určuje, co se bude provádět, můžeme říci mnoho slov předtím i potom.

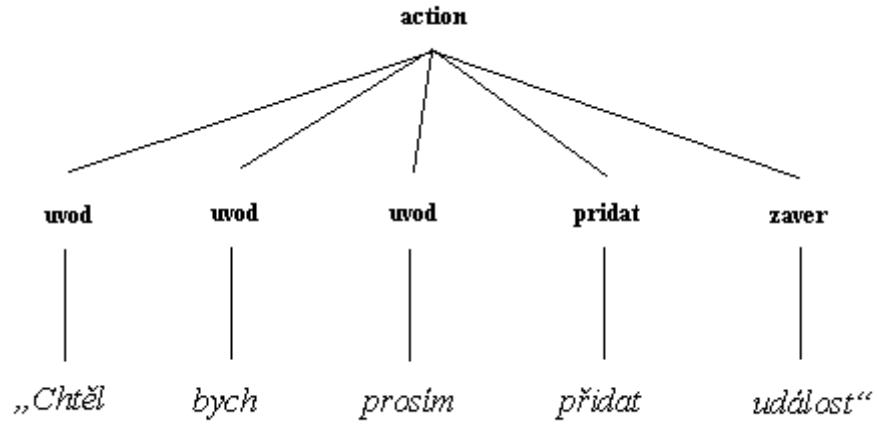
Zapsání gramatiky:

```
public <action> = ((<uvod>* <VIRTUAL> <precist> <VIRTUAL> <zaver>*) |
                           (<uvod>* <VIRTUAL> <pridat> <VIRTUAL> <zaver>*));
```

Tento zápis určuje, že můžeme vyslovit slova z kategorie `<precist>` nebo (nebo se označuje znakem „|“) `<pridat>`. Před a za těmito slovy mohou být další slova, žádné nebo jich více (označuje znak „*“), která dělají promluvu přirozenější, ale pro porozumění systému jsou nepodstatná.

V `<precist>` jsou slova významově stejná, ale jinak zapsána. Jsou tam například „přečti“, „čti“, „číst“ a podobně. V `<pridat>` jsou slova jako „vytvořit“, „zapsat“ a tak dále. Ostatní slova jako například „chtěl bych“, „záznam“, „události“ a mnoho dalších, jsou zapsány v `<uvod>` a `<zaver>`. Díky této gramatice pak můžeme říci třeba větu *„Chtěl bych prosím přidat událost“*.

Gramatika obsahuje tagy, a pokud je vysloveno něco z `<precist>` předá se rozpoznávači slovo „read“. Pokud je vysloveno něco z `<pridat>` , předá se rozpoznávači „create“. Okolní slova se nepředávají.



Obr. 6: Strom gramatiky Action.

6.3.3. Gramatika Times

Gramatika Times slouží k rozpoznávání časových údajů. Správné rozpoznání časového údaje je důležité například pro vkládání záznamů nebo upravování jejich začátku resp. konce. U této gramatiky je důležité zvolit vhodné tagy pro jednotlivá slova, aby se promluva dala co nejsnadněji rozpoznat.

V naší gramatice určujeme datum a čas. Vždy je nutné uvést datum, den a to buď absolutně (24. května) nebo relativně (zítra). Dále je možné uvést čas, hodiny a minuty. Pokud hodiny ani minuty nejsou uvedeny, pracuje se s časem jako by byl uveden 00:00.

Mezi časovými údaji se také nacházejí předložky, které poté v rozpoznávání hrají také důležitou roli. Předložek může být několik, „v“, „na“, „od“, „do“ a tak dále.

Čeština je velmi rozmanitý jazyk plný nespisovných slov, proto aby systém pracoval co nejlépe, jsem se snažil tato slova zahrnout do gramatiky. Další změna slova je s danou předložkou před slovem. Toto jsem se také snažil zachytit v gramatice.

Proto například pro 23 minut můžeme mít několik slov se stejným významem:

„dvacettři“
„dvacetitří““
„dvacetitřech“
„dvacátýtřetí“
„dvacátétřetí“

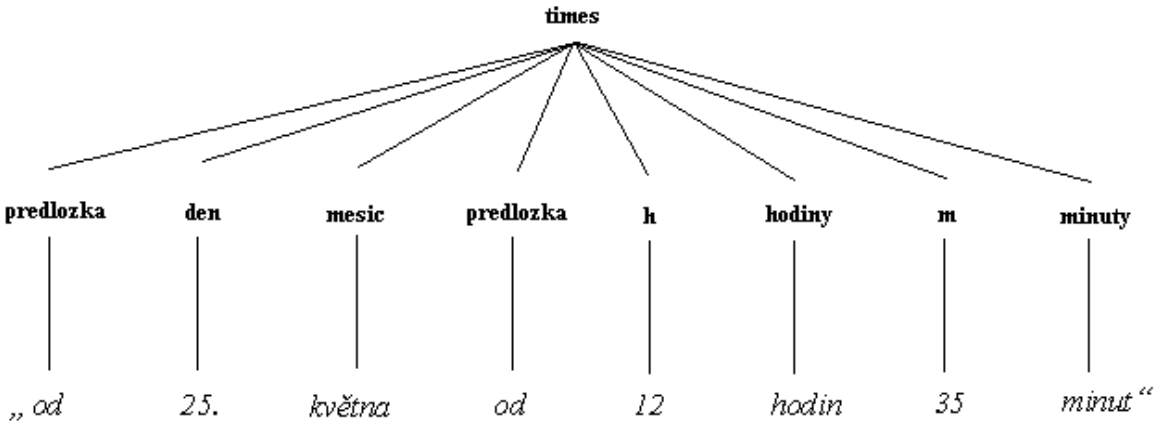
Tato slova mají stejný význam, a proto mají společný tag. Vyslovením jednoho z nich se předá ke zpracování číslovka „23“. Stejně tak jsou otagovány i hodiny, měsíce, předložky a podobně. Slova jako „hodin“ nebo „minut“ jsou vypouštěna.

Zapsání gramatiky Times vypadá následovně:

```
public <times> = [<predlozka>]
              [(<rel_den> | <den> | (<den> <VIRTUAL> <mesic>))]
              <VIRTUAL>
              [<predlozka>]
              <VIRTUAL>
              [((<h> <VIRTUAL> [<hodiny>] <VIRTUAL><m> <VIRTUAL> [<minuty>])
               | (<h> <VIRTUAL> [<hodiny>]))];
```

Podle této gramatiky můžeme vyslovit volitelně předložku, poté relativní den, nebo den v měsíci a nebo den i s měsícem. Poté můžeme vyslovit další předložku a volitelně hodiny a minuty a nebo jenom hodiny.

Další vylepšení, které by mohlo být, by bylo určování času pomocí dělení celku hodin. Tím se myslí například „půl páté odpoledne“, „čtvrt na čtyři ráno“ a takto to vytvořit pro každou hodinu a také pro nespisovné varianty těchto časových určení. Toto v této práci implementováno není.



Obr. 7: Strom gramatiky Times.

6.3.4. Gramatika Words

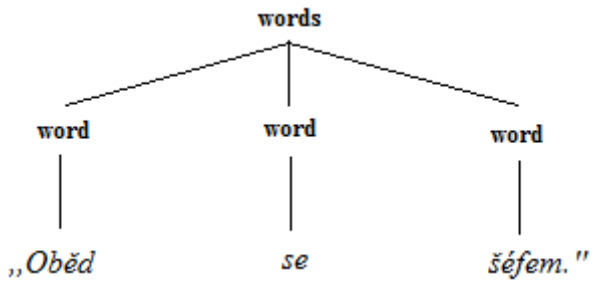
Gramatika Words slouží k vyplňování názvu události, míst konání a popisu událostí. Na tato místa uživatel může chtít vložit prakticky cokoliv, proto by taky gramatika měla obsahovat co nejvíce slov. V ideálním případě by obsáhla všechna slova, která člověk může vyslovit. Což samozřejmě není možné.

Jelikož gramatika bude velká ale i tak omezená, mohl by se vyskytnout případ, kdy uživatel chce zapsat slovo, které v gramatice není. Stejně tak uživatel může používat slova, která nikdo jiný neříká, a tudíž by se nedostala do gramatiky. To můžou být například přezdívky. Za tímto účelem uživatel má možnost si přidat do gramatiky vlastní slova (pomocí webové části naší aplikace).

Gramatika Words neobsahuje také žádné tagy, co je vysloveno, to se předá ke zpracování. Slova, která patří do gramatiky Words poté právě poznáme podle toho, že neobsahují tagy.

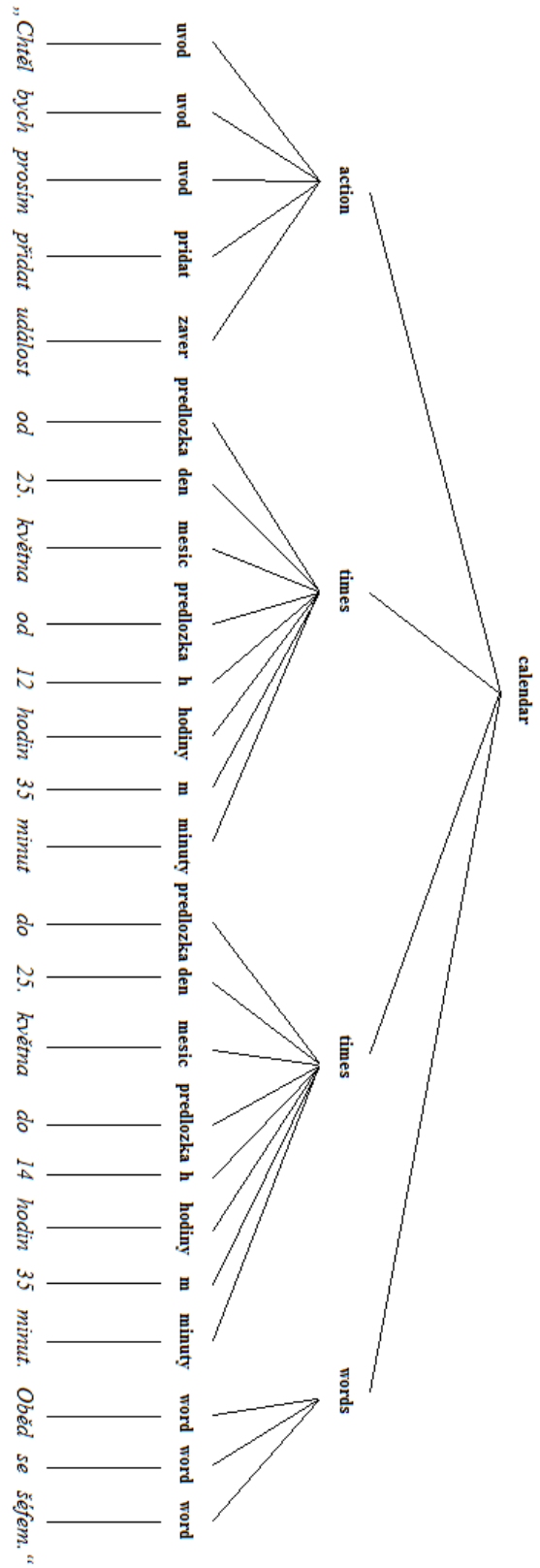
Z gramatiky Words pak uživatel může vyslovit žádné nebo několik slov za sebou, které dávají dohromady například celý název události.

V mojí práci nebylo možné vytvořit tak obsáhnou gramatiku, proto gramatika Words obsahuje jen několik desítek slov.



Obr. 8: Strom gramatiky Words.

Celý strom gramatiky pro větu *„Chtěl bych prosím přidat událost od 25. května od 12 hodin 35 minut do 25. května do 14 hodin 35 minut. Oběd se šéfem.“* pak vypadá následovně:



Obr. 9: Strom pro celou větu.

6.4. O parsování

Pomocí gramatik, předá uživatel programu řetězec slov, které je nutno dále zpracovávat. Řetězec obsahuje buď jednotlivá slova, nebo jejich tagy. Podle toho jak jsem navrhl vstupní gramatiku, bude probíhat parsování. Čím lépe gramatiku vytvořím, tím snazší bude pak danou promluvu rozpoznat. Parsování je v podstatě rozdělení řetězce na části, které obsahují určitou informaci, a tuto informaci z toho získáme.

Promluva uživatele:
„Chtěl bych prosím přidat událost od 25. května od 12 hodin 35 minut do 25. května do 14 hodin 35 minut. Oběd se šéfem.“

Řetězec, který se předá ke zpracování:
create PREDLOZKA 25. May PREDLOZKA 12 35 PREDLOZKA 25. May PREDLOZKA 14 35 Oběd se šéfem.

6.5. Parsování vstupu

Vstupní gramatika na začátku dialogu je Calendar. Z toho vyplývá, že může být vyslovena akce, dva časové údaje a jakékoliv další slova.

Ke zpracování byl vytvořen skript parser.php, který obsahuje funkci parseAll(), kde vstupem, je řetězec s tagy a výstupem jednotlivé dílčí části.

```
list($action, $title, $timeStart, $timeEnd) = parseAll($vstup);
```

V promluvě se rozpoznává nejprve první slovo. Pokud se slovo rovná *read* nebo *create*, předá se toto slovo do proměnné *$action*. Pokud je první slovo jiné, nepředá se nic, protože uživatel nevyslovil akci a bude vyzván znovu k vyslovení požadavku.

Další slova jsou možné časové údaje. Prochází se promluva postupně od druhého slova a pokud se slovo rovná *PREDLOZKA* nebo je to číslo s tečkou, označí se jako čas. Čas také může být slovo obsahující *day* (tj. například 1day, což je tag pro slovo „zítra“). Dále jako čas se označí anglicky psané názvy měsíců v roce a číslovky (numerický datový typ), které zastupují hodiny a minuty. Slova se přiřazují k časovému údaji, dokud vyhovují předchozím podmínkám, pokud ne, další promluva se označí jako prostá věta a uloží se do proměnné *$title*. To zaručuje, že v názvu události se mohou objevit stejná slova jako v časovém určení. Nesmí ovšem tímto slovem začínat daný název události, protože by se přiřadil k určení času. Název události tedy může být *„Sraz na ulici 5. května“*. Nesmí být ovšem *„5. automobilový sraz“*.

Jelikož mohou být určeny dvě promluvy z gramatiky Times za sebou, pro určení času začátku události a pro určení konce události, je třeba rozlišit, zda posloupnost časových údajů obsahuje dvě určení času nebo pouze jedno. Bylo to vyřešeno tak, že pokud se jedná o dvě určení času, v řetězci, po vyslovení hodin nebo minut následuje *PREDLOZKA*. A nebo

obsahuje dvakrát slovo pro den v měsíci. Například *„od prvního 12 hodin do druhého 12 hodin“*.

6.6. Parsování času

Nyní se zaměříme na parsování časového údaje. V mojí aplikaci pracujeme s časem ve formátu DATE_ATOM. Vstupem do parseru jsou tagy z gramatiky Times a ty je potřeba do tohoto formátu převést.

Uživatel tedy řekne:

„Od 25. Května od 12 hodin 35 minut“

Programu se předá vstup s tagy:

PREDLOZKA 25. May PREDLOZKA 12 35

A je potřeba získat čas v DATE_ATOM:

2012-05-25T12:35:00+02:00

Nejprve si systém rozdělí řetězec s tagy do pole (dále POLE1). Poté vytvoří další pole, které určuje význam slov (dále POLE2). Prochází postupně POLE1 a rozpoznává význam slov a vkládám to do POLE2. Tedy například pokud v POLE1 na 3. pozici je název měsíce (May), vloží se do POLE2 na 3. pozici `month`. Pro hodiny a minuty se vloží do POLE2 `time` a tak dále.

Pro příklad viz výše, vypadá POLE2 následovně:

PREDLOZKA day month PREDLOZKA time time

V tomto okamžiku se porovnává, zda se jedná o dva časové údaje nebo pouze o jeden jak je popsáno v kapitole 6.5.

Pro převod na DATE_ATOM se využije PHP funkce strtotime(), která slouží k vytvoření času z řetězce. Této funkci tedy můžeme předat slova a ona jej převede na datum a čas. Můžeme buď předat den relativně (+1day) nebo absolutně (25. May 2012). A dále k tomu připojit čas (12:30). Právě do této podoby je potřeba převést řetězec.

Nyní již nejsou potřeba slova PREDLOZKA, proto jsou odstraněna. Pokud je v řetězci den vložen relativně, vloží se ke zpracování a přidají se hodiny a minuty. Pokud hodiny nejsou uvedeny, předpokládá se, že je 00 hodin a stejně tak s minutami, pokud nejsou uvedeny, předpokládá se 00 minut. Pokud je den uveden absolutně, předá se jako den, měsíc a rok. Rok se doplní aktuální, a pokud měsíc chybí, doplní se také aktuální. Jak již bylo uvedeno. Pomocí funkce strtotime() se pak převede čas ze slova na časový formát. Abych byl přesný, funkce strtotime() vrací čas ve formátu timestamp, což je počet vteřin uplynulých od

1.1.1970. K převodu na DATE_ATOM využívám funkci Date(). Pokud do funkce strtotime vložím řetězec, který neumí zpracovat, vrátí 0, resp. datum 1.1.1970 00:00.

Příklad využití funkce strtotime():

```
2012-05-05T22:23:07+02:00 <= Date(DATE_ATOM,strtotime("+1day +2hour"))
2012-01-12T11:50:00+01:00 <= Date(DATE_ATOM,strtotime("12. January 2012 11:50"))
2012-01-12T00:00:00+01:00 <= Date(DATE_ATOM,strtotime("january 12"))
2012-05-04T10:50:00+02:00 <= Date(DATE_ATOM,strtotime("10 am 50 minute"))
2012-05-06T10:00:00+02:00 <= Date(DATE_ATOM,strtotime("2day 10:00"))
1970-01-01T01:00:00+01:00 <= Date(DATE_ATOM,strtotime("20. April 10:12"))
```

6.7. Parsování nevhodných promluv

Někdy se také může stát, že uživatel vysloví na začátku špatný příkaz, proto je dobré s tím počítat, aby systém nereagoval špatným způsobem. Nepřesnosti mohou být také způsobeny nepřesným rozpoznáním promluvy v interpretu, což může být způsobeno například tím, že se uživatel nachází v hlučném prostředí nebo má nedostatečný signál mobilního telefonu.

Důležité je, aby když uživatel chce vytvořit událost, aby se nestalo, že se vytvoří událost s úplně jiným názvem a v úplně jiný čas. To je zajištěno tím, že systém vysloví údaje, které při vytvoření rozpoznal a zeptá se uživatele, zda opravdu chce takovou událost vytvořit. V případě že uživatel zadá čas ve špatném formátu, parser tento čas ignoruje a systém se popřípadě znovu zeptá na časový údaj.

7. Webová aplikace

První věcí, která byla potřeba vytvořit k mojí aplikaci, byla webová stránka, pomocí níž by se mohl uživatel zaregistrovat do systému. Přístup do systému, mají pouze registrovaní uživatele, kteří vyplní své telefonní číslo. Zavoláním z toho to čísla, systém automaticky identifikuje volajícího. Registrovaní uživatelé také na této webové stránce mohou měnit svoje registrační údaje.

Další funkcí webového rozhraní je to, že se uživatelé mohou dozvědět užitečné informace o našem hlasovém rozhraní pro online kalendář, co to je, jak se používá, jak se ovládá a vše o registraci.

Poslední funkcí webového rozhraní je administrační část. Ta slouží ke správě uživatelů. Administrátor tak může upravovat informace o uživatelích nebo je smazat.

K vytvoření bylo použito HTML, PHP, CSS, JavaScript a MySQL.

7.1. Uživatelské role

Nepřihlášený uživatel - má možnost se registrovat a dozvědět se více informací o systému.

Přihlášený uživatel - přihlášený uživatel je ten, co se nejprve zaregistroval a poté se přihlásil do webové aplikace. Má stejné možnosti jako nepřihlášený uživatel a navíc může měnit svoje registrační údaje.

Administrátor - administrátor po přihlášení do administrační části, má možnost měnit údaje registrovaných uživatelů nebo tyto uživatele jednotlivě smazat.

7.2. Tabulka s uživateli

Tabulka s registrovanými uživateli byla uložena v MySQL databázi. Byly vytvořeny tyto buňky:

ID - identifikační číslo uživatele, nenulové a vložením uživatele do databáze se tato hodnota zvýší o jedničku.

Login - přihlašovaní jméno uživatele.

Heslo - heslo uživatele, v databázi je toto heslo z bezpečnostních důvodů zakryptováno.

Telefonní číslo - telefonní číslo, ze kterého bude uživatel volat do systému.

ID kalendáře - ID kalendáře, které je nutné, aby systém měl přístup do kalendáře uživatele.

Oslovení - oslovení slouží k uvítání uživatele.

Slova - slova, která uživatel často používá a neobsahuje je gramatika systému.

MySQL dotaz pro vytvoření této tabulky:

```
CREATE TABLE `vgc_users` (
  `id` INT UNSIGNED NOT NULL AUTO_INCREMENT ,
  `login` TEXT NOT NULL ,
  `heslo` TEXT NOT NULL ,
  `tel_cislo` TEXT NOT NULL ,
  `idcal` TEXT NOT NULL ,
  `osloveni` TEXT NOT NULL ,
  `slova` TEXT NOT NULL ,
   PRIMARY KEY ( `id` )
 ) ENGINE = MYISAM ;
```

7.3. Registrační formulář

K vyplnění tabulky s informacemi o uživatelích byl vytvořen registrační formulář, pomocí něhož uživatel tyto údaje vyplní. Byla v něm vytvořena tato pole:

Login – sem uživatel vyplní svůj zvolený login. Tato položka je povinná. Pokud je login špatně vyplněný tzn., že je pole buď prázdné, nebo zvolený login již existuje, objeví se při odeslání formuláře chybová hláška: „Login není platný nebo je již registrovaný!“

Heslo – sem uživatel vyplní zvolené heslo. Tato položka musí být vyplněná. Pokud je pole při odeslání prázdné, objeví se hláška: „Nesprávné heslo!“

Kontrola hesla – sem uživatel znovu vyplní heslo, které si zvolil. Slouží to pro kontrolu, aby si byl jistý, že heslo vyplnil správně. Při vyplnění, se pomocí JavaScriptu, vedle textového pole, průběžně zobrazuje, zda se hesla shodují nebo ne. Zda se hesla shodují, se samozřejmě také kontroluje při odeslání formuláře. Pokud se neshodují, objeví se hláška: „Hesla se neshodují!“ Tato položka je povinná.

Tel. číslo – zde uživatel vyplní své číslo, ze kterého pak bude volat do systému. Telefonní číslo musí obsahovat 9 číslic a je nutné ho vyplnit. Nesmí se také shodovat s žádným již registrovaným číslem. Pokud je tato položka vyplněna špatně, objeví se chybová hláška: „Telefonní číslo je ve špatném formátu nebo již je registrované!“

ID kalendáře – zde uživatel vyplní ID svého kalendáře, které jak již bylo řečeno, je nutné k čtení a zapisování událostí. Tato položka je povinná a tudíž nesmí být pole prázdné. Pokud je, objeví se při odeslání hláška: „ID kalendáře je ve špatném formátu!“

Oslovení – Sem uživatel vyplní svoje oslovení. Může sem vyplnit prakticky cokoliv například „Martine“, „pane Martine“ a podobně. Tato položka je nepovinná.

Obr. 10: Registrační formulář.

Vaše slova – Tato položka je také nepovinná a uživatel sem může napsat svá často používaná slova, která nejsou obsažená v gramatice. Přezdívky přátel a podobně. Slova musí být oddělená čárkou.

Pět plus čtyři je – kontrolní otázka, která musí být vyplněna. Pokud vyplněna není nebo je vyplněna špatně, objeví se hláška: „Kontrolní otázka je špatně!“

Reset – tlačítko „Reset“ vymaže všechny vyplnění pole.

Odeslat – slouží k odeslání formuláře ke kontrole údajů a zapsání do databáze.

Co to je? – u polí „ID kalendáře“ a „Vaše slova“ je odkaz „Co to je?“. Po kliknutí na tento odkaz se uživatel dozví o tomto textovém poli, co do něj vyplnit a kde získá ID kalendáře.

Všechny pole jsou také kontrolována, aby nebylo možné zde vyplnit kusy zdrojového kódu, který by pak mohl způsobit nesprávné fungování aplikace. Při odeslání formuláře s neplatnými údaji se objeví všechny potřebné chybové hlášky, popsané viz výše. Všechny údaje zůstanou vyplněni a uživatel tak může změnit pouze ty, ve kterých udělal chybu a nemusí celý formulář vyplňovat znovu. Pokud je formulář vyplněn správně, objeví se hláška: „Registrace proběhla úspěšně.“

7.4. Adresářová struktura

Nyní si ukážeme výpis souborů, přesněji adresářovou strukturu, které bylo nutno vytvořit k webové části Hlasového rozhraní pro online kalendář.

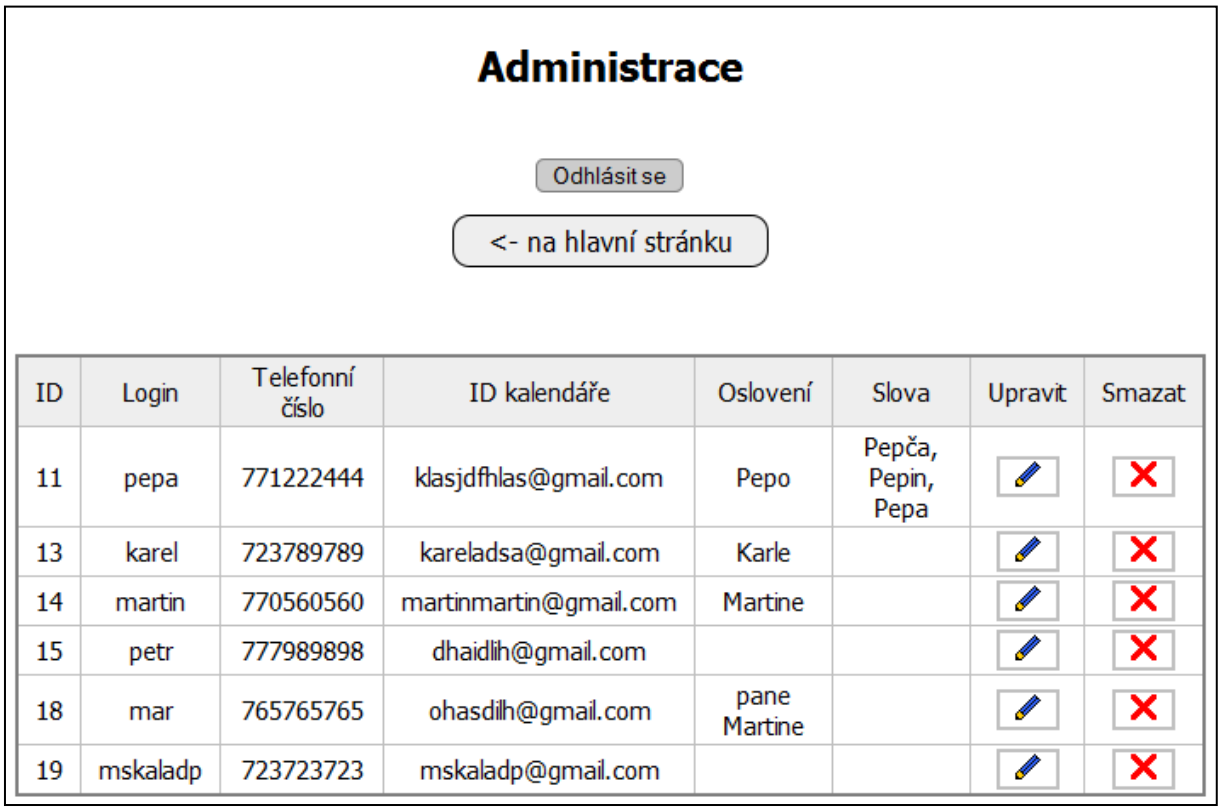
- **Kořenový adresář**
    - **script**
        - jquery-1.4.4.min.js
        - script.js
    - **source**
        - **screeny**
            - 1.gif
            - 2.gif
            - 3.gif
        - bg.jpg
        - delete.gif
        - edit.gif
        - icon.gif
        - ovladani.gif
    - **style**
        - style.css
    - admin.php
    - adminId.php
    - config.php
    - crypt.php
    - help_idcal.php
    - help_slova.php
    - index.php
    - podrobnosti.php
    - pre_reg.php

- reg.php
- setup.php

7.5. Popis souborů

Jednotlivé soubory a složky si popíšeme, jaké mají funkce v našem systému.

admin.php – administrační soubor, ve kterém administrátor může spravovat informace o uživatelích. Měnit jejich data nebo uživatele vymazat.



ID	Login	Telefonní číslo	ID kalendáře	Oslovení	Slova	Upravit	Smazat
11	pepa	771222444	klasjdfhlas@gmail.com	Pepo	Pepča, Pepin, Pepa		
13	karel	723789789	kareladsa@gmail.com	Karle			
14	martin	770560560	martinmartin@gmail.com	Martine			
15	petr	777989898	dhaidlih@gmail.com				
18	mar	765765765	ohasdilh@gmail.com	pane Martine			
19	mskaladp	723723723	mskaladp@gmail.com				

Obr. 11: Ukázka tabulky z administrační části.

adminId.php – kontroluje přihlašování údaje administrátora.

bg.jpg – obrázek, logo Googlu, které je na pozadí.

config.php – konfigurační soubor, ve kterém jsou uloženy přihlašovací údaje do databáze.

crypt.php – tento soubor obsahuje kryptovací algoritmus, kterého se pak využívá ke kódování a dekódování hesla, když se ukládá a čte z databáze nebo z cookies.

delete.gif – obrázek křížku na tlačítku při mazání uživatelů.

edit.gif – obrázek tužky na tlačítku při editaci uživatelů.

help_idcal.php – tento soubor obsahuje nápovědu o získání svého ID kalendáře. Tento soubor je zobrazí při kliknutí na „Co to je?“ v registračním formuláři u textového pole „ID kalendáře“.

help_slova.php – Tento soubor, stejně jako předchozí, obsahuje nápovědu, tentokrát k položce „Vaše slova“.

icon.gif – ikona zobrazující se v prohlížečích.

index.php – hlavní soubor, ze kterého se uživatel dostane do všech částí webové aplikace a dozví se všechny potřebné informace.

jquery-1.4.4.min.js [17] – soubor který obsahuje jQuery. jQuery je rychlá a stručná JavaScript knihovna, která zjednodušuje HTML dokument.

ovladani.gif – pomocný obrázek k zobrazení ovládání.

podrobnosti.php - soubor, ve kterém se uživatel dozví více o ovládání hlasového systému.

pre_reg.php – přeregistrační soubor, který maže cookies.

reg.php – soubor, který obsahuje registrační formulář, a pomocí něhož se provádí všechny kontroly, změna údajů a zapisování do databáze.

screeny – obsahuje obrázky k nápovědě.

script.js – obsahuje zdrojový kód JavaScriptu, který je použitý v aplikaci.

setup.php – slouží k úvodnímu vytvoření tabulky v MySQL databázi.

script – složka, která obsahuju skriptu JavaScriptu.

source – obsahuje použité obrázky.

style – složka style, obsahuje soubor style.css, ve kterém je CSS kód, který formátuje celý obsah webové aplikace.

8. Hlasová aplikace

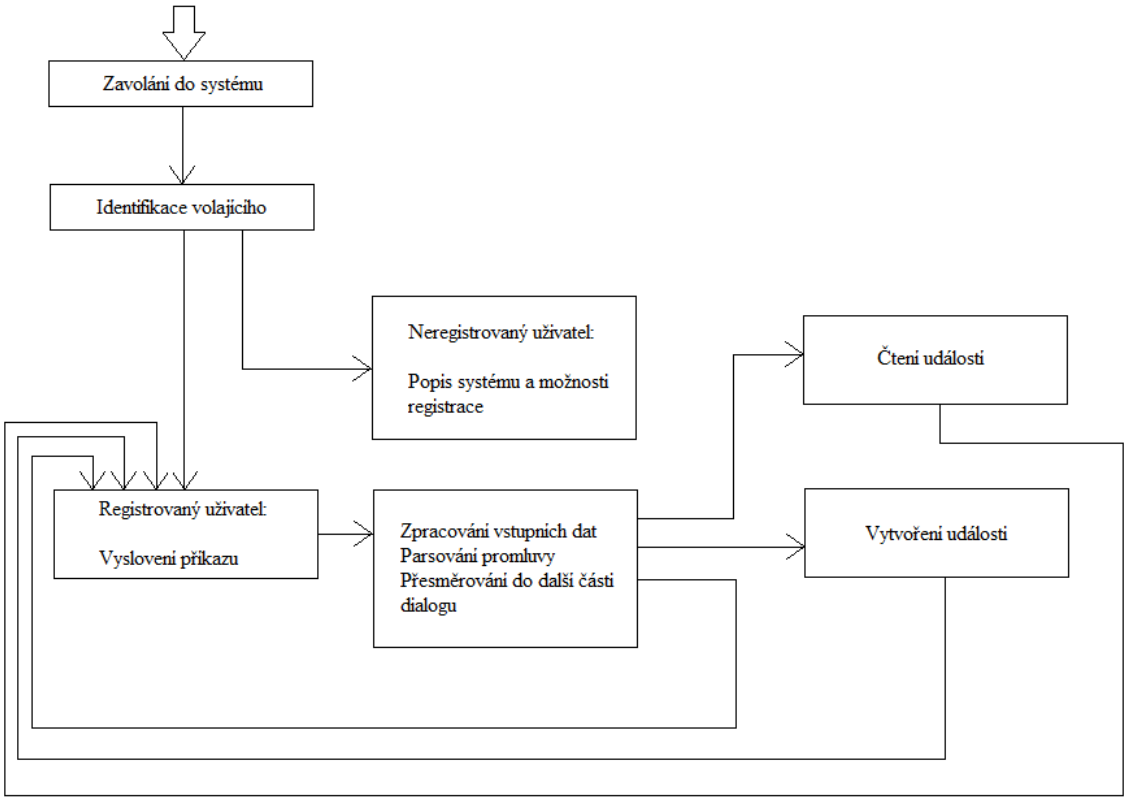
Nyní se podíváme blíže na hlasovou aplikaci, díky níž můžeme komunikovat s kalendářem pomocí hlasu, přidávat do něj události, číst je, popřípadě je také modifikovat nebo je mazat. V hlasové aplikaci jsme využili všech poznatku, které byly popsány v předchozích kapitolách.

8.1. Uzel

Jelikož v kalendáři můžeme provádět mnoho akcí, nebylo možné všechny tyto akce vytvořit v této práci, proto jsem se dialog snažil navrhnout takovým způsobem, aby byly vyřešeny všechny hlavní problémy a aby případné další funkcionality bylo možné přidat co nejjednodušeji.

Proto jsem navrhl jakýsi uzel, kam bude uživatel přesměrován, až se ověří jeho totožnost. Zde vysloví příkaz nebo název akce, kterou chce vykonat, program ho přesměruje do části dialogu, kde se vykonají potřebné akce a zase se vrátí zpět do uzlu, kde bude moci vyslovit další příkaz, který již s tím předchozím nesouvisí.

8.1.1. Struktura dialogu



Obr. 12: Uzel.

Na obrázku je znázorněna struktura uzlu a s připojením aplikace na čtení událostí a s připojením aplikace na vytvoření události.

Po zavolání do systému se zjistí číslo volajícího. To se porovná s databází registrovaných uživatelů, a pokud je uživatel registrovaný, přejde se do části dialogu, kde bude moci zadat příkaz. Pokud systém volajícího nerozpozná, přesměruješ na skript, kde si uživatel může poslechnout nějaké informace o systému a o registraci.

Registrovaný uživatel je vyzván k vyslovení příkazu. Příkazy může vyslovovat, viz kapitola o gramatikách. Po vyslovení příkazu, se v dalším skriptu promluva zpracuje. Pokud vyslovil správně danou akci bude přesměrován do časti dialogu, která slouží ke čtení nebo přidávání akce. Pokud vyslovil uživatel příkaz nevhodně, bude vyzván znovu k vyslovení.

8.1.2. Adresářová struktura

Nyní si ukážeme výpis souborů, které bylo třeba vytvořit v hlavní části hlasové aplikace pro naši práci Hlasové rozhraní pro online kalendář.

- **Kořenový adresář**
  - **createEvent**
  - **readEvent**
  - checkUser.php
  - config.php
  - crypt.php
  - desc.php
  - engine.php
  - index.php
  - node.php
  - parser.php

8.1.3. Popis souborů

Soubory si popíšeme, jakou mají funkci v našem systému.

createEvent – Adresář obsahující soubory pro přidávání záznamů do kalendáře. Jednotlivé soubory, které se v něm nacházejí, si popíšeme později.

readEvent – Adresář obsahující souboru pro čtení událostí. Také si ho přiblížíme později.

checkUser.php – Zde probíhá identifikace uživatele, zda je registrovaný do systému a na základě toho rozhodování, kam dále bude přesměrován. Registrovaný uživatel bude přesměrován do **node.php** a neregistrovaný do **desc.php**.

config.php – V tomto skriptu najdeme přihlašovací údaje do databáze.

crypt.php – Stejně jako u webové části zde máme skript, které slouží ke krytování, protože některé údaje v databázi jsou z bezpečnostních údajů zakryptovány.

desc.php – V tomto souboru je obsažen dialog, kde se neregistrovaný uživatel dozví více informací o systému.

engine.php – Slouží k rozhodování, které akce (a další údaje) byla vyslovena a podle směřuje tak dialog do **createEvent**, **readEvent** nebo zpět do **node.php**.

index.php – Úvodní skript hlasové aplikace, kde se zjistí číslo volajícího.

node.php – Zde registrovaný uživatel vysloví příkaz. K tomuto skriptu je připojena gramatika Calendar, která byla popsána v sekci gramatiky.

parser.php – Tento soubor je připojen do **engine.php** a slouží k parsování promluvy.

8.1.4. Ukázky použití

Ukážeme si, jak vypadá, když registrovaný uživatel zavolá do systému a jak probíhá komunikace:

- Uživatel vytočí číslo do systému.
- Systém rozpozná uživatele a přesměruje do **node.php**.
- Systém: *„Dobrý den Martine. Mluvte prosím.“*
- Uživatel: *„Chci přidat novou událost na zítra na 12:30“*
- Systém rozpozná promluvu a pokračuje se v části, která slouží na přidání události (createEvent/index.php).

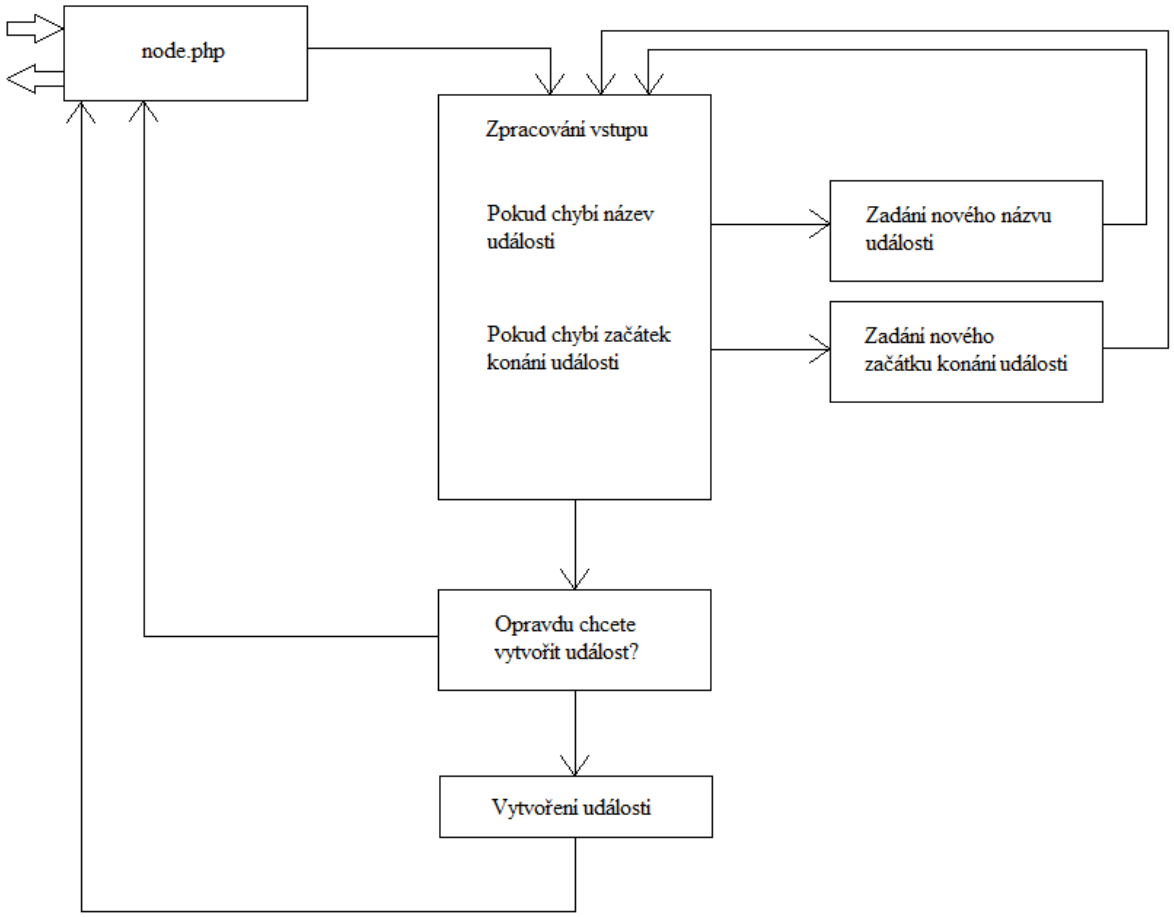
8.2. Vytvoření události

Vytvoření události je dílčí aplikace v našem systému. Protože na začátku můžeme vyslovit prakticky cokoliv, může se stát, že uživatel vysloví pouze „*vytvořit*“, proto je potřeba získat další informace.

Při vytváření může systém získat název události, pokud není uveden nebo může získat začátek konání události. Konec konání události zde získat nemůže, předpokládá se, že se událost bude konat hodinu. Pokud uživatel chce přidat i konec konání události, musí to říci na začátku, při vytváření události, od kdy do kdy se událost koná. Nebo popřípadě to může změnit při čtení událostí.

V tomto okamžiku se vyplácí, že jsem gramatiky rozdělil na několik dílčích, protože tam kde zjišťujeme začátek konání události, vím, že se jedná o časový údaj a tedy použiju gramatiku Times. Podobně tam kde zjišťujeme název konání události, použiju gramatiku Words. Díky tomu je pak snazší parsování promluvy a systém se vyvaruje tak některým chybám v promluvě uživatele.

8.2.1. Struktura dialogu



Obr. 13: Vytvoření události.

Do dialogu pro vytvoření události vstoupí sysstém z **node.php**, zpracují se vstupy, popřípadě se doplní, pokud chybí čas začátku nebo název. Dále se uživateli zopakují všechny podrobnosti o událost a systém vyzve uživatele, aby potvrdil vytvoření události. Pokud uživatel vysloví *„ne“*, dialog se vrátí do **node.php**. Pokud uživatel souhlasí s vytvořením události, událost se vytvoří, a poté dialog přejde také do **node.php**.

8.2.2. Adresářová struktura

- **createEvent**
  - **Zend**
  - create.php
  - index.php
  - newTimeStart.php
  - newTitle.php
  - query.php

- vgc_library.php

8.2.3. Popis souborů

Zend – Složka obsahující knihovny Zend Gdata potřebné k práci s kalendářem.

create.php – V tomto souboru doje k vytvoření nové události a k následnému přesměrování do **node.php**.

index.php – Na tento skript se odkazuje z **node.php**. Zjistí, zda jsou uvedeny všechny potřebné vstupy a podle toho připojí buď **newTimeStart.php** nebo **newTitle.php** a nebo **query.php**.

newTimeStart.php – Zde se zjišťuje nový začátek události. Sem je připojena gramatika Times.

newTitle.php – Zde zjišťujeme nový název události. K tomuto souboru je připojena gramatika Words.

query.php – Obsahuje otázku, zda uživatel opravdu chce přidat událost. Podle toho jestli uživatel řekne *„ano“* nebo *„ne“* směřuje další dialog.

vgc_library.php – Tento soubor byl blíže popsán v kapitole o Zend Gdata (5.4.1.). Obsahuje funkce, pomocí kterých můžeme pracovat s kalendářem.

8.2.4. Ukázky použití

Ukážeme si dva příklady na vytvoření události. V prvním případě uživatel na začátku řekne *„Chci vytvořit událost na zítra 12 hodin 30 minut. Snídaně s Petrem.“* Ukázka začíná v okamžiku, kdy dialog přejde do části na vytvoření události.

Systém: *„Název události je: Snídaně s Petrem. Začátek konání události je: 9.5. 12:30. Konec konání události je 9.5. 13:30. Opravdu chcete přidat tuto událost?“*
Uživatel: *„Ne“*
Systém: „*Událost nebyla vytvořena. Za moment budete moci zadat další příkaz.“*

Další příklad ukazuje, jak bude probíhat dialog, pokud uživatel na začátku vysloví minimum informací. Řekne tedy například *„Přidat událost“*. Ukázka znázorňuje, jak systém získá další informace a vytvoří událost.

Systém: *„Řekněte prosím název událost.“*
Uživatel: *„Snídaně s Petrem.“*
Systém: *„Řekněte prosím začátek konání události.“*
Uživatel: *„Zítra 12 hodin 30 minut.“*
Systém: *„Název události je: Snídaně s Petrem. Začátek konání události je: 9.5. 12:30. Konec konání události je 9.5. 13:30. Opravdu chcete přidat tuto událost?“*
Uživatel: *„Ano“*
Systém: „*Událost byla vytvořena. Za moment budete moci zadat další příkaz.“*

8.3. Čtení událostí

Další dílčí aplikací je čtení událostí. Při čtení je možné také čtenou událost smazat anebo změnit nějaké její údaje jako je začátek startu, místo konání a podobně.

Čtení může probíhat buď od aktuálního času, a nebo pokud uživatel v promluvě řekne čas nebo určitý den, čtení probíhá od tohoto časového určení.

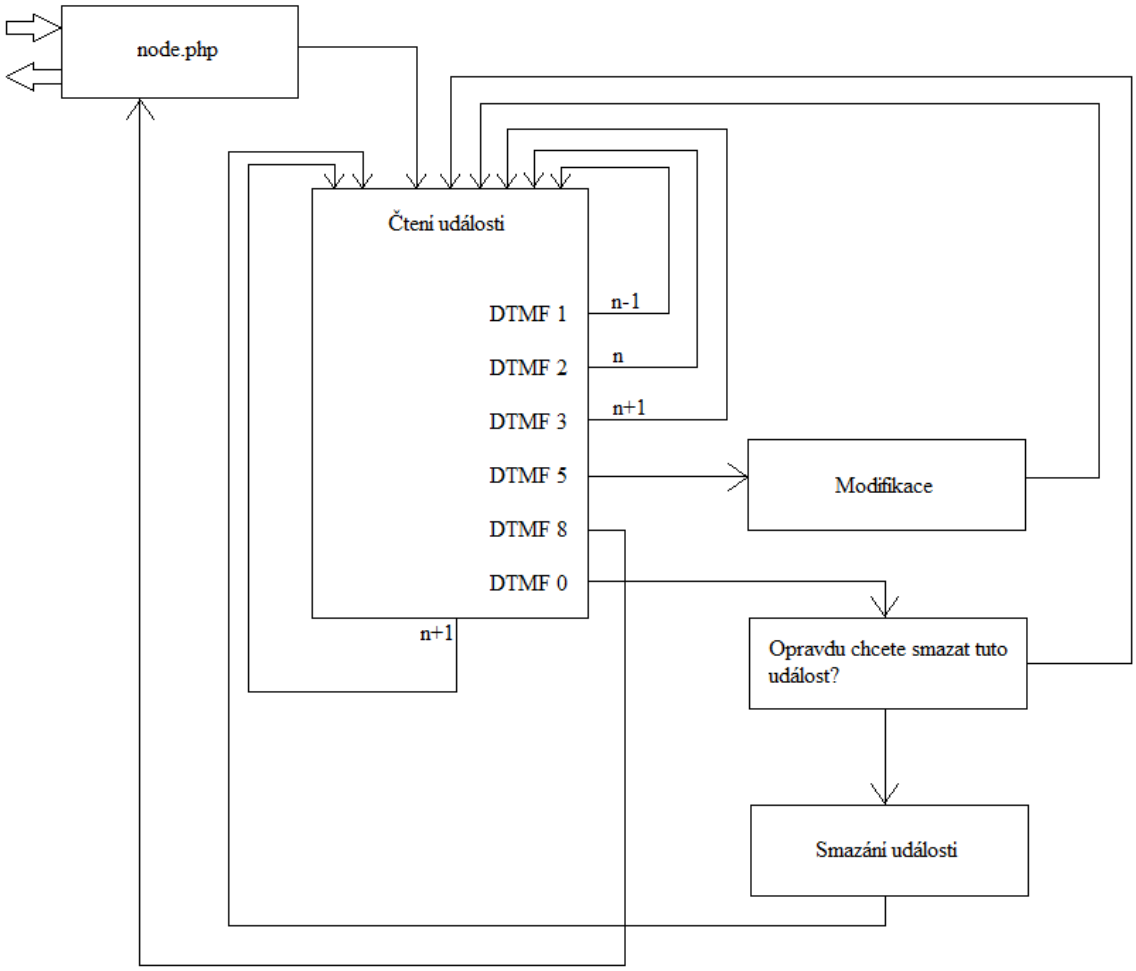
Při čtení je také dobré zajistit, aby uživatel mohl mezi jednotlivými událostmi přeskakovat nebo pokud špatně rozuměl tak danou událost zopakovat. Toto je zajištěno dtmf volbou. Pomocí dtmf volby uživatel přejde ke změně události, mazání nebo se vrátí k vyslovení nového příkazu.

8.3.1. Struktura dialogu

Do dialogu pro čtení vstoupíme opět z **node.php** po vyslovení příkazu pro čtení. Začne se číst událost a uživatel má následující možnosti:

Stisk „1“ – Začne se číst předchozí událost.
Stisk „2“ – Čtená událost se zopakuje od začátku.
Stisk „3“ – Začne se číst následující událost.
Stisk „5“ – Přejde se k modifikaci události.
Stisk „8“ – Přesměruje se na začátek k vyslovení dalšího příkazu.
Stisk „0“ – Přejde se ke smazání události.

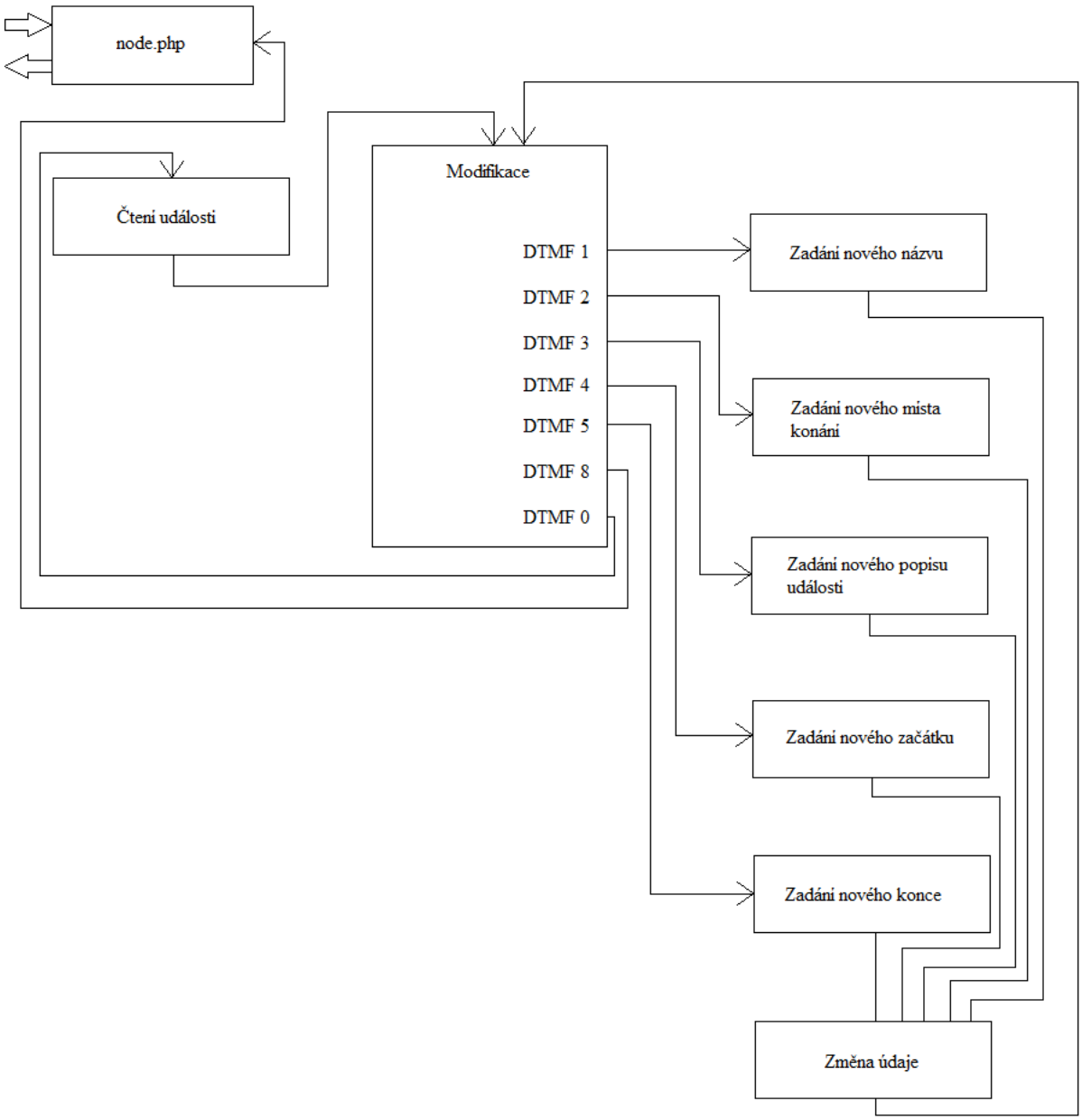
Pokud nebude vybrána žádná akce, pokračuje se ve čtení následující události.



Obr. 14: Čtení události.

Dále si podrobněji probereme modifikaci událostí. Když se uživatel dostane do této části dialogu, muže změnit 5 věcí u události. Název, místo konání, popis, začátek konání a konec. Bude mu vždy přečten například název události, a pokud ho chce změnit, provede to DTMF volbou.

Stisk „1“ – Změní se název události.
Stisk „2“ – Změní se místo konání.
Stisk „3“ – Změní se popis událost.
Stisk „4“ – Změní se začátek konání.
Stisk „5“ – Změní se konec konání.
Stisk „8“ – Přesměruje se na začátek k vyslovení dalšího příkazu.
Stisk „0“ – Vrátí se zpět na čtení událostí.



Obr. 15: Modifikace události.

Ke změně názvu, místa konání a popisu je připojena gramatika Words. Ke změně časových údajů je připojena gramatika Times.

8.3.2. Adresářová struktura

I zde si ukážeme adresářovou strukturu aplikace pro čtení událostí.

- **readEvent**
  - **Zend**
  - create.php

- delete.php
- fill.php
- index.php
- query.php
- update.php
- vgc_library.php

8.3.3. Popis souborů

Jednotlivé soubory si také popíšeme, jakou mají funkci v našem systému.

Zend – Složka obsahující knihovny Zend Gdata potřebné k práci s kalendářem.

create.php – Do tohoto souboru vstupuje promluva z modifikace, při změně údajů. Promluva se zde rozparsuje a uloží se do kalendáře.

delete.php – Tento soubor slouží ke smazání události z kalendáře.

fill.php – Zde uživatel vyplňuje nové údaje. Nový název a tak dále. Dále se pokračuje do **create.php**, kde se vyplněné údaje zpracují.

index.php – Hlavní soubor pro čtení událostí. Zde se čtou jednotlivé události a odtud se uživatel dostane na smazání události nebo na její modifikaci, popřípadě zpět na zadání jiného příkazu.

query.php – V tomto skriptu se systém zeptá uživatele, zda opravdu chce smazat událost. Pokračuje se do souboru **delete.php**.

update.php – Hlavní skript pro modifikaci události. Zde si uživatel vybere, který údaj chce změnit a odtud se přesměruje do **fill.php**.

vgc_library.php – Tento soubor byl blíže popsán v kapitole o Zend Gdata. Obsahuje funkce, pomocí kterých můžeme pracovat s kalendářem.

8.3.4. Ukázky použití

Nyní si ukážeme příklad rozmluvy uživatele se systémem. V prvním příkladu si ukážeme, jak to vypadá, když při čtení událostí chce uživatel změnit název události a zadat nový konec události. Ve druhém případě se ukážeme, jak to vypadá, když uživatel při čtení přeskakuje mezi jednotlivými událostmi a jednu událost pak smaže.

Na začátku uživatel řekl *„Chtěl bych přečíst události“*:

Systém: *„09.05. od 12:00 do 09.05. 13:00 Schůzka s Karlem. V Plzni. Důležitá schůzka o novém projektu.“*

Uživatel: Stisk „5“.

Systém: *„Nyní můžete změnit událost. Název události je: Schůzka s Karlem. Pokud ho chcete změnit, stiskněte jedničku. Místo konání události je: V Plzni. Pokud ho chcete změnit, stiskněte dvojku. Popis události je: Důležitá schůzka o novém projektu. Pokud ho chcete změnit, stiskněte trojku. Začátek konání události je. 09.05. 12:00. Pokud ho chcete změnit, stiskněte čtyřku. Konec konání události je. 09.05. 13:00. Pokud ho chcete změnit, stiskněte pětku. Pokud je událost v pořádku, stiskněte nulu a pokračujte ve čtení událostí.*

Uživatel: Stisk „1“.

Systém: *„Nyní řekněte prosím nový název události.“*

Uživatel: *„Schůzka s Petrem.“*

Systém: *„Nový název události je: Schůzka s Petrem. Budete moci změnit další údaj nebo se nulou vrátit na čtení událostí nebo osmičkou na začátek dialogu.“*

Systém: *„Nyní můžete změnit událost…… a pokračujte ve čtení událostí.*

Uživatel: Stisk „5“.

Systém: *„Nyní řekněte prosím nový čas konce události.“*

Uživatel: *„9. května 13:30.“*

Systém: *„Nový čas konce události je: 9.5. 13:30. Budete moci změnit další údaj nebo se nulou vrátit na čtení událostí nebo osmičkou na začátek dialogu.“*

Další dialog ukazuje přeskakování událostí a smazání jedné z nich.

Systém: *„09.05. od 12:00 do 09.05. 13:00 Schůzka s Karlem. V Plzni. Důležitá schůzka o novém projektu.“*

Systém: *„09.05. od 14:00 do 09.05. 15:00 Schůzka s Martinou kvůli práci.“*

Systém: *„09.05. od 20:00 do 09.05. 21:00 Večeře s Markétou. Na střelnici v Klatovech.“*

Uživatel: Stisk „1“.

Systém: *„09.05. od 14:00 do 09.05. 15:00 Schůzka s Martinou kvůli práci.“*

Uživatel: Stisk „0“.

Systém*: „Opravdu chcete smazat událost Schůzka s Martinou kvůli práci?“*

Uživatel: *„Ano.“*

Systém: *„Událost byla smazána. Čtení bude pokračovat.“*

Dále si ukážeme, jak vypadá takový VoiceXML dokument, který je vygenerovaný systémem. Tento dokument slouží k přečtení jedné události a umožňuje uživateli zvolit si, co s událostí chce dělat (smazat, změnit) nebo umožňuje přeskočit na jiné události. Tento dokument se předá VoiceXML interpretu a je přečten.

```
<?xml version="1.0" encoding="utf-8"?>
  <vxml version="1.0">
    <help>
      Po přečtení záznamu máte následující možnosti. Stiskutím jedničky
      přejdete na další záznam, stisknutím dvojky záznam zopakujete,
      stisknutím trojky přejdete na předchozí záznam. Stisknutím pětky
      můžete záznam modifikovat. Stisknutím nuly záznam smažete. Stisknutím
      osmičky se vrátíte na začátek dialogu.
     <reprompt/>
    </help>
    <menu id="zprava">
        <property name="inputmodes" value="dtmf"/>
        <property name="maxdigits" value="1"/>
        <property name="timeout" value="1"/>
        <prompt>

            09.05. od 12:00 do 09.05. 13:00 Schůzka s Karlem. V Plzni.
            Důležitá schůzka o novém projektu.

        </prompt>
        <choice dtmf="*" event="help"/>
        <choice dtmf="1" next="index.php?n=-1&amp;time=MjAxMi0wNS0wOVQwMT
               ozNTo0NCswMjowMA==" />
        <choice dtmf="2" next="index.php?n=0&amp;time=MjAxMi0wNS0wOVQwMT
               ozNTo0NCswMjowMA==" />
        <choice dtmf="3" next="index.php?n=1&amp;time=MjAxMi0wNS0wOVQwMT
               ozNTo0NCswMjowMA==" />
        <choice dtmf="0" next="query.php?title=Schůzka_s_Karlem.&amp;n=0
               &amp;time=MjAxMi0wNS0wOVQwMTozNTo0NCswMjowMA==" />
        <choice dtmf="5" next="update.php?n=1&amp;time=MjAxMi0wNS0wOVQwMT
               ozNTo0NCswMjowMA==" />
        <choice dtmf="8" next="../node.php" />
        <noinput>
          <goto next="index.php?n=1&amp;time=MjAxMi0wNS0wOVQwMTozNTo0NCsw
               MjowMA==" />
        </noinput>
        <nomatch>
          <goto next="index.php?n=1&amp;time=MjAxMi0wNS0wOVQwMTozNTo0NCsw
               MjowMA==" />
        </nomatch>

     </menu>
</vxml>
```

9. Závěr

Prvním cílem této práce bylo vytvořit webovou aplikaci pro hlasové rozhraní pro on-line kalendář. Vytvořil jsem tuto aplikaci. Uživatelé mají možnost se do systému registrovat a poté také mohou své údaje pozměnit. Administrátor systému má pak možnost také spravovat informace jednotlivých uživatelů. V této webové aplikaci se také uživatelé dozví více informací o systému a o ovládání a momentálně je dostupná na adrese http://voicexml.zcu.cz/zis/vgc/web/.

Dále jsem měl za úkol vytvořit hlasové rozhraní. Nejprve bylo nutné vybrat vhodný nástroj, který by zajišťoval přístup do Google kalendáře pomocí PHP. Dlouhou dobu jsem testoval Google API, které je zatím v beta verzi. Poté jsem ale vyzkoušel práci s uživatelskou knihovnou Zend Gdata a rozhodl jsem se, že použiju právě ji. Zend Gdata řešila některé problémy, které nastávaly při použití Google API a její použití bylo také jednodušší.

Další důležitou věcí bylo navržení vhodných gramatik. Gramatiky, které jsem používal v mém systému, byly pouze testovací, to znamená, že obsahovaly malé množství slov. Například gramatika Times by se dala rozšířit o časové údaje, které vyjadřují část celku, například půl páté odpoledne, čtvrt na tři ráno. Také gramatika Action by měla obsahovat mnohem více slov. Gramatika Words by měla ale obsahovat nejvíce slov a pokrýt co nejvíce promluv, které může člověk říci. Gramatiky jsou připraveny tak, že tam jednotlivá slova stačí už pouze dopsat do souboru a tím se gramatiky rozšíří a nemusí se měnit programová část. Testovací gramatiky s omezeným počtem slov byly použity z toho důvodu, že gramatika se na začátku dialogu kompiluje a to trvá určitý čas. Pokud bych měl obsáhlé gramatiky, uživatel by musel čekat příliš dlouho, něž by se gramatika načetla. To by se dalo vyřešit překompilováním gramatiky.

V hlasové aplikaci jsem musel nejprve vytvořit jakýsi uzel, který by zajišťoval připojení gramatiky a parsování promluvy. K tomuto uzlu by se pak daly snadno připojit jednotlivé dílčí aplikace, které by pracovaly s kalendářem. Toto jsem také splnil a vytvořil aplikace na čtení událostí a na přidávání událostí.

Aplikace na vytvoření události, vytvoří událost z rozparsované vstupní promluvy, popřípadě pokud na začátku řekneme málo potřebných informací, tato aplikace se na tyto informace zeptá.

Aplikace na čtení události, získá dané události od určitého času a říká je uživateli. Uživatel může při tomto čtení mezi událostmi přeskakovat nebo je modifikovat a nebo mazat.

Díky vytvoření uzlu je tak možné snadno přidávat do systému další aplikace jako je například hledání událostí podle klíčového slova a podobně. Stačilo by vložit klíčová slova do gramatiky Action a do parsovací části. Poté už jen z uzlu na tuto další aplikaci odkazovat.

10. Použitá literatura

[1] Psutka Josef, Müller Luděk, Matoušek Jindřich, Radová Vlasta: Mluvíme s počítačem česky, Academia, Praha 2006
[2] Wikipedia EN – VoiceXML
http://en.wikipedia.org/wiki/VoiceXML
[3] Voice XML a ďalšie skriptovacie jazyky
http://nil.uniza.sk/sip/services/voice-xml-dalsie-skriptovacie-jazyky
[4] Wikipedia CS - Google
http://cs.wikipedia.org/wiki/Google
[5] Google - Společnost
https://www.google.com/intl/cs/about/corporate/company/index.html
[6] Google – Naše filozofie
http://www.google.cz/intl/cs/about/corporate/company/tenthings.html
[7] Google - Služby
http://www.google.cz/intl/cs/about/products/
[8] Google – Co je Google Earth?
http://support.google.com/earth/bin/answer.py?hl=cs&answer=176145&topic=2376010&ctx=topic
[9] Wikipedia CS - Youtube
http://cs.wikipedia.org/wiki/YouTube
[10] Google user group – Google API
http://wiki.gug.cz/pro-vyvojare-1/google-api
[11] Google support - Calendar
http://support.google.com/calendar/
[12] Google Developers – Google Calendar API
https://developers.google.com/google-apps/calendar
[13] Malý Martin, Zdroják.cz, REST: Architektura pro webové API
http://zdrojak.root.cz/clanky/rest-architektura-pro-webove-api/
[14] Google Developers – Google Apps Calendar API: Downloads
https://developers.google.com/google-apps/calendar/downloads
[15] Google Developers – Client Libraries
http://code.google.com/intl/cs/apis/gdata/docs/client-libraries.html
[16] Wikipedia CS – Zend Framework
http://cs.wikipedia.org/wiki/Zend_Framework
[17] jQuery
http://www.jquery.com
[18] Skála Martin: Bakalářská práce – Zpravodajský informační server, Plzeň 2010

11. Seznam obrázků