EKONOMICKÁ UNIVERZITA V BRATISLAVE

Fakulta Hospodárskej Informatiky

Evidenčné číslo: 103004/I/2014/1400397899

Aplikácia ASP.NET webových služieb v úrokovej správe bežného bankového účtu

(Diplomová práca)

2014 **Martin Hužvár**

EKONOMICKÁ UNIVERZITA V BRATISLAVE

Fakulta Hospodárskej Informatiky

Aplikácia ASP.NET webových služieb v úrokovej správe bežného bankového účtu

(Diplomová práca)

Študijný program: Manažérske rozhodovanie a informačné technológie

Študijný odbor: 3.3.24 – Kvantitatívne metódy v ekonómii

9.2.10 – Hospodárska informatika

Školiace pracovisko: Katedra Aplikovanej Informatiky

Vedúci záverečnej práce: Ing. Igor Košťál, PhD.

Bratislava 2014 **Martin Hužvár**

Čestné vyhlásenie

Čestne vyhlasujem, že som záverečnú prácu vypracoval samostatne a že som uviedol všetku použitú literatúru.

Dátum: .....

Poďakovanie

Chcem sa poďakovať vedúcemu mojej diplomovej práce Ing. Igorovi Košťálovi, PhD. za rady, pripomienky, komentáre, vedenie a usmernenie pri vytváraní a spracovaní mojej práce.

Hužvár, Martin: Aplikácia ASP.NET webových služieb v úrokovej správe bežného bankového účtu. – Ekonomická univerzita v Bratislave. Fakulta Hospodárskej Informatiky; Katedra Aplikovanej Informatiky. – Vedúci záverečnej práce : Ing. Igor Košťál, PhD.– Bratislava: skratka FHI EU, 2014, 52s.

ABSTRAKT

Diplomová práca spracováva tému využitia webových služieb v úrokovej správe bežného bankového účtu. Najprv objasňuje základné teoretické znalosti z dvoch rozdielnych sfér a to úročenie v bankovom sektore a webové služby. Na záver sú tieto dve oblasti spojené v jednej webovej službe, ktorá je naprogramovaná v jazyku C# na platforme ASP.NET s HTML klientom pre webový prehliadač a ktorá na základe vstupných údajov používateľa pristupuje k dátam jednotlivých účtov uložených v XML súbore, prepočítavá úrok na základe zvolenej úrokovej periódy a zobrazuje odpoveď na zadanú úlohu vo forme výstupu dialógového okna a zápisu výsledku do XML súboru.

Diplomová práca obsahuje štyry kapitoly. Nachádza sa v nej 9 obrázkov znázorňujúcich procesné grafy, dizajn a výsledky webovej služby. V prvej kapitole je opísaný súčasný stav riešenej problematiky. Druhá kapitola obsahuje teoretické východiská. V prvej podkapitole sa pojednáva o základoch úročenia v bankovom sektore, sú opísané základné pojmy, druhy úročenia a základné vzorce využívané pri úročení. Ďalšia podkapitola pojednáva o webových službách. Je v nej vysvetlená teória a pojmy týkajúce sa webových služieb, základné charakteristiky, prvky a systémy využité vo webových službách. Táto podkapitola taktiež obsahuje obrázky a grafy znázorňujúce úlohy a využitie webových služieb v servisne orientovanej architektúre.

Tretia kapitola sa zaoberá cieľmi a metodikou práce. Charakterizované sú pracovné postupy použité pri tvorení webovej služby, ako aj ciele a čiastkové ciele, ktoré slúžili na analýzu problematiky témy diplomovej práce a implementáciu jej riešenia.V štvrtej kapitole je detailne opísaná webová služba, ktorá predstavuje výsledky práce, teda využitie ASP.NET webovej služby pri úrokovej správe bežného bankového účtu. Každá dôležitá časť a proces programu sú popísané, rovnako aj celý proces získania výsledkov.V poslednej, piatej kapitole, s názvom Záver sú zhrnuté a zhodnotené ciele, postupy a výsledky celej práce.

Kľúčové slová: webové služby, úrokovanie, ASP.NET

Hužvár, Martin: Using ASP.NET web services in interest management of common bank account. –University of Economic. Faculty of Business Informatics; Department of Applicable Informatics – Supervisor: Ing. Igor Košťál, PhD. – Bratislava: FHI EU, 52p.

ABSTRACT

Article is focused on the topic of using web services in the interest administration of the common bank account. Article clears the basic theoretical knowledge from the both areas - interest rate and web services. In the end, both of those areas are connected in the program, which is made in the C# language on the ASP.NET platform with HTML client for the web browser. Program approaches the user data saved in the XML file, counts the interest by the user`s ordered interest period and displays the results of the task on the dialog screen and furthermore, writes the result to the file too.

Diploma thesis is divided to four chapters. There are 9 pictures describing process diagrams and web service design and results. In the first chapter is described the nowadays conditions regarding to the topic of the work.

Second chapter deals with the theoretical basics of the both areas, bank interests and web services. In the first subchapter, there are information about the interest terms, interests formulas and different approaches in interest counting. In the next subchapter, there are information about the web services. Basic theory and terms such as WSDL, SOAP and XML are explained, features and properties are described. There are also several figures and diagrams describing the basic roles of the web services and the processes inside them.

Third chapter is focused on aims and methods of this work. Goals and sub-goals are defined and explained as well as all the methods used to reach those goals. In fourth chapter is every step as same as the whole process of the web services implementation described and explained. Explanations go through the program lines and describe whole process of gaining the final results: from signing the user in by the birth number and account number, accessing the data in XML file, choosing the interest period, counting and displaying the results. In the last, fifth chapter are summed up the aims, methods and results of the whole diploma thesis.

Key words: web services, bank interest, ASP.NET

Obsah

Zoznam grafov, obrázkov a príkladov

Zoznam skratiek

MEP - Message Exchange Protocol

FCL - Framework Class Library

XSLT - Extensible Stylesheet Language Transformations

HTTP - Hypertext Transfer Protocol

SOAP - Service Oriented Architecture Protocol

SMTP - Simple Mail Transfer Protocol

RMI/IIOP - Remote Method Invocation interface over the Internet Inter-Orb Protocol

XML - Extensible Markup Language

SOA - Service Oriented Architecture

XSD - XML Schema Definition

TCP - Transmission Control Protocol

UML - Unified Modeling Language

WSDL - Web Service Definition Language

WS-BPEL - Web Services Business Process Execution Language

PKI - Public Key Infrastructure

CM - Cash Management

WSIL - Web Service Inspection Language

UDDI - Universal Description, Discovery and Integration

HTML - Hypertext Markup Language

ASCII - American Standard Code For Information Interchange

1 Úvod do súčasného stavu riešenej problematiky

Webové služby sa dnes postupne zavádzajú do celého bankového sektora. Vďaka ich univerzálnosti a nezávislosti na technológii si môže klient napríklad vyhľadať informácie o depozitných úrokových sadzbách, poplatkoch a penále z vkladov a úverov alebo menových kurzoch.

V oblasti správy bežného účtu používateľa sa využívajú webové služby hlavne na transport súborov a dát medzi bankou a zákazníkom. Dáta sú zvyčajne kryptované SSL protokolom [3] a zákazníci sú identifikovaní verejným kľúčom PKI, prideleným bankou. Súbory sú zvyčajne v XML formáte alebo lokálnom Cash Management (CM) formáte a vďaka nim si môže zákazník skontrolovať stav účtu, zadať novú transakciu, či zobraziť prehľad vykonaných transakcií.

Úroková správa bežného účtu však nepatrí k službám bežne poskytovaným bankovými inštitúciami na Slovensku. Bankové webové servery poskytujú rôzne webové služby a riešenia, ale zo siedmich preskúmaných bankových webových portálov iba jeden poskytoval webovú aplikáciu, ktorá dokázala vypočítať mesačný úrok klienta. Táto aplikácia však nebola integrovaná do bankového systému a nepracovala nezávisle s klientskymi dátami, ale vystupovala iba ako jednoduchá aplikácia vo forme kalkulačky, ktorá na základe zadaných hodnôt vkladu, výšky úrokovej miery a úrokových období vypočíta hodnotu klientskeho vkladu. Táto aplikácia nedokázala komunikovať s inými systémami, čo je charakteristickým prvkom pre webové služby a preto sa ani v tomto prípade nedá hovoriť o webovej službe. Na ostatných bankových webových serveroch sa nenachádzala ani takáto jednoduchá aplikácia, ktorá by používateľom umožnila automaticky vypočítať úroky za určité obdobie.

Podobná situácia sa odohráva aj v zahraničných bankách, kde sa síce oveľa častejšie nachádzajú jednoduché aplikácie, nazývané úrokové kalkulačky, ktoré klientovi umožňujú výpočet mesačného a niekde aj ročného úroku, avšak z niekoľko preskúmaných zahraničných bankových serverov ani jeden neposkytoval webovú službu, ktorá by bola schopná pracovať s klientskymi dátami a vypočítala by rôzny typ úrokov na požiadanie. Preto sa táto práca zameriava na návrh a praktickú implementáciu takej webovej služby, ktorá by bankovým klientom umožnila prihlásiť sa na webový server pomocou čísla účtu

a hesla, spracovala by klientove údaje o vklade na účte a výške mesačného úroku a poskytla by šesť rôznych alternatív výpočtu úroku, menovite denný úrok z vkladu, týždenný úrok, mesačný, kvartálny, polročný a ročný úrok z vkladu. Aplikácia by pracovala s úrokom na základe mesačnej úrokovej periódy, kde sa úroky z vkladu pripisujú na účet na konci mesiaca. Výsledok by bol klientovi prezentovaný vo forme výstupu z aplikácie a informácia by sa ukladala do súboru na disku.

1.1 Teoretické východiská

Táto kapitola opisuje základné poznatky z oblasti matematických úrokových operácií a webových služieb.

1.1.1 Úročenie

V tejto práci som využil teoretické východiská dvoch rôznych vedných oblastí, finančnej matematiky a informatiky. Najprv je potrebné vysvetliť základné pojmy a definície pochádzajúce z finančnej matematiky, ako napríklad istina, úrok, úroková miera a úroková perióda a vzťahy medzi nimi.

„Peňažnú sumu, ktorú poskytuje veriteľ dlžníkovi za určitý poplatok, nazývame kapitál (istina). Poplatok, ktorý platí dlžník veriteľovi za používanie jeho peňazí sa nazýva úrok. Veľkosť úroku sa určuje ako percentová časť istiny za úrokové obdobie. Časové obdobie, za ktoré percentová miera určuje úrok ako časť kapitálu, sa nazýva úroková perióda. Percentovú mieru, zodpovedajúcu určitej perióde, nazývame úrokovou mierou.“[1] Úrokové miery majú rôzne označenia, ktoré závisia od úrokovej periódy. Existuje ročná úroková miera – per annum (p.a.), ďalej polročná – per semestrem (p.s.), taktiež štvrťročná úroková miera – per quartalem (p.q.), mesačná – per mensem (p.m.) a týždenná – per septimanam (p.sept.). [1]

V praktickej implementácii webovej služby boli použité výpočty ročnej úrokovej miery, polročnej úrokovej miery, štvrťročnej, mesačnej, týždennej a dennej úrokovej miery. Proces spojený s výpočtom úrokov sa nazýva úrokovanie.

[1] *PIRČ, V. a kol. 2008. Finančná Matematika. Košice. Katedra matematiky FEI Technickej univerzity v Košiciach. ISBN: 978-80-8073-986-7*

„Existujú dve základné spôsoby úrokovania:

- jednoduché úrokovanie
- zložené úrokovanie"[2]

1.1.1.1 Jednoduché úrokovanie

V prípade jednoduchého úrokovania sa úrok určuje v každej perióde z konštantného začiatočného vkladu. V prípade zloženého úrokovania sa úrok počíta z kapitálu zväčšeného o úroky z predchádzajúceho obdobia.

„Podľa splatnosti úroku hovoríme o dekurzívnom (polehotnom) úrokovaní a anticipatívnom (predlehotnom) úrokovaní"[3].

Dekurzívne úročenie je charakteristické pripísaním úroku na konci úrokovacej periódy, kdežto anticipatívne úročenie charakterizuje splácanie úroku na začiatku úrokovej periódy.

Na bežných bankových účtoch s jednoduchým úročením sa lineárne úrokovanie deje pripisovaním úrokov a zvyšovaním hodnoty kapitálu na základe vkladového obdobia.

„Zvlášť sa úrokuje každý vklad, a síce na základe nasledujúcich základných informácií:

1. dátumu vkladu
2. dátumu pripísania úrokov
3. nominálnej úrokovej miery v percentách vzťahujúcej sa na 1 rok ."[4]

Dátum vkladu reprezentuje začiatok doby úrokovania a dátum pripísania úrokov predstavuje ukončenie doby úrokovania.

Základný vzorec na výpočet úroku jednoduchým úročením je:

$$U = PV \,.\, i \,.\, t \,[1]$$

[2] *PIRČ, V. a kol. 2008. Finančná Matematika. Košice. Katedra matematiky FEI Technickej univerzity v Košiciach. ISBN: 978-80-8073-986-7*

[3] *PIRČ, V. a kol. 2008. Finančná Matematika Košice. Katedra matematiky FEI Technickej univerzity v Košiciach. ISBN: 978-80-8073-986-7*

[4] *PIRČ, V. a kol. 2008. Finančná Matematika. Košice. Katedra matematiky FEI Technickej univerzity v Košiciach. ISBN: 978-80-8073-986-7*

pričom skratka „*U*“ znamená úrok, „*i*“ je úroková sadzba, „*PV*“ je začiatočná hodnota kapitálu a „*t*“ je dĺžka úrokového obdobia vyjadrená v jednotkách úrokovej periódy. Ak úrok *u* pripočítame k začiatočnej hodnote *PV* , dostaneme konečnú (budúcu) hodnotu *FV* po čase *t* vyjadrenom v úrokovacích periódach.

$$FV = PV + PV.i.t = PV.(1 + i.t) \quad [1]$$

V praxi sa často namiesto skutočného úrokového obdobia využíva doba - 30 dňový mesiac a namiesto skutočného počtu dní v roku sa využíva 360 dní. [1]

1.1.1.2 Zložené úročenie

V implementovanej webovej službe je využité zložené úrokovanie, ktoré je dnes používané bežne v bankovom sektore a preto sa táto práca podrobnejšie venuje tomuto spôsobu úročenia.

Pri zloženom úrokovaní sú úroky pravidelne pripisované v stanovených časových intervaloch. Vznikajú tak úroky z úrokov. Suma kapitálu po prvom roku úročenia s úrokovou periódou jeden rok je vyjadrená vzorcom:

$$FV = PV + PV.i = PV.(1 + i) \quad [1]$$

kde skratka „FV“ predstavuje konečnú budúcu hodnotu kapitálu po zúročení, „PV“ označuje začiatočnú hodnotu kapitálu a „*i*“ je úroková sadzba.

Na konci druhého roka bude základný kapitál zväčšený o úrok z prvého obdobia a tiež o úrok z druhého obdobia, v ktorom je už zarátaný úrok z prvého obdobia. Môžeme to preto vyjadriť nasledovne:

$$FV_2 = FV_1.(1+i) = PV.(1+i)^2 \quad [1]$$

Po *n* obdobiach má vzorec podobu:

$FV_n = PV . (1+i)^n$ [1]

Tento vzorec bol využitý aj v implementovanej webovej službe na výpočet kvartálneho, polročného a ročného úroku. Jeho implementácia má v prípade ročného úroku podobu:

$FV = PV . (1 + i_2)^{12} - \text{PV}$ [1]

Ak chceme vyjadriť vzťah medzi jednoduchým a zloženým úrokovaním, napríklad kvôli prepočtu rentability jednotlivých typov úrokovania, získame ho nasledovne:

$FV_2 = PV . (1 + i_2)^{\text{n}}$ [1]
je vzorec zloženého úrokovania s úrokovou mierou, ktorá po určitom období vynesie kapitál FV_2.

$FV_1 = PV . (1 + i_1.n)$ [1]
je vzorec jednoduchého úrokovania s úrokovou mierou, ktorá po určitom období vynesie kapitál FV_1

Predpokladajme že základný kapitál PV je v obidvoch prípadoch rovnaký, tak ako aj dĺžka úrokovej periódy a obdobie úrokovania. Ak sa $\text{FV}_1 = \text{FV}_2$, potom

$i_1 = \frac{(1+i_2)n - 1}{n}$ [1]

Zo vzorca opísaného vyššie vidieť priamu závislosť medzi jednotlivými typmi úrokovania vo vyššie popísaných podmienkach. Ak je teda obdobie úročenia kratšie ako úroková perióda, napríklad pri výpočte týždenného alebo denného úroku, pričom sa v banke úroky pripisujú na konci mesiaca, je jednoduchšie a výhodnejšie jednoducho predeliť mesačný úrok číslom popisujúcim obdobie úročenia. Napríklad na vyjadrenie výšky týždenného úroku je vo webovej službe použitý tento vzorec:

$FV_1 = PV . (1 + i_1.n)/30$ [1]

Univerzálne sa počíta s 30 dňovými mesiacmi. Najprv sa teda vypočíta úrok za celý mesiac a následne sa rozdelí medzi tridsať dní. Podobne sa vypočíta aj týždenný úrok:
$FV_1 = PV . (1 + i_1.n)/4$ [1]

1.1.2 Webové služby a ASP.NET technológia

Po objasnení základných definícií a vzorcov z oblasti finančnej matematiky prejdeme v tejto kapitole k definícii webových služieb, aby sme pochopili teoretické základy tejto oblasti.

Webové služby sa postupne stávajú lídrom v oblasti spracovania a transportu informácií. Je to zrejmé aj z faktu, že mnohé veľké korporácie prispôsobujú fungovanie svojich softvérov práve webovým službách a snažia sa o ich integráciu. Taktiež sa o webových službách veľa píše v publikáciách a v článkoch z rôznych podnikových odvetví a takmer v každej oblasti informačných technológií. Prečo sa teda webové služby stávajú trendom a dostávajú do stredu pozornosti? Je to vďaka ich veľkému potenciálu, ktorý spočíva v schopnosti webových služieb integrovať firemné procesy. Táto integrácia je založená na technológiách umožňujúcich prepájanie rôznych aplikácií, pracujúcich na odlišných platformách a úrovniach a pracujúcich s rozličným hardvérom a dokonca vyvinutých v rôznych programovacích jazykoch. Okrem toho sa webové služby stávajú univerzálnym nástrojom na komunikáciu, riadenie a dosahovanie cieľov v spoločnosti a jej podnikových procesoch, pretože sú jednoducho ovládateľné a využívajú protokoly nezávislé na platforme, ako HTTP či XML.

1.1.2.1 *Základná charakteristika webových služieb*

Pod názvom webová služba sa ukrýva abstraktná charakteristika operácie alebo činnosti, ktorá sa vykoná programom alebo aplikáciou. Definícia Konzorcia W3C znie:
„Webová služba je softvérový systém navrhnutý na podporu interakcie strojov cez počítačovú sieť. Poskytuje rozhranie, ktoré je strojovo spracovateľné (špecificky WSDL). Ostatné systémy prichádzajú do interakcie s webovou službou spôsobom, ktorý je

predpísaný ich popisom pomocou SOAP správ, typicky prenášaných prostredníctvom HTTP s XML serializáciou v spojení s inými webovými štandardmi.“[5]

Pre webovú službu sú charakteristické tieto tri technológie [2]:

1. SOAP protokol
2. WSDL jazyk
3. UDDI alebo WSIL štandard

Základom webových služieb je dokument napísaný vo WSDL (web service description language) a protokol SOAP. Ďalej je služba založená aj na UDDI a WSIL štandardoch, ktoré slúžia na objavenie služieb a robia webové služby univerzálnymi a interoperabilnými.

Samotné webové služby predstavujú silný prostriedok informačného systému podniku na komunikáciu medzi jednotlivými aplikáciami. Dve základné a najpodstatnejšie požiadavky, ktoré webové služby úspešne pokrývajú, sú bezpečnosť a podpora transakcií. *Bezpečnosť* je zaistená súhrnom pravidiel a prístupov, ktoré slúžia na ochranu citlivých dát. Tieto procesy sú zabezpečené používaním šifrovania, digitálneho podpisu, autorizácie a iných. *Transakcie* sú také operácie, ktoré sa buď vykonajú všetky úspešne, alebo sa ich vplyv anuluje a zmena bude vrátená do pôvodného stavu. Preto charakterizujeme transakcie ako atomické.

1.1.2.2 SOAP

Skratka „SOAP“ znamená „jednoduchý protokol na prístup k objektom“[6] (Simple Object Access Protocol). Iná charakteristika tejto skratky je výraz „protokol pre servisne orientovanú architektúru“ (SOA Protocol). SOAP bol vyvinutý spoločnosťou Microsoft a neskôr k jeho tvorbe prispeli aj Lotus či IBM spoločnosti. SOAP jasne vymedzuje komunikáciu s webovou službou. Pracuje na báze peer-to-peer a využíva XML správy, ktoré predstavujú volanie webovej služby so zámerom vykonania určitej operácie resp. žiadosť o určité informácie. Tieto správy tiež reprezentujú výsledok predošlých operácií

[5] *BOOTH, D. a kol. 2004. Web Service Architecture. [cit. 2014-07-04] Dostupné na internete: < http://www.w3.org/TR/ws-arch/>.*

[6] *BOX, D. A kol. 2000. Simple Object Access Protocol. [cit. 2014-07-04]. Dostupné na internete: <http://www.immagic.com/eLibrary/ARCHIVES/SUPRSDED/W3C/W000520N.pdf>*

resp. získané dáta. Správy následne môžu byť posielané rôznymi transportnými protokolmi ako HTTP (resp. HTTPS) a SMTP.

Štruktúra SOAP správy začína vymedzením „soap:Envelope“, teda koreňového elementu, ktorého menný priestor odkazuje na SOAP Envelope XML schému.[10] Ďalším elementom je „soap:Body“, ktorý v podstate tvorí telo správy a je povinný. V tomto tele správy sa nachádzajú dáta, ktoré správa prenáša a ktoré súhrne pomenovávame „message payload“. Tieto dáta putujú od odosielateľa až k príjemcovi. Ďalej existuje „Soap:Fault“ element, ktorý je voliteľným elementom tela správy. Slúži na prenos chybových správ, zahŕňa informácie o chybovom kóde, príčine a pôvodcovi chyby. Súčasťou obálky môže byť aj nepovinný „soap: Header“ element, ktorý obsahuje bloky poskytujúce informácie o spôsobe, ako so správou pracovať. Telo správy ani hlavička nie sú protokolom SOAP nijako vymedzené, pretože tieto údaje sú špecifické pre konkrétne aplikácie, ktoré so SOAP správami pracujú.

1.1.2.3 Jazyk WSDL

„Web Services Description Language je jazyk, ktorý popisuje jediné rozhranie na komunikáciu s webovými službami. Toto rozhranie udáva, aké operácie služba poskytuje a akými operáciami sa dá služba volať.“[7] Verejné rozhranie je kľúčovou časťou webových služieb. Verejné rozhranie je informácia ktorá umožňuje, že je webová služba zavolaná inou službou. Táto činnosť je možná vďaka priradeniu identity pre danú službu.

Charakteristické vlastnosti WSDL [3]:

1. Unifikácia spracovania správ – nezávisle distribuované objektové systémy a aj systémy orientované na správy sú podporované WSDL

2. Podpora viacerých protokolov a transportov – WSDL podporuje množstvo typov protokolov určených na komunikáciu. SOAP zďaleka nie je jediným protokolom slúžiaci na výmenu správ, existujú ďalšie ako RMI/IIOP, TCP a iné.

3. Rozšíriteľnosť – WSDL jazyk charakterizuje viaceré stránky služieb – spôsob prezentácie správ zatiaľ čo sú transportované, formát správ a cieľ, kde majú byť správy poslané.

---

[7] *LAŠÁK, M. 2010. Verzovanie webových služieb v SOA aplikáciách. Brno. 64 s. [cit. 2012-04-01] Dostupné na internete: <http://is.muni.cz/th/172840/fi_m/dp_lasak.pdf>.*

4. Charakteristika kontextu správ– najprv klient zisťuje, čo daná služba poskytuje a až potom hľadá spôsob, ako službu použiť a kde sa daná služba nachádza. Jazyk WSDL rozlišuje tieto časti a ponúka cez popisy služby použitie rôznych typov interakcií na odlišných miestach.

5. Podpora typových systémov – jedným z cieľov WSDL je zachovanie voľnosti jazyka, ktorým sú definované, posielané alebo prijímané správy.

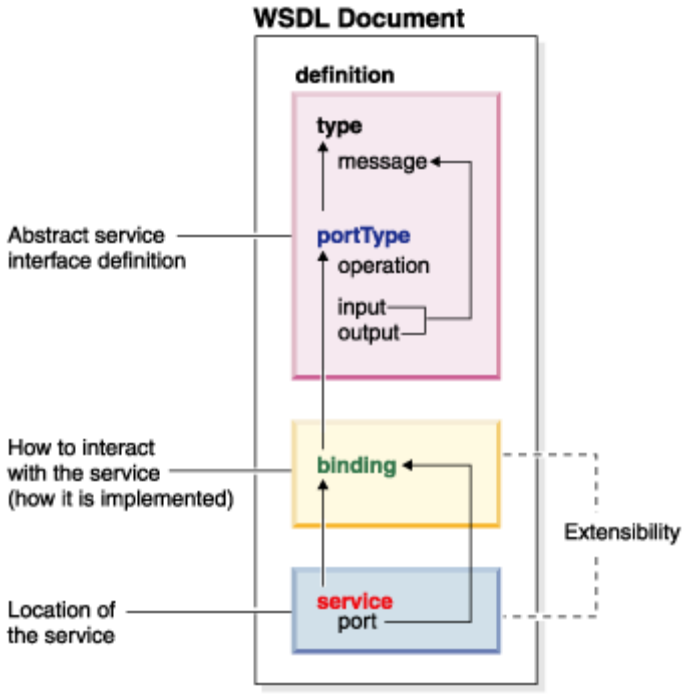
6. Žiadna sémantika – jazyk WSDL charakterizuje služby z pohľadu štruktúry a nie sémantiky. Podľa názvu operácie sa nedá zistiť, čo konkrétne služby urobí. Exituje však možnosť doplnenia tejto funkcionality.

7. Bez poradia - nič nedefinuje, ktoré operácie a v akom poradí budú nasledovať, poradie určuje klient alebo človek a nazývame to abstraktný proces. Taktiež vďaka WSDL môže byť vykonaných súčasne viacej operácii.

„WSDL jazyk popisuje webové služby na dvoch fundamentálnych stupňoch: abstraktnom a konkrétnom[8].“ V abstraktnej úrovni popisuje webové služby v podmienkach správ, ktoré posiela a prijíma. Tieto správy sú popísané zvyčajne vo formáte XML Schema a preto nezávisle na komunikačných technológiách. Popísaná je štruktúra správ a operácie ktoré služba poskytuje. V konkrétnej úrovni sú špecifikované prenosy správ a komunikačné technológie , ktoré musí klient vlastniť, aby sa mohli dané operácie uskutočniť.

Na nasledujúcom obrázku je vysvetlené, ako funguje WSDL dokument a z čoho sa skladá.

[8] *CHINNICI, R. a kol. 2007. Web Service Description Language Version 2.0 part 1: Core language. [cit. 2013-11-11]. Dostupné na internete:<http://www.w3.org/TR/wsdl20/#intro_ws>*



Obr. 1 : WSDL dokument[9]

Na začiatku dokumentu sa nachádza koreňový element „definitions“, ktorý vlastní obidva časti dokumentu. V abstraktnej časti sa nachádza element „types“ definujúci typy, ktoré sú použité vo WSDL dokumente a pri transporte správ. Element „message“ je element reprezentujúci abstraktnú správu, ktorou služba komunikuje s konzumentom a v dokumente sa môže vyskytovať viacero krát. Každý element musí mať jedinečný „name“ atribút, ktorý charakterizuje jedinečné meno pre každý element. Ďalej element „message“ vlastní aspoň jeden „part“ element, ktorý reprezentuje informáciu danej správy, teda parametre a návratové hodnoty z jednotlivých operácií, ktoré služba vykoná. Takisto aj element „part“ musí vlastniť jedinečný atribút „name“. Ďalej však môže obsahovať aj atribút „type“ definujúci dátový typ daného elementu alebo atribút „element“ odkazujúci na už vytvorený typ. Ďalší element ktorý definuje konkrétnu operáciu jeho služby je element „operation“. WSDL dokument by mal vlastniť minimálne jednu operáciu. Operácia vlastní ľubovoľné množstvo dcérskych elementov „input()“, „output()“ a „fault()“. „Input()“ je vlastne správa konzumenta zaslaná službe, „output()“ je správa služby zaslaná konzumentovi a „fault()“ je chybová správa poslaná službou pri výskyte chyby.

[9] ZAIKIN, M. 2009. Sun Certified Developer for Java Web Services 5.0. cit. [2014-21-09]. Dostupné na internete: < http://java.boot.by/scdjws5-guide/index.html>

“Podľa výskytu dcérskych elementov hovoríme o tzv. MEP (Message Exchange Pattern)

1. Request-Response – operácia prijme vstupnú správu a poskytne na ňu odpoveď vo forme výstupnej správy alebo chybovej správy (I => O/F)

2. One-Way - jednosmerná operácia prijímajúca vstupnú správu, ktorá nič nevracia.

3. Solicit-Response – predstavuje inverznú operáciu k 1. Služba pošle výstupnú správu alebo chybovú správu a čaká na vstup od konzumenta (O/F => I).

4. Notification – operácia len pošle konzumentovi výstupnú správu alebo chybovú správu a nečaká na odpoveď. (O)”[10]

Posledným elementom v abstraktnej časti je element „portType“, ktorý vymedzuje rozhranie služby z pohľadu operácií.

Prechádzame do konkrétnej časti WSDL dokumentu, v ktorej sú definované dva elementy – „binding“ a „service“. „Binding“ slúži na spojenie elementu „portType“ s komunikátorom, ktorý je použitý klientmi alebo konzumentmi za účelom volania rôznych operácií služby. Zvyčajne sa v dokumente môže nachádzať viacero „binding“ elementov, ktoré takto poskytujú viacero rozhraní pre klientov. „Binding“ tiež obsahuje elementy „operation“, ktoré charakterizujú operácie z „portType“, a tiež obsahuje dcérske elementy „operation“ – input, output a fault. „Binding“ tiež obsahuje „extensibility elements“, ktoré určujú komunikačné technológie použité pri prenosoch – atribút „transport“ a tiež formáty týchto správ – atribút „style“ a „use“. Posledným elementom v konkrétnej časti je element „service“, ktorý slúži na zaistenie sieťovej adresy pre konkrétny „binding“ element. Element „service“ obsahuje element „port“ definujúci tzv. „endpoint“, alebo fyzickú adresu služby.

1.1.2.4 UDDI a WSIL

UDDI predstavuje register, vďaka ktorému sú iné webové služby alebo klienti schopní lokalizovať hľadanú webovú službu.

[10] *Christensen, E. a kol. 2008. Web Service Description Language. Dostupné online: <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>*

„Skladá sa z dvoch hlavných komponentov:

- Webový register, slúžiaci na vyhľadávanie webových služieb. Hociktorý poskytovateľ môže zverejniť informácie o svojej službe, ktorú poskytuje.
- Štandardná XML schéma pre popis služby. Dáta v registri služieb majú formu XML dokumentov so štandardnými poľami používanými na popis služby“[11].

„WSIL“ dopĺňa UDDI vlastnosti týkajúce sa objavovania alebo nachádzania webových služieb. Kým UDDI poskytuje register so širokou škálou služieb, WSIL dokáže identifikovať lokálne služby dostupné na miestnom serveri alebo web stránke. Inšpekčný štandard WSIL dokáže preskúmať server a zobraziť zoznam poskytovaných služieb, ktorý obsahuje často tiež smerovače na WSDL dokumenty alebo iné webové služby v UDDI registri. Preto predstavuje WSIL jednoduchý a ľahký spôsob získavania informácií o dostupných lokálnych službách bez nutnosti vyhľadávania v UDDI registri.[9]

1.1.2.5 Jazyky XML, XSD, XSLT

Jayzk XML (Extensible Markup Language), ktorý umožňuje pridávať inteligenciu do surových dát dokumentov, je značkovací jazyk. Znamená to, že je schopný priraďovať význam a kontext ľubovoľnej informácii prenášanej internetovými protokolmi. Z XML jazyka vznikol jazyk XSD (XML Schema Definition Language) a taktiež jazyk XSLT (XSL Transformation Language). Obidva jazyky sa stali dôležitými časťami jadra XML sady.

V praxi tvorí jazyk XML štruktúru a formát posielaných správ, ktoré sa prenášajú medzi službami, integrita a validita dát v správach je zabezpečená jazykom XSD a jazyk XSLT zabezpečuje komunikáciu medzi nezlučiteľnými reprezentantmi dát a mapovanie schém.

[11] *LIBERTY, J. , HURWITZ, D. 2003. Programming ASP.NET. Beijing. Cit.[2014-06-03] Dostupné online: <http://www.google.sk/books?hl=sk&lr=&id=7lf1cC6SXQ0C&oi=fnd&pg=PR9&dq=asp.net&ots=Ei42sIjkft&sig=euz3qhtEjL1WD7bXV15pRKZuqYc&redir_esc=y#v=onepage&q=asp.net&f=false>.*

1.1.2.6 *Úlohy webových služieb*

Činnosti vykonávané webovými službami sú rôzne, napr. prenášanie, prijímanie, iniciácia správy. Tieto aktivity sú vykonávané v rámci určitých rolí. Základné role alebo úlohy, v ktorých služby vystupujú sú žiadateľ, poskytovateľ a sprostredkovateľ [5].

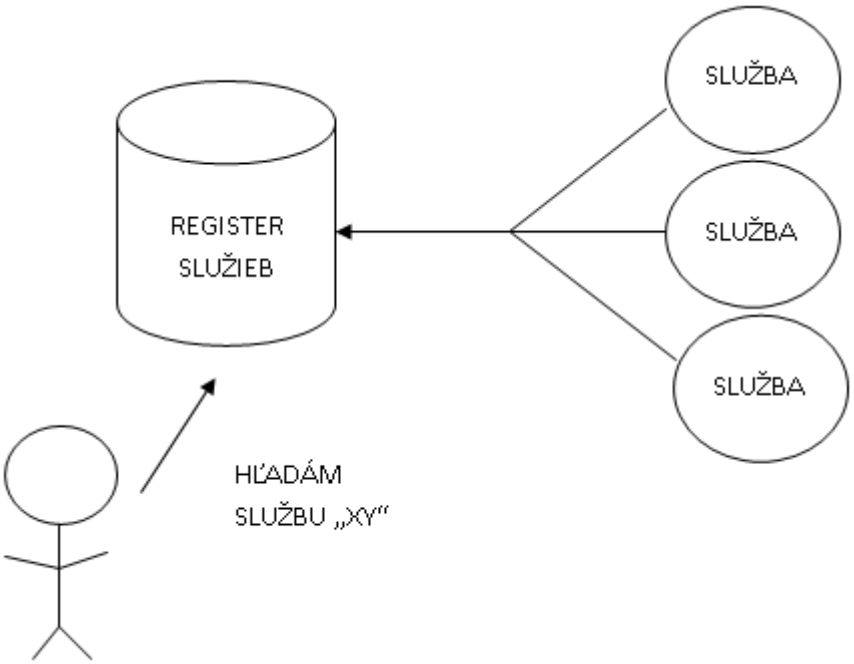
1.1.2.6.1 Rola poskytovateľa

Poskytovateľ je zvyčajne rola vyvolaná externým zdrojom, napríklad žiadateľom. Poskytovateľ sprístupňuje popis služby a tým poskytuje informácie o vlastných funkciách a svojich činnostiach. Podobá sa to úlohe servera v jednoduchej architektúre klient - server. Je však nutné rozlišovať entitu poskytovateľa a agenta poskytovateľa služieb. Agent poskytovateľa služieb je služba samotná, ktorá iba reprezentuje firmu či jednotlivca, ale entita poskytovateľa je určitý konkrétny jednotlivec alebo spoločnosť, ktorá danú službu vykonáva.

1.1.2.6.2 Rola žiadateľa

Rola žiadateľ služieb je rozdelená na termíny entita žiadateľ služieb, ktorého charakterizuje jednotlivec alebo firma a agent žiadateľa služieb, termín charakterizovaný samotnou webovou službou. Webová služba je v roli žiadateľa práve vtedy, ak volá poskytovateľa služby zaslaním správy a ak prehľadáva a zhodnocuje najpriaznivejšieho poskytovateľa služieb procesom porovnávania dostupných charakteristík služieb. Správa teda obsahuje požiadavky na poskytovateľov, ktorí túto službu vykonávajú a štruktúry správ sú definované v popisoch jednotlivých služieb.

Použitiu danej služby žiadateľom predchádza jej vyhľadávanie. V registri služieb sú informácie o službách a poskytovateľoch, ale tiež informácie potrebné pre komunikáciu medzi poskytovateľom a žiadateľom. Proces vyhľadania a použitia služby je znázornený na nasledujúcom grafe.

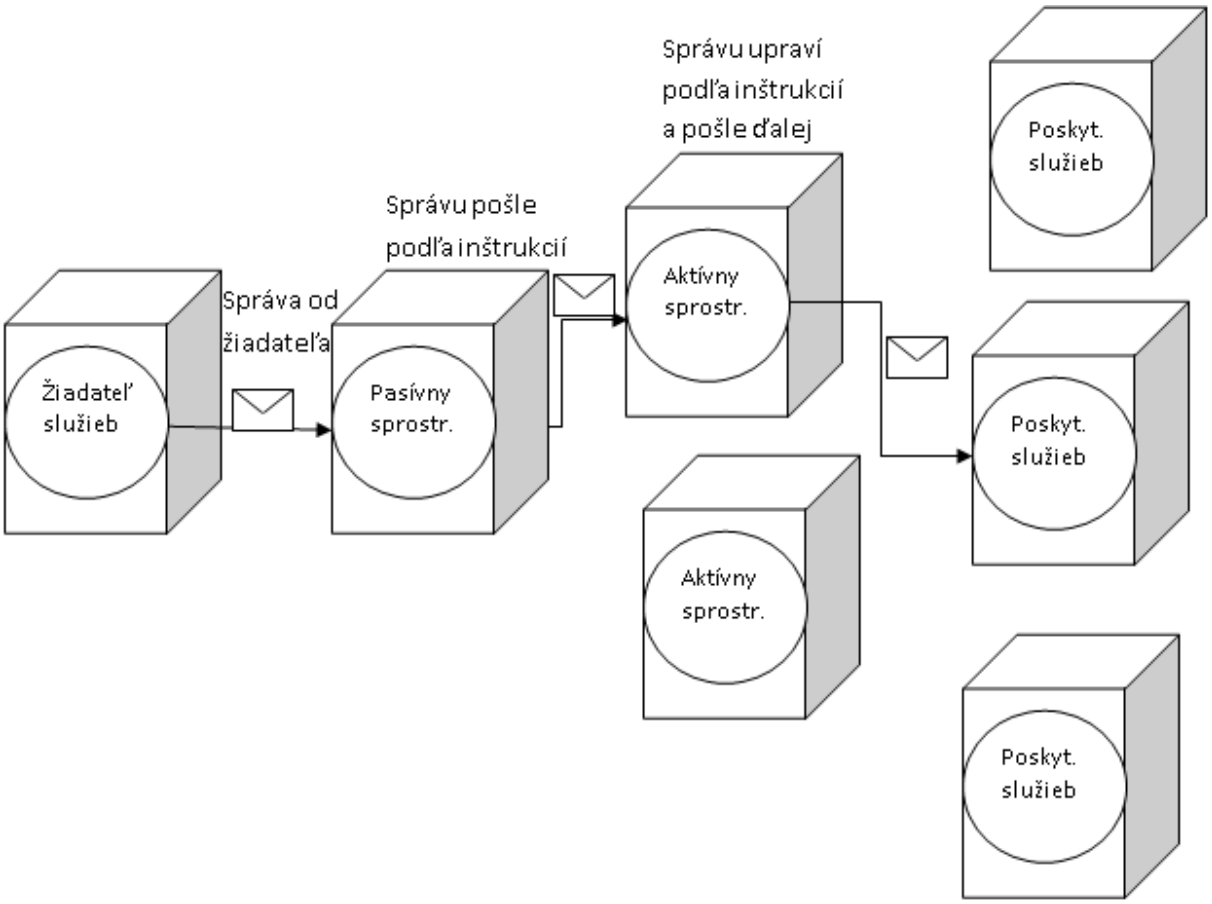


Obr.2: Umiestnenie popisov služieb[5]

1.1.2.6.3 Rola sprostredkovateľa

Vo webových službách existuje ešte jedna podstatná rola. Rola sprostredkovateľa smeruje a spracováva správy po jej odoslaní a pred prijatím do konečného cieľa, teda počas jej transportu. Sprostredkovateľov rozdeľujeme na dva základné typy: aktívny a pasívny. Aktívny sprostredkovateľ je ten, ktorý smeruje správu konečnému cieľu tak, že obsah správy spracuje a pozmení. V praxi to znamená, že sprostredkovateľ nazrie do hlavičky SOAP v správe a vykoná inštrukcie podľa toho, aké informácie tam nájde a pripíše alebo pozmení dáta.

Pasívny sprostredkovateľ má za úlohu smerovanie správ do ďalších uzlov. Využíva informácie hlavičiek správ SOAP za účelom identifikácie cesty správy, alebo používa vlastnú logiku smerovania tak, že určuje vykonanie požiadaviek jednotlivým komunikačným uzlom. Nazývame ho pasívnym práve pre skutočnosť, že obsah správy nemení. Obr. 3 zobrazuje aktívneho aj pasívneho sprostredkovateľa.



Obr. 3 – Aktívny aj pasívny sprostredkovatelia[5]

1.1.2.7 Webové služby v praxi

Role, ktoré sme popisovali – sprostredkovateľ, poskytovateľ, žiadateľ, sú všeobecné a nezávislé od typu funkcionality, ktorú daná webová služba poskytuje. Tieto role sú vlastne všeobecné stavy v akých môže služba vystupovať. V skutočnom svete postupne vznikli triedy či modely služieb založené na povahe aplikačnej logiky. Stalo sa tak kvôli rôznym rolám, ktoré na seba služby preberali. Medzi prvý typ služieb sa zaraďujeme model služieb riadenia. Tento primárny typ zapuzdruje skupinovú logika riadenia v bezpečne definovanom funkčnom rozhraní. Spustenie služby tohto typu nemusí byť izolované, pretože sú zvyčajne časťou väčšieho celku - viacerých služieb, a preto nadväzujú alebo súvisia s inými službami, aj keď sú ideálne nezávislé.

Využitie modelu služieb riadenia [5]:

- reprezentácia spoločnej entity alebo sady informácií,
- reprezentácia logiky riadiaceho procesu,
- základne stavebné jednotky pre reprezentáciu logiky riadenia,
- súčasť tvorby služieb.

Ďalší model s názvom pomocné služby sú služby, ktoré poskytujú všeobecnú funkcionalitu.

Dôvody využitia pomocných služieb [5]:

- vysoká miera autonómie,
- propagácia vlastnosti vnútornej spolupráce v SOA,
- umožňujú opätovné použitie,
- sprostredkujú iné služby nezávislé na konkrétnom riešení.

Posledný model sa často nazýva model služieb správ. Tieto služby sú organizované v sadách nezávislých služieb. Každá sada má účasť na celkovej úlohe riadenia. Služba správy plní úlohu skladania a koordinácie, ktoré je možné priradiť ako hlavné funkcie vyhradenej službe alebo službe, ktorá umožňuje spúšťanie úlohy nezávisle.

Využitie a vlastnosti služby správy [5]:

- tvorba príležitostí na znovu použitie,
- implementácia a podpora princípov komponovateľnosti.

1.1.2.8 ASP.NET technológia

Existuje viacero platforiem, na ktorých je možné vyvíjať webové služby. ASP.NET je platforma, ktorá bola použitá pri vývoji webovej služby v tejto práci.

Spoločnosť Microsoft začala s vývojom technológie ASP.NET v roku 2000 pod názvom ASP+. Táto technológia predstavuje časť projektu .NET, ktorý obsahuje veľkú sadu programovacích tried navrhnutých tak, aby zabezpečili a uspokojili špecifické programovacie potreby.

.NET je charakteristický:[7] :

- 5 oficiálnymi programovacími jazykmi (C#, Visual Basic .NET, Managed C++, J#, Jscript .NET),
- množstvom knižníc, známymi pod pojmom FCL(Framework Class Library),
- ako súčasť Microsoft Server 2003.

„ASP.NET je „server-side“ platforma pre webové aplikácie.“[12] Pomocou tejto platformy je možné vyvíjať dynamické webové stránky, webové aplikácie a webové služby. Táto technológia umožňuje programátorom vyvíjať ASP.NET aplikácie používaním ktoréhokoľvek programovacieho jazyka podporujúceho .NET. Zároveň sú ASP.NET aplikácie schopné spracovávať SOAP správy využívaním SOAP rozšírenia.

1.1.2.9 HTML klient

Aby mohla webová služba efektívne komunikovať s jej používateľom, je potrebné vytvoriť „user-friendly“ klienta. Webová služba v tejto práci komunikuje s klientom naprogramovaným v HTML jazyku.

„HTML definuje flexibilný, praktický a prenosný formát dokumentov pre internet[13]. Každý HTML dokument sa okrem textu skladá aj z tagov „<>“ a inštrukcií v nich, ktoré napovedajú webovému prehliadaču, ako zobraziť text. Text je napísaný medzi začiatočným tagom a konečným tagom, odlíšeným „/“. Každý HTML dokument začína <html> tagom a tiež obsahuje dve časti: <head> a <body>. Informácie o dokumente sú zapísané medzi tagmi <head> a informácie ktoré sa zobrazujú v okne prehliadača sú zapísané medzi tagmi <body>.[8]

[12]LIBERTY, J. , HURWITZ, D. 2003. Programming ASP.NET. Beijing. Cit.[2014-06-03] Dostupné na internete: <http://www.google.sk/books?hl=sk&lr=&id=7lf1cC6SXQ0C&oi=fnd&pg=PR9&dq=asp.net&ots=Ei42sIjkft&sig=euz3qhtEjL1WD7bXV15pRKZuqYc&redir_esc=y#v=onepage&q=asp.net&f=false>.

[13]MUSCIANO, C. , KENNEDY B. 2007. Creating Effective Web Pages: HTML&XHTML. USA. Cit[2014-08-11]. Dostupné na internete <http://www.oreilly.de/catalog/html6/chapter/ch02.pdf>.

Práve vďaka schopnosti HTML kódu zobrazovať text podľa inštrukcií zapísaných medzi riadkami kódu sa často HTML jazyk využíva ako klient pre webové služby. HTML klient vytvára „user-friendly“ dizajn, s ktorým je jednoduché pracovať a komunikuje užívateľské požiadavky volaným webovým službám.

2 Ciele a metodika práce

V tejto kapitole sú stručne zhrnuté ciele a postupy ktoré boli použité pri tvorení práce.

2.1 Ciele práce

Ciele diplomovej práce sú zamerané na prieskum v oblasti využitia webových služieb v bankovom sektore, prehĺbenie poznatkov o úročení v bankách a návrh a implementácia webovej služby riešiacej danú problematiku.

2.1.1 Čiastkový cieľ práce 1

Prieskum súčasného stavu využitia webových služieb v úrokovej správe bežného bankového účtu je prvý čiastkový cieľ práce. Zahrňuje vyhľadanie a štúdium ponúkaných bankových procesov a ich implementácie týkajúcej sa úrokovej správy bežného účtu na webových serveroch vybraných bánk pôsobiacich na Slovensku, zhodnotenie počtu dostupných bankových aplikácií prístupných pre bežného používateľa pracujúcich s úrokovou správou účtov, ďalej prieskum a zhodnotenie ich funkcií, možného využitia a klientsku prístupnosť.

2.1.2 Čiastkový cieľ práce 2

Druhým cieľom tejto práce je stručne opísať a vysvetliť teoretické východiská týkajúce sa úročenia v bankách a teoretické východiská webových služieb. Tieto teoretické východiská majú za úlohu priblížiť a objasniť základné poznatky z týchto dvoch odlišných oblastí, aby bolo možné následne pochopiť praktickú implementáciu poznatkov vo forme webovej služby.

2.1.3 Čiastkový cieľ práce 4

Posledným cieľom je vytvorenie webovej služby, ktorá uspokojivo poskytuje úrokovú správu bežného bankového účtu v prípade že takáto aplikácia neexistuje alebo nie je poskytovaná bankovým sektorom na Slovensku, alebo rieši zlepšenie a rozšírenie funkčnosti súčasných webových služieb v úrokovej správe bežného bankového účtu.

2.2 Metodika práce

V tejto podkapitole sú stručne zhrnuté a vysvetlené jednotlivé metódy použité pri tvorbe diplomovej práce.

2.2.1 Metóda analýzy

Metóda analýzy je jednou zo základných metodík použitých v tejto práci pri analýzach odborných publikácií, článkov a materiálov. Touto metódou boli kriticky analyzované poznatky z oblastí webových služieb a úrokovania. Väčšie témy tejto práce boli v teoretických východiskách rozdelené na menšie celky, charakteristické prvky a znaky aby sa takýmto spôsobom z veľkých a neforemných tém formovali podstatné vlastnosti a prvky jednotlivých oblastí.

2.2.2 Metóda dedukcie

Metódou dedukcie sa zo všeobecných faktov, vlastností, vzťahov a štruktúr o úročení a o webových službách vytvárali konkrétne malé čiastkové závery, dôsledky a riešenia. Metóda dedukcie bola použitá pri dedukovaní konkrétnych vzorcov výpočtu jednotlivých úrokov v teoretických východiskách.

2.2.3 Metóda komparácie

Ďalej bola použitá metóda komparácie teda porovnávania faktov, vzťahov a štruktúr so zreteľom na objasnenie rozdielov a podobností určitých procesov, funkcií a vlastností týkajúcich sa poskytovaných webových služieb využívaných v bankovom sektore a želanej

webovej službe, ktorá má za úlohu poskytovať úrokovú správu bežného bankového účtu. Táto metodika bola použitá pri zhodnotení výsledkov práce a pomáha pochopiť potenciál súčasného stavu riešenej problematiky rovnako ako aj rozdiely a podobnosti medzi novovytvorenou webovou službou a poskytovanými úrokovými aplikáciami v bankovom sektore.

2.2.4 Metóda abstrakcie

Metóda abstrakcie bola v práci použitá za účelom výberu vlastností a vzťahov, ktoré sú pre daný jav podstatné a odstránenia tých vlastností, ktoré zabraňujú preniknutiu k jadru skúmaných problémov. Každá oblasť, jav či problém zahŕňa množstvo vlastností a vzťahov, ktoré pre ich skúmanie nie sú dôležité. Metóda abstrakcie takéto nedôležité alebo nepodstatné fakty a vzťahy redukuje a vynecháva. Táto metóda bola využitá pri abstrahovaní najdôležitejších vlastností a prvkov webovej služby a ich implementácii v kapitole Výsledky práce.

2.2.5 Metóda syntézy

Po tom, čo boli témy a problémy tejto práce analyzované, poznatky porovnávané, dedukované a abstrahované, bolo potrebné nájsť ich prienik nie len v teoretickej ale aj praktickej rovine. Metóda syntézy je typická hľadaním súvislostí medzi jednotlivými oblasťami a prepájaním znakmi vykazujúcimi podobnosť alebo súvislosť. Táto metóda bola použitá v kapitole Výsledky práce, ktorá obsahuje syntézu poznatkov z oblasti webových služieb a úrokovania.

2.2.6 Metóda indukcie

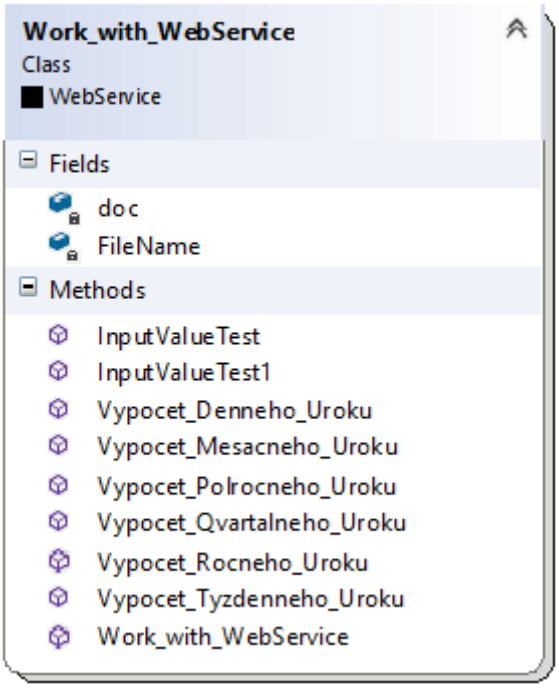
Metóda indukcie prispela pri skúmaní a analyzovaní poznatkov oboch oblastí, keď sa z faktov a vzťahov vytvárali všeobecné pravidlá, zákonitosti a závery.

3 Výsledky práce

Výsledkom práce a riešením problematiky využitia webových služieb pri úrokovej správe bežného bankového účtu je webová služba, ktorá je naprogramovaná v programovacom jazyku C# na platforme a s využitím servera ASP.NET. Používateľ sa prihlási do webovej služby zadaním čísla svojho účtu a rodného čísla. Ak sú údaje správne, bude si môcť vybrať z šiestich rôznych operácii, ktoré vykoná s jeho vkladom webová služba. Konkrétne sú to výpočty ročného, polročného, kvartálneho, mesačného, týždenného a denného úroku. Výsledky operácií sú zobrazené v okne klienta, v dialógovom okne a taktiež je vytvorený súbor s číslom účtu, výškou vkladu, typom úroku a časovou stopou v názve súboru, do ktorého je informácia o zvolenom úroku zapísaná. Implementovaný program je popísaný v nasledujúcich podkapitolách.

3.1 Diagram tried

Diagram tried webovej služby „Martin_Huzvar_WebService“ zobrazuje hlavnú triedu „Work_with_WebService“, ktorá obsahuje členské metódy „InputValueTest“ a „InputValueTest1“, slúžiace na preverenie správnosti zadaných dát, ďalej šesť členských metód obsahujúcich logiku výpočtu jednotlivých úrokov a nakoniec funkcia „Work_with_WebService“, ktorá prečíta a uloží obsah databázy klientskych účtov, uložených na lokálnom disku servera, na ktorom funguje webová služba.



Obr. 4: Diagram tried webovej služby

3.2 Implementácia webovej služby

Na nasledujúcich stranách je zobrazená a okomentovaná implementácia webovej služby. V jej hlavnej triede „Work_with_WebService“ sa nachádza šesť členských webových metód, ktoré pracujú s vkladom a úrokom používateľa, dve členské metódy, slúžiace na testovanie vstupov zadaných používateľom a konštruktor triety „Work_with_WebService“.

3.2.1 Prečítanie XML dokumentu

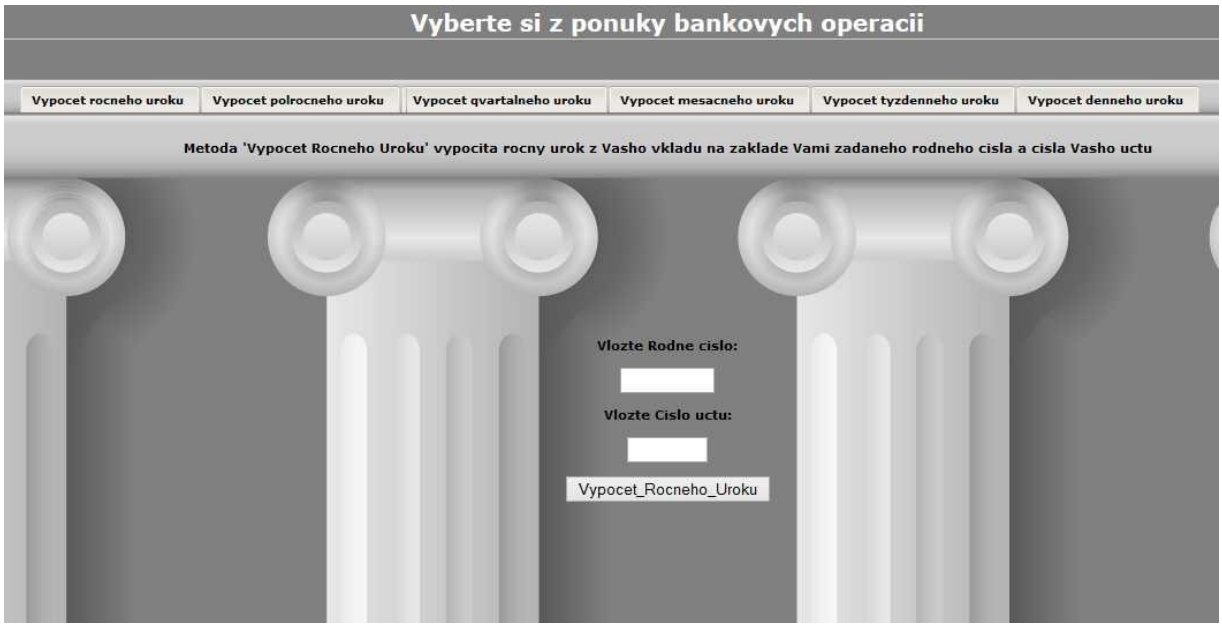
„Work_with_WebService“ je názov webovej služby, ktorá vlastní a zahŕňa všetky naprogramované objekty. Táto webová služba obsahuje 6 metód na výpočet rôznych úrokov a 2 funkcie na testovanie vstupných hodnôt. Spustenú webovú službu na ASP.NET klientovi zobrazuje Obr.10. Členská premenná „Filename“ obsahuje cestu k súboru s klientskymi dátami uloženými na disku. Metóda „Load“ objektu „doc“ prečíta a uloží klientske dáta ako XML dáta a umožní tak službe pracovať s nimi.

```
public class Work_with_WebService : System.Web.Services.WebService
   {
      private string FileName =  @"C:\Users\Martin Huzvar\Desktop\Martin_Huzvar_WebServices
\Martin_Huzvar_WebService\bin\Data.xml";
      private XmlDocument doc;

      public Work_with_WebService()
      {
         doc = new XmlDocument();
         doc.Load(FileName);
      }
```

3.2.2 *Výpočet ročného úroku*

Webová metóda s názvom „Vypocet_Rocneho_Uroku“ je metóda, ktorá keď je volaná, prijme a spracuje dve premenné – „Rodne_Cislo“ a „Cislo_Uctu“, obidve vo forme reťazcov. Tieto dáta sú zadané používateľom a poslané HTML klientom do tejto metódy pri jej volaní. HTML klient s formulárom na zadávanie používateľských údajov je zobrazený na obr. 5.

```
[WebMethod(Description = "Vypocet rocneho uroku z Vasho vkladu na zaklade Vami zadaneho rodneho
cisla a cisla Vasho uctu")]
      public string Vypocet_Rocneho_Uroku(string Rodne_Cislo, string Cislo_Uctu)
      {
```



Obr.5 : HTML klient, Vypočet roč. úr.

Nasledujúci kód volá funkcie „InputValueTest“ a „InputValueTest1“, ktoré otestujú zadané vstupné hodnoty „Rodne_Cislo“ a „Cislo_Uctu“. V prípade že vstupné dáta testom neprešli, systém presmeruje stránku prehliadača na úvodnú stránku a upozorní používateľa na nesprávne hodnoty rodného čísla alebo čísla účtu. Ak zadané dáta prejdú testami, aplikácia pokračuje ďalšími úkonmi.

```
            if (InputValueTest(Rodne_Cislo, "Vypocet_Rocneho_Uroku") == 0 ||
InputValueTest(Cislo_Uctu, "Vypocet_Rocneho_Uroku") == 0)
        {
            System.Diagnostics.Process.Start ("http://localhost:4641/Web_Files/index.html");
            return "chybny vstup, budete presmerovany na uvodnu stranku";
        }

        else if (InputValueTest1(Rodne_Cislo, Cislo_Uctu) == 0)
        {
            System.Diagnostics.Process.Start("http://localhost:4641/Web_Files/index.html");
            return "chybny vstup, budete presmerovany na uvodnu stranku";
        }
        else
        {
```

Následne premenná “path1” definuje miesto, kde sa bude neskôr ukladať XML dokument obsahujúci výsledky volaných metód. Objekt „sw1“ je objektom triedy „StreamWriter“, ktorý slúži na zápis dát do súborov. Objekt „sw1“ je inicializovaný a zadefinovaný tromi atribútmi. Textový reťazec „Path1“ nasmeruje objekt „sw1“ na miesto, kde bude XML súbor vytvorený, atribút „false“ špecifikuje postup v prípade, že daný súbor už existuje. V takom prípade bude tento súbor prepísaný. Posledný atribút „Encoding.ASCII“ definuje typ znakov zapísaných do súboru. Ďalej sú v kóde definované textové reťazce, ktoré budú použité neskôr.

```
            string path1 = @"C:\Users\Martin Huzvar\Desktop\Martin_Huzvar_WebServices\
Martin_Huzvar_WebService\method_output.xml";
            StreamWriter sw1 = new StreamWriter(path1, false, Encoding.ASCII);

            string x = "";
            string return_x = "";
            string c = "";
            string d = "";
```

Objekt „root“ je objektom triedy „XmlElement“ a je do neho uložený koreňový element XML dokumentu načítaný v objekte „doc“. Ďalej, objekty „nodeList“ a „nodeList2“ sú objekty triedy „XmlNodeList“. Do objektu „nodeList“ sú zapísané hodnoty uzla „vklad“ z objektu „root“. Tieto dáta sú vyfiltrované a vyhľadané podľa elementov

„Rodne_Cislo“ a „Cislo_Uctu“ ktoré sú dcérskymi elementami elementu „ucet“. Element „ucet“ je zas uložený v rodičovskom elemente „databaza“. Objekt „nodeList2“ získa rovnakým spôsobom hodnotu elementu „mesac_urok“. Číselná premenná „nodeListCount“ prijme číslo, ktoré je výsledkom inštančnej metódy „Count“ volanej objektom „nodeList“. Táto metóda spočíta počet hodnôt v objekte „nodeList“.

```
                XmlElement root = doc.DocumentElement;
        XmlNodeList nodeList;
        XmlNodeList nodeList2;
        nodeList = root.SelectNodes("/databaza/ucet[rod_c=" + Rodne_Cislo + " ][ cislo_u=" +
Cislo_Uctu + "]/vklad");
        nodeList2 = root.SelectNodes("/databaza/ucet[rod_c=" + Rodne_Cislo + " ][ cislo_u=" +
Cislo_Uctu + "]/mesac_urok");
        int nodeListCount = nodeList.Count;
        int i = 0;
```

Ďalej, inštančná metóda „write“ volaná objektom „sw1“, ktorý už bol inicializovaný a pozná súbor, do ktorého má dáta zapísať, spôsob, ako ich zapísať a kódovanie znakov, ktoré má použiť, zapíše do súboru nasledujúci text: „`Hodnota rocneho uroku klienta s rodnym cislom " + Rodne_Cislo + " pri zlozenom uroceni s urokovou periodou mesiac je „`“. Premenná „Rodne_Cislo“ bola zadaná používateľom služby. Neskoršie bude do tohto súboru zapísaná aj hodnota výsledku metódy, teda hodnota úroku, a celý textový reťazec bude zobrazený na výslednej webovej stránke.

```
                sw1.Write("\n\n"+"Hodnota rocneho uroku klienta s rodnym cislom " + Rodne_Cislo
+ " pri zlozenom uroceni s urokovou periodou mesiac je " );
```

Následne je spustený cyklus „while“ ktorý sa ukončí až vtedy, keď sa vykonajú inštrukcie toľkokrát, koľko je uzlov v „nodeList“. Položky objektu „nodeList“ a „nodeList2“, teda vklady a mesačné úroky klienta, sú po jednom získavané inštančnou metódou „Item“ a ukladané do objektov „title“ a „title2“. Ďalej sú hodnoty „title“ a „title2“ vrátené metódou „InnerText“ a uložené do textového reťazca „x“ a „d“. Funkciou „Convert.ToDouble()“ sú hodnoty vkladu a mesačnej úrokovej miery konvertované z textového reťazca do číselného formátu „double“ a uložené do premenných „vklad“ a „mesac_ur_miera“. Následne sa ročný úrok vypočíta ako hodnota mesačnej úrokovej miery, ku ktorej je pripočítané číslo 1 a to celé je umocnené na dvanástu, keďže v jednom roku je 12 mesiacov a táto hodnota je vynásobená vkladom. Výsledok predstavuje úrok po 12 mesiacoch spolu s vkladom, preto je na záver vklad odpočítaný a získaná je čistá hodnota ročného úroku. Hodnota ročného

úroku je zaokrúhlená na dve desatinné miesta inštančnou metódou „Round“ objektu „Math“ a zapísaná do súboru metódou „Write“ objektu „sw“ a objektu „sw1“.

```
                    while (i < nodeListCount)
            {
                XmlNode title = nodeList.Item(i);
                XmlNode title2 = nodeList2.Item(i);
                x += title.InnerText + "\n";
                d += title2.InnerText + "\n";
                double vklad = Convert.ToDouble(x);
                double mesac_ur_miera = Convert.ToDouble(d);
                double rocny_urok = Math.Round((vklad * Math.Pow(1 + mesac_ur_miera, 12) -
vklad),2);
                c = Convert.ToString(rocny_urok);
                return_x += title.OuterXml;

                string path = @"C:\Users\Martin Huzvar\Desktop\Martin_Huzvar_WebServices\
Martin_Huzvar_WebService\bin\cislo_uctu-" + Cislo_Uctu + "_vklad-" + vklad + "_rocny_urok_cas" +
DateTime.Now.ToString("dd-MM-yyyy_hh-mm-ss") + ".xml";
                StreamWriter sw = new StreamWriter(path, false, Encoding.ASCII);


                sw.Write("<output>" + "\n\n");
                sw.Write(c);
                sw.Write("\n</output>");
                sw.Close();


                sw1.Write(c);
                sw1.Close();
                i++;
            }
```

Na konci tejto webovej metódy na výpočet ročného úroku je zobrazené dialógové okno inštančnou metódou „Show“ triedy „MessageBox“, v ktorom je zobrazený výsledok výpočtov. Po stlačení tlačidla “OK” je webová aplikácia presmerovaná a hodnota ročného úroku je zobrazená v “TextArea” webovej aplikácie „index1.html“.

```
MessageBox.Show("Hodnota rocneho uroku klienta s rodnym cislom " + Rodne_Cislo + " pri zlozenom
uroceni s urokovou periodou mesiac je\n\n" + c, "Vystup metody 'Vypocet_Rocneho_Uroku'",
MessageBoxButtons.OK, MessageBoxIcon.Exclamation, MessageBoxDefaultButton.Button1,
MessageBoxOptions.ServiceNotification);
            System.Diagnostics.Process.Start("http://localhost:4641/Web_Files/index1.html");
                return return_x;
            }
        }
```

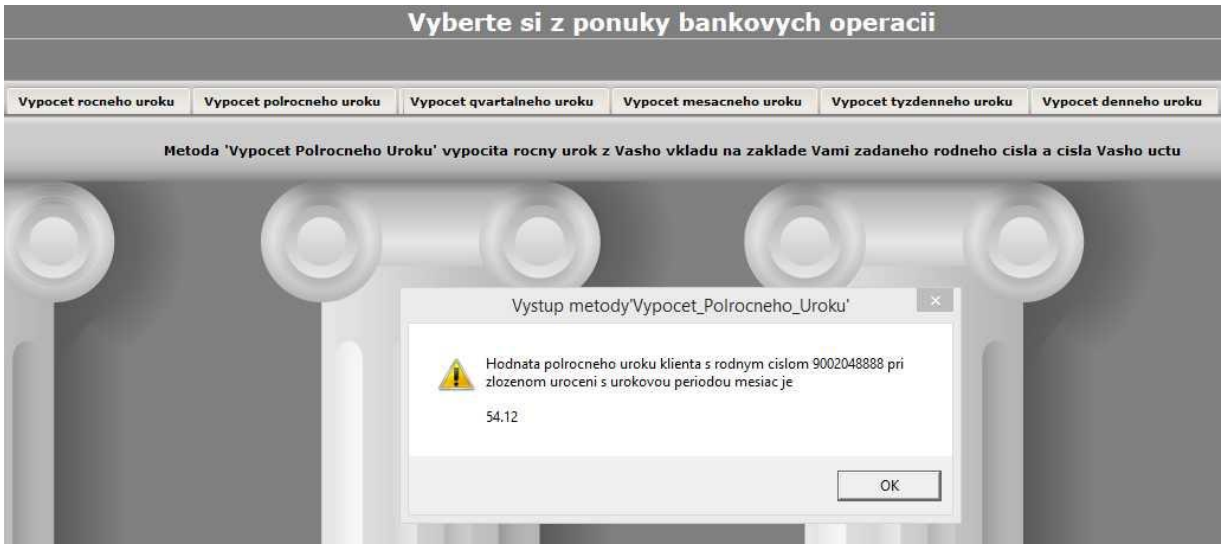
3.2.3 *Výpočet polročného úroku*

Webová metóda „Vypocet_polrocneho_uroku“ sa procesmi a postupmi podobá na webovú metódu „Vypocet_rocneho_uroku“. Spôsob získania vstupných údajov klienta, spracovania XML dokumentu či overenia správnosti vložených dát je rovnaký. Rozdiel je

v spôsobe výpočtu polročného úroku. V zvýraznenom programovom kóde vidieť, že v cykle „while“ je XML dokument rovnako prečítaný a rozdelený do jednotlivých uzlov, ale pri výpočte polročného úroku je použitý iný vzorec. Polročný úrok je vypočítaný ako šiesta mocnina mesačnej úrokovej miery, ku ktorej je pripočítané číslo 1 a to celé je vynásobené vkladom klienta. Od toho čiastkového výsledku je odpočítaný vklad. Výsledkom je polročný úrok, ktorý je rovnako spracovaný, a zobrazený v dialógovom okne ako aj v novej webovej aplikácii „index1.html“ ako ročný úrok. Táto situácia je zobrazená na obr. 6.

```
[WebMethod(Description = "Vypocet polrocneho uroku z Vasho vkladu na zaklade Vami zadaneho rodneho cisla a cisla Vasho uctu")]
      public string Vypocet_Polrocneho_Uroku(string Rodne_Cislo, string Cislo_Uctu)
      {
         if (InputValueTest(Rodne_Cislo, "Vypocet_Polrocneho_Uroku") == 0 || InputValueTest(Cislo_Uctu, "Vypocet_Polrocneho_Uroku") == 0)
         {
System.Diagnostics.Process.Start("http://localhost:4641/Web_Files/index.html");
            return "chybny vstup, budete presmerovany na uvodnu stranku";
         }
         else if (InputValueTest1(Rodne_Cislo, Cislo_Uctu) == 0)
         {
System.Diagnostics.Process.Start("http://localhost:4641/Web_Files/index.html");
            return "chybny vstup, budete presmerovany na uvodnu stranku";
         }
         else
         {
            string path1 = @"C:\Users\Martin Huzvar\Desktop\Martin_Huzvar_WebServices\Martin_Huzvar_WebService\method_output.xml";
            StreamWriter sw1 = new StreamWriter(path1, false, Encoding.ASCII);
            string x = "";
            string return_x = "";
            string c = "";
            string d = "";
            XmlNodeList nodeList;
            XmlNodeList nodeList2;
            XmlElement root = doc.DocumentElement;
            nodeList = root.SelectNodes("/databaza/ucet[rod_c=" + Rodne_Cislo + "][ cislo_u=" + Cislo_Uctu + "]/vklad");
            nodeList2 = root.SelectNodes("/databaza/ucet[rod_c=" + Rodne_Cislo + "][ cislo_u=" + Cislo_Uctu + "]/mesac_urok");
            int nodeListCount = nodeList.Count;
            int i = 0;
            sw1.Write("\n\n" + "Hodnota polrocneho uroku klienta s rodnym cislom " + Rodne_Cislo + " pri zlozenom uroceni s urokovou periodou mesiac je ");

            while (i < nodeListCount)
            {
               XmlNode title = nodeList.Item(i);
               XmlNode title2 = nodeList2.Item(i);
               x += title.InnerText + "\n";
               d += title2.InnerText + "\n";
               double vklad = Convert.ToDouble(x);
               double mesac_ur_miera = Convert.ToDouble(d);
```

```
            double polrocny_urok = Math.Round(( vklad * Math.Pow(1 + mesac_ur_miera, 6)
- vklad),2);
            c = Convert.ToString(polrocny_urok);
            return_x += title.OuterXml;
            string path = @"C:\Users\Martin
Huzvar\Desktop\Martin_Huzvar_WebServices\Martin_Huzvar_WebService\bin\cislo_uctu-" +
Cislo_Uctu + "_vklad-" + vklad + "_polrocny_urok_cas" + DateTime.Now.ToString("dd-MM-
yyyy_hh-mm-ss") + ".xml";
            StreamWriter sw = new StreamWriter(path, false, Encoding.ASCII);
            sw.Write("<output>" + "\n\n");
            sw.Write(c);
            sw.Write("\n</output>");
            sw.Close();
            sw1.Write(c);
            sw1.Close();
            i++;
        }


            MessageBox.Show("Hodnota polrocneho uroku klienta s rodnym cislom " + Rodne_Cislo + "
pri zlozenom uroceni s urokovou periodou mesiac je\n\n" + c, "Vystup metody
            'Vypocet_Polrocneho_Uroku'", MessageBoxButtons.OK, MessageBoxIcon.Exclamation,
MessageBoxDefaultButton.Button1, MessageBoxOptions.ServiceNotification);

System.Diagnostics.Process.Start("http://localhost:4641/Web_Files/index1.html");
            return return_x;
        }
    }
```



Obr. 6 : Výsledok v dialog. okne

3.2.4 *Výpočet kvartálneho úroku*

Webová metóda „Vypocet_Kvartalneho_Uroku“ využíva podobné procesy na získanie a spracovanie dát, ako webová metóda „Vypocet_Rocneho_Uroku“. Rozdiel je v použitom vzorci na výpočet kvartálneho úroku. Ten je vypočítaný ako 1 + mesačná úroková miera a to celé umocnené na tretiu, kvôli trom úrokovaným mesiacom. Tento medzivýsledok je

vynásobený vkladom. Tak vznikol úrok za kvartálne obdobie spolu s počiatočným vkladom. Od tohto čísla sa vklad odpočíta a vznikne čistý kvartálny úrok, ktorý webová služba spracuje a zobrazí podobne, ako v predchádzajúcich prípadoch.

```
[WebMethod(Description = "Vypocet kvartalneho uroku z Vasho vkladu na zaklade Vami zadaneho rodneho cisla a cisla Vasho uctu")]
      public string Vypocet_Kvartalneho_Uroku(string Rodne_Cislo, string Cislo_Uctu)
      {
         if (InputValueTest(Rodne_Cislo, "Vypocet_Kvartalneho_Uroku") == 0 || InputValueTest(Cislo_Uctu, "Vypocet_Kvartalneho_Uroku") == 0)
         {
System.Diagnostics.Process.Start("http://localhost:4641/Web_Files/index.html");
            return "chybny vstup, budete presmerovany na uvodnu stranku";
         }
                  else if (InputValueTest1(Rodne_Cislo, Cislo_Uctu) == 0)
         {
            System.Diagnostics.Process.Start("http://localhost:4641/Web_Files/index.html");
            return "chybny vstup, budete presmerovany na uvodnu stranku";
         }
         else
         {

            string path1 = @"C:\Users\Martin Huzvar\Desktop\Martin_Huzvar_WebServices\Martin_Huzvar_WebService\method_output.xml";
            StreamWriter sw1 = new StreamWriter(path1, false, Encoding.ASCII);
            string x = "";
            string return_x = "";
            string c = "";
            string d = "";
            XmlNodeList nodeList;
            XmlNodeList nodeList2;
            XmlElement root = doc.DocumentElement;
            nodeList = root.SelectNodes("/databaza/ucet[rod_c=" + Rodne_Cislo + "][ cislo_u=" + Cislo_Uctu + "]/vklad");
            nodeList2 = root.SelectNodes("/databaza/ucet[rod_c=" + Rodne_Cislo + "][ cislo_u=" + Cislo_Uctu + "]/mesac_urok");
            int nodeListCount = nodeList.Count;
            int i = 0;
            sw1.Write("\n\n" + "Hodnota kvartalneho uroku klienta s rodnym cislom " + Rodne_Cislo + " pri zlozenom uroceni s urokovou periodou mesiac je ");

            while (i < nodeListCount)
            {
               XmlNode title = nodeList.Item(i);
               XmlNode title2 = nodeList2.Item(i);
               x += title.InnerText + "\n";
               d += title2.InnerText + "\n";
               double vklad = Convert.ToDouble(x);
               double mesac_ur_miera = Convert.ToDouble(d);
               double kvartalny_urok = Math.Round((vklad * Math.Pow(1 + mesac_ur_miera, 3) - vklad),2);
               c = Convert.ToString(kvartalny_urok);
               return_x += title.OuterXml;
               string path = @"C:\Users\Martin Huzvar\Desktop\Martin_Huzvar_WebServices\Martin_Huzvar_WebService\bin\cislo_uctu-" + Cislo_Uctu + "_vklad-" + vklad + "_kvartalny_urok_cas" + DateTime.Now.ToString("dd-MM-yyyy_hh-mm-ss") + ".xml";
```

```
            StreamWriter sw = new StreamWriter(path, false, Encoding.ASCII);
            sw.Write("<output>" + "\n\n");
            sw.Write(c);
            sw.Write("\n</output>");
            sw.Close();
            sw1.Write(c);
            sw1.Close();
            i++;

        }

            MessageBox.Show("Hodnota kvartalneho uroku klienta s rodnym cislom " + Rodne_Cislo + "
pri zlozenom uroceni s urokovou periodou mesiac je\n\n" + c, "Vystup metody
            'Vypocet_Kvartalneho_Uroku'", MessageBoxButtons.OK, MessageBoxIcon.Exclamation,
MessageBoxDefaultButton.Button1, MessageBoxOptions.ServiceNotification);

System.Diagnostics.Process.Start("http://localhost:4641/Web_Files/index1.html");
            return return_x;
        }
    }
```

3.2.5 *Výpočet mesačného úroku*

Vo webovej metóde „Vypocet_mesacneho_uroku“ je proces získavania hodnoty mesačného úroku trochu jednoduchší. Keďže je úročiace obdobie jeden mesiac, na získanie mesačného úroku stačí vynásobiť vklad a mesačnú úrokovú mieru. V nižšie uvedenom kóde je cyklus „while“ obsahujúci formulu na výpočet mesačného úroku zvýraznený.

```
[WebMethod(Description = "Vypocet mesacneho uroku z Vasho vkladu na zaklade Vami zadaneho
rodneho cisla a cisla Vasho uctu")]
    public string Vypocet_Mesacneho_Uroku(string Rodne_Cislo, string Cislo_Uctu)
    {
        if (InputValueTest(Rodne_Cislo, "Vypocet_Mesacneho_Uroku") == 0 ||
InputValueTest(Cislo_Uctu, "Vypocet_Mesacneho_Uroku") == 0)
        {
System.Diagnostics.Process.Start("http://localhost:4641/Web_Files/index.html");
            return "chybny vstup, budete presmerovany na uvodnu stranku";
        }
        else if (InputValueTest1(Rodne_Cislo, Cislo_Uctu) == 0)
        {
            System.Diagnostics.Process.Start("http://localhost:4641/Web_Files/index.html");
            return "chybny vstup, budete presmerovany na uvodnu stranku";
        }
        else
        {
            string path1 = @"C:\Users\Martin
Huzvar\Desktop\Martin_Huzvar_WebServices\Martin_Huzvar_WebService\method_output.xml";
            StreamWriter sw1 = new StreamWriter(path1, false, Encoding.ASCII);
            string x = "";
            string return_x = "";
            string c = "";
            string d = "";
            XmlNodeList nodeList;
            XmlNodeList nodeList2;
```

```
            XmlElement root = doc.DocumentElement;
            nodeList = root.SelectNodes("/databaza/ucet[rod_c=" + Rodne_Cislo + "][ cislo_u=" +
Cislo_Uctu + "]/vklad");
            nodeList2 = root.SelectNodes("/databaza/ucet[rod_c=" + Rodne_Cislo + "][ cislo_u=" +
Cislo_Uctu + "]/mesac_urok");
            int nodeListCount = nodeList.Count;
            int i = 0;
            sw1.Write("\n\n" + "Hodnota mesacneho uroku klienta s rodnym cislom " + Rodne_Cislo + " pri
zlozenom uroceni s urokovou periodou mesiac je ");

            while (i < nodeListCount)
            {
                XmlNode title = nodeList.Item(i);
                XmlNode title2 = nodeList2.Item(i);
                x += title.InnerText + "\n";
                d += title2.InnerText + "\n";
                double vklad = Convert.ToDouble(x);
                double mesac_urok_miera = Convert.ToDouble(d);
                double mesac_urok = Math.Round((vklad * mesac_urok_miera),2);
                c = Convert.ToString(mesac_urok);
                return_x += title.OuterXml;
                string path = @"C:\Users\Martin
Huzvar\Desktop\Martin_Huzvar_WebServices\Martin_Huzvar_WebService\bin\cislo_uctu-" +
Cislo_Uctu + "_vklad-" + vklad + "_mesacny_urok_cas" + DateTime.Now.ToString("dd-MM-
yyyy_hh-mm-ss") + ".xml";
                StreamWriter sw = new StreamWriter(path, false, Encoding.ASCII);
                sw.Write("<output>" + "\n\n");
                sw.Write(c);
                sw.Write("\n</output>");
                sw.Close();
                sw1.Write(c);
                sw1.Close();
                i++;
            }
            MessageBox.Show("Hodnota mesacneho uroku klienta s rodnym cislom " + Rodne_Cislo + "
je\n\n" + c, "Vystup metody 'Vypocet_Mesacneho_Uroku'", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation, MessageBoxDefaultButton.Button1,
MessageBoxOptions.ServiceNotification);

System.Diagnostics.Process.Start("http://localhost:4641/Web_Files/index1.html");
            return return_x;
        }
    }
```

3.2.6 *Výpočet týždenného úroku*

Pri výpočte týždenného úroku sa vychádza z predpokladu, že sa úroky na účte pripisujú raz za mesiac, a to na konci mesiaca. Preto je týždenný úrok vlastne podielom mesačného úroku a štyroch týždňov. Vo webovej metóde „Tyzdenna_urokova_miera“ sa najprv do premennej „x“ načíta vklad klienta a potom sa do premennej „d“ načíta mesačná úroková miera. Ďalej je mesačná úroková miera predelená štyrmi týždňami, čo predstavuje týždennú úrokovú mieru. Tá je nakoniec prenásobená vkladom a vzniknutý výsledok predstavuje týždenný úrok. Zvýraznený kód obsahuje túto formulu.

```
        [WebMethod(Description = "Vypocet tyzdenneho uroku z Vasho vkladu na zaklade Vami zadaneho rodneho cisla a cisla Vasho uctu")]
    public string Vypocet_Tyzdenneho_Uroku(string Rodne_Cislo, string Cislo_Uctu)
    {
        if (InputValueTest(Rodne_Cislo, "Vypocet_Tyzdenneho_Uroku") == 0 || InputValueTest(Cislo_Uctu, "Vypocet_Tyzdenneho_Uroku") == 0)
        {
            System.Diagnostics.Process.Start("http://localhost:4641/Web_Files/index.html");
            return "chybny vstup, budete presmerovany na uvodnu stranku";
        }
        else if (InputValueTest1(Rodne_Cislo, Cislo_Uctu) == 0)
        {
            System.Diagnostics.Process.Start("http://localhost:4641/Web_Files/index.html");
            return "chybny vstup, budete presmerovany na uvodnu stranku";
        }
        else
        {
            string path1 = @"C:\Users\Martin Huzvar\Desktop\Martin_Huzvar_WebServices\Martin_Huzvar_WebService\method_output.xml";
            StreamWriter sw1 = new StreamWriter(path1, false, Encoding.ASCII);
            string x = "";
            string return_x = "";
            string c = "";
            string d = "";
            XmlNodeList nodeList;
            XmlNodeList nodeList2;
            XmlElement root = doc.DocumentElement;
            nodeList = root.SelectNodes("/databaza/ucet[rod_c=" + Rodne_Cislo + "][ cislo_u=" + Cislo_Uctu + "]/vklad");
            nodeList2 = root.SelectNodes("/databaza/ucet[rod_c=" + Rodne_Cislo + "][ cislo_u=" + Cislo_Uctu + "]/mesac_urok");
            int nodeListCount = nodeList.Count;
            int i = 0;
            sw1.Write("\n\n" + "Hodnota tyzdenneho uroku klienta s rodnym cislom " + Rodne_Cislo + " pri zlozenom uroceni s urokovou periodou mesiac je ");

            while (i < nodeListCount)
            {
                XmlNode title = nodeList.Item(i);
                XmlNode title2 = nodeList2.Item(i);
                x += title.InnerText + "\n";
                d += title2.InnerText + "\n";
                double vklad = Convert.ToDouble(x);
                double mesac_ur_miera = Convert.ToDouble(d);
                double tyzden_ur_miera = mesac_ur_miera  / 4;                double tyzden_urok = Math.Round((vklad * tyzden_ur_miera),2);
                c = Convert.ToString(tyzden_urok);
                return_x += title.OuterXml;
                string path = @"C:\Users\Martin Huzvar\Desktop\Martin_Huzvar_WebServices\Martin_Huzvar_WebService\bin\cislo_uctu-" + Cislo_Uctu + "_vklad-" + vklad + "_tyzdenny_urok_cas" + DateTime.Now.ToString("dd-MM-yyyy_hh-mm-ss") + ".xml";
                StreamWriter sw = new StreamWriter(path, false, Encoding.ASCII);
                sw.Write("<output>" + "\n\n");
                sw.Write(c);
                sw.Write("\n</output>");
                sw.Close();
                sw1.Write(c);
                sw1.Close();
                i++;
            }
```

```
            MessageBox.Show("Hodnota tyzdenneho uroku klienta s rodnym cislom " + Rodne_Cislo + "
pri zlozenom uroceni s urokovou periodou mesiac je \n\n" + c, "Vystup metody
'Vypocet_Tyzdenneho_Uroku'", MessageBoxButtons.OK, MessageBoxIcon.Exclamation,
MessageBoxDefaultButton.Button1, MessageBoxOptions.ServiceNotification);
            System.Diagnostics.Process.Start("http://localhost:4641/Web_Files/index1.html");
            return return_x;
        }
    }
```

3.2.7 *Výpočet denného úroku*

Podobne ako v prípade týždennej úrokovej miery je denná úroková miera vypočítaná z mesačnej úrokovej miery. Mesačná úroková miera je predelená číslom 30, ktoré reprezentuje 30 dní v mesiaci. Táto hodnota zobrazujúca už dennú úrokovú mieru je vynásobená vkladom. Zvýraznený kód zobrazuje proces výpočtu dennej úrokovej miery. Výsledok predstavuje denný úrok, ktorý je podobne ako v ostatných metódach spracovaný a prezentovaný používateľovi webovej služby. Na obr. 7 je zobrazený druhý spôsob, ako HTML klient zobrazuje výsledok webovej metódy, a to pomocou textovej oblasti v jeho HTML kóde.

```
[WebMethod(Description = "Vypocet denneho uroku z Vasho vkladu na zaklade Vami zadaneho
rodneho cisla a cisla Vasho uctu")]
    public string Vypocet_Denneho_Uroku(string Rodne_Cislo, string Cislo_Uctu)
    {
        if (InputValueTest(Rodne_Cislo, "Vypocet_Denneho_Uroku") == 0 || InputValueTest(Cislo_Uctu,
"Vypocet_Denneho_Uroku") == 0)
        {
System.Diagnostics.Process.Start("http://localhost:4641/Web_Files/index.html");
            return "chybny vstup, budete presmerovany na uvodnu stranku";
        }
        else if (InputValueTest1(Rodne_Cislo, Cislo_Uctu) == 0)
        {
System.Diagnostics.Process.Start("http://localhost:4641/Web_Files/index.html");
            return "chybny vstup, budete presmerovany na uvodnu stranku";
        }
        else
        {
            string path1 = @"C:\Users\Martin
Huzvar\Desktop\Martin_Huzvar_WebServices\Martin_Huzvar_WebService\method_output.xml";
            StreamWriter sw1 = new StreamWriter(path1, false, Encoding.ASCII);
            string x = "";
            string return_x = "";
            string c = "";
            string d = "";
            XmlNodeList nodeList;
            XmlNodeList nodeList2;
            XmlElement root = doc.DocumentElement;
            nodeList = root.SelectNodes("/databaza/ucet[rod_c=" + Rodne_Cislo + "][ cislo_u=" +
Cislo_Uctu + "]/vklad");
```

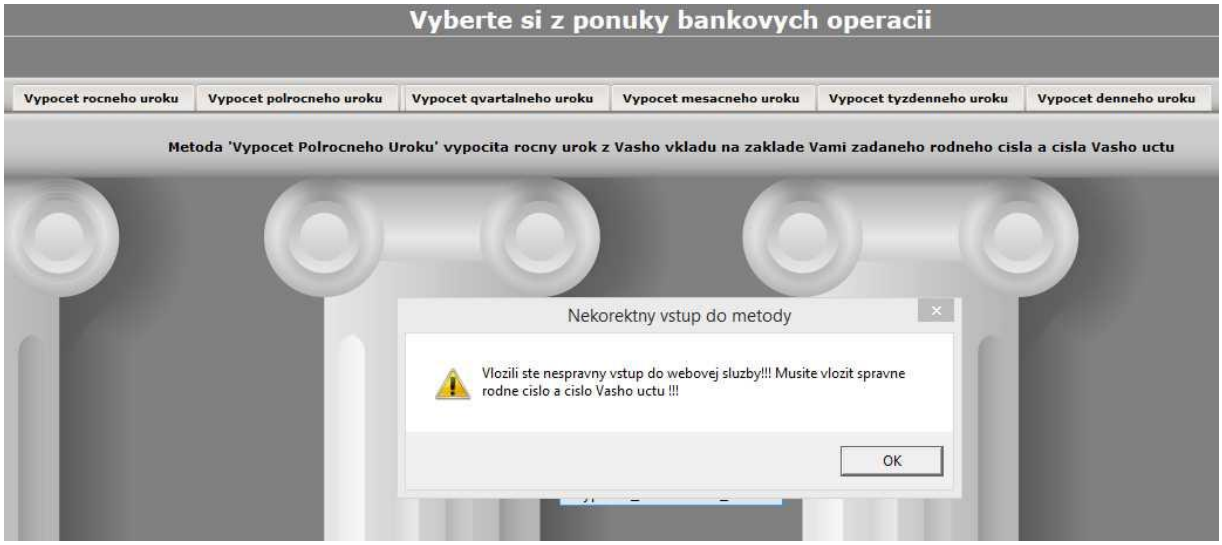
```
            nodeList2 = root.SelectNodes("/databaza/ucet[rod_c=" + Rodne_Cislo + "][ cislo_u=" + Cislo_Uctu + "]/mesac_urok");
            int nodeListCount = nodeList.Count;
            int i = 0;
            sw1.Write("\n\n" + "Hodnota denneho uroku klienta s rodnym cislom " + Rodne_Cislo + " pri zlozenom uroceni s urokovou periodou mesiac je ");

            while (i < nodeListCount)
            {
                XmlNode title = nodeList.Item(i);
                XmlNode title2 = nodeList2.Item(i);
                x += title.InnerText + "\n";
                d += title2.InnerText + "\n";
                double vklad = Convert.ToDouble(x);
                double mesac_ur_miera = Convert.ToDouble(d);
                double denna_ur_miera = mesac_ur_miera / 30;
                double denny_urok = Math.Round((vklad * denna_ur_miera),2);
                c = Convert.ToString(denny_urok);
                return_x += title.OuterXml;
                string path = @"C:\Users\Martin Huzvar\Desktop\Martin_Huzvar_WebServices\Martin_Huzvar_WebService\bin\cislo_uctu-" + Cislo_Uctu + "_vklad-" + vklad + "_denny_urok_cas" + DateTime.Now.ToString("dd-MM-yyyy_hh-mm-ss") + ".xml";
                StreamWriter sw = new StreamWriter(path, false, Encoding.ASCII);
                sw.Write("<output>" + "\n\n");
                sw.Write(c);
                sw.Write("\n</output>");
                sw.Close();
                sw1.Write(c);
                sw1.Close();
                i++;
            }
            MessageBox.Show("Hodnota denneho uroku klienta s rodnym cislom " + Rodne_Cislo + " pri zlozenom uroceni s urokovou periodou mesiac je \n\n" + c, "Vystup metody 'Vypocet_Denneho_Uroku'", MessageBoxButtons.OK, MessageBoxIcon.Exclamation, MessageBoxDefaultButton.Button1, MessageBoxOptions.ServiceNotification);
            System.Diagnostics.Process.Start("http://localhost:4641/Web_Files/index1.html");
            return return_x;
        }
    }
```

Obr. 7 : Textová oblasť s výsledkom

3.2.8 *Testovanie vstupných dát*

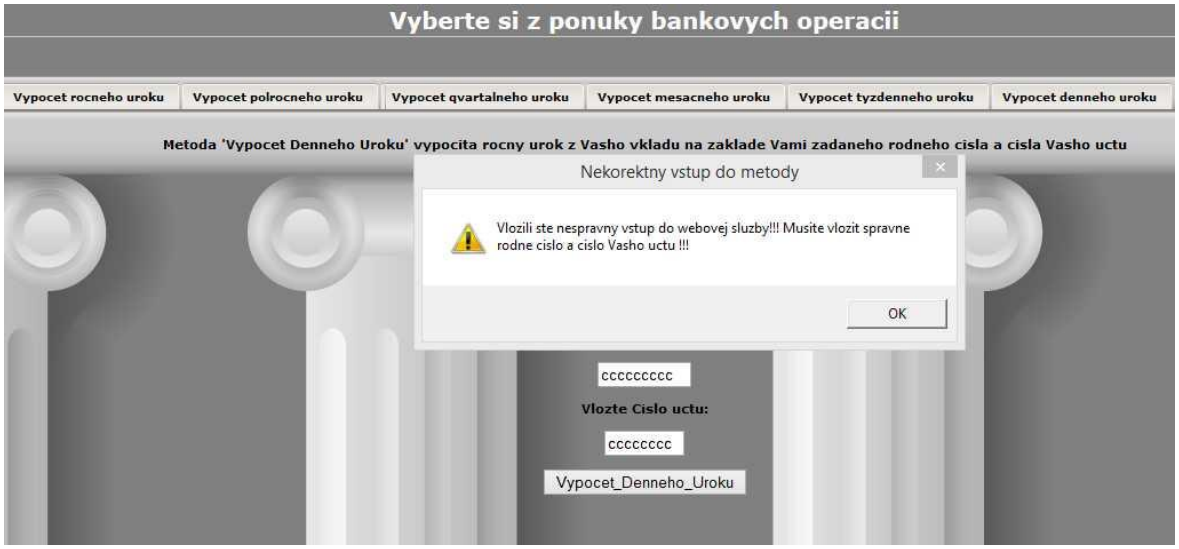
Na testovanie vstupných dát – rodného čísla a čísla účtu klienta, ktoré používateľ webovej služby zadáva ako svoje prihlasovacie meno a heslo, sa využívajú dve newebové metódy. Prvá metóda „InputValueTest“ vezme dáta, ktoré klient zadá do formulára v HTML klientovi a vyhodnotí, či sú dáta skutočne zadané, alebo chýba rodné číslo či číslo účtu. Premenná „InputValue“ predstavuje hodnotu zadaných dát a premenná „MethodName“ je webová metóda, ktorá túto newebovú metódu volá. V podmienke „if“ je zadané kritérium, podľa ktorého ak je „InputValue“ prázdna, systém vypíše používateľovi správu o prázdnom vstupe do webovej metódy. V opačnom prípade funkcia vráti hodnotu 1 a webová služba pokračuje v činnosti. Na obr. 8 je zobrazený prípad, keď je vstupný údaj prázdny a systém upozorňuje používateľa na zadanie správnych hodnôt.

```
public int InputValueTest(string InputValue, string MethodName)
    {
        if (InputValue == "")
        {
            string message = "Nevlozili ste ziaden vstup do metody '" + MethodName + "' webovej sluzby
                    !!!\nMusite vlozit spravne rodne cislo a cislo Vasho uctu !!!";
            string caption = "Prazdny vstup do metody '" + MethodName + "'";
            MessageBoxButtons buttons = MessageBoxButtons.OK;
            MessageBox.Show(message, caption, buttons, MessageBoxIcon.Exclamation,
MessageBoxDefaultButton.Button1, MessageBoxOptions.DefaultDesktopOnly);
            return 0;
        }
        else
        {
            return 1;
        }
    }
```



Obr. 8: Nekorektný vstup

Ďalej služba testuje zadané dáta klienta metódou „InputValueTest1“. Táto metóda načíta rodné číslo a číslo účtu, ktoré používateľ zadá, vyhľadá a načíta XML dokument do objektu „root“. Do objektov „nodeList“ a „nodeList2“ sú načítané hodnoty vkladu a mesačného úroku používateľa so zadaným rodným číslom a číslom účtu pomocou metódy „SelectNodes“. Ak bola hodnota jedného zo vstupov používateľa zadaná nesprávne, webová služba nenájde hodnotu vkladu takéhoto klienta a preto ostane objekt „nodeList“ alebo „nodeList2“ prázdny. Následne sú spočítané hodnoty objektov „nodeList“ a „nodeList2“.Ak je jeden objekt alebo druhý objekt prázdny, teda neobsahuje žiadne dáta o vklade alebo mesačnej úrokovej miere, používateľ dostane správu s informáciou o nesprávnom vstupe. Nesprávna môže byť hodnota rodného čísla, čísla účtu, alebo nesprávna kombinácia týchto dvoch vstupov. Situáciu, keď sú obidve hodnoty rodného čísla a čísla účtu nesprávne zobrazuje obr. 9.



Obr. 9: Nespravne rod.c. a cislo. u.

```
public int InputValueTest1(string Rodne_Cislo, string Cislo_Uctu)
      {
         XmlElement root = doc.DocumentElement;
         XmlNodeList nodeList;
         XmlNodeList nodeList2;
         nodeList = root.SelectNodes("/databaza/ucet[rod_c=" + Rodne_Cislo + " ][ cislo_u=" +
Cislo_Uctu + "]/vklad");
         nodeList2 = root.SelectNodes("/databaza/ucet[rod_c=" + Rodne_Cislo + " ][  cislo_u=" +
Cislo_Uctu + "]/mesac_urok");
         int nodeListCount = nodeList.Count;
         int nodeListCount2 = nodeList2.Count;
         if (nodeListCount == 0 || nodeListCount2 == 0)
         {
            string message = "Vlozili ste nespravny vstup do webovej sluzby!!! Musite vlozit spravne
rodne cislo a cislo Vasho uctu !!!";

            MessageBoxButtons buttons = MessageBoxButtons.OK;
```

```
            string caption = "Nekorektny vstup do metody";
            MessageBox.Show(message, caption, buttons, MessageBoxIcon.Exclamation,
                MessageBoxDefaultButton.Button1, MessageBoxOptions.DefaultDesktopOnly);
            return 0;
        }
        else
        {
            return 1;
        }
```

3.2.9 Vzorka dát z vytvorenej klientskej databázy v XML formáte

Táto časť XML dokumentu je použitá ako klientska databáza, s ktorou pracuje vytvorená webová služba. Prvý tag <?xml> definuje document ako XML document, atribút „version“ špecifikuje verziu XML jazyka a „encoding“ spôsob dekódovania. Prvý element „databaza“ je koreňovým elementom, obsahujúci všetky ostatné elementy. Jeho dcérsky element je „ucet“ s atribútom „banka“. „ucet“ obsahuje svoje dcérske elementy a ich hodnoty, menovite číslo účtu, meno a priezvisko vlastníka, jeho vklad, rodné číslo, ID číslo účtu a mesačný úrok priradený účtu. Príslušná metóda webovej služby vyhľadáva práve hodnoty elementov „vklad“ a „mesac_urok“ a to na základe elementov „cislo_u“ a „rod_c“, ktoré zadávajú používatelia služby.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<databaza>
<ucet banka="Business Bank">
  <cislo_u>24448391</cislo_u>
  <meno>Martin</meno>
  <priezvisko>Huzvar</priezvisko>
  <vklad>10000</vklad>
  <rod_c>9002048888</rod_c>
  <ucet_id>5749205471</ucet_id>
  <mesac_urok>0.0009</mesac_urok>
</ucet>
```

3.2.10 HTML klient komunikujúci s webovou metódou Vypocet_Rocneho_Uroku

Kód zobrazený nižšie vytvára HTML klienta, ktorý sa skladá z 3 častí. Tag <html> ohraničuje celý dokument, tag <head> ohraničuje hlavičku, v ktorej sa nachádza titulka „title“ pomenúvajúca titulku webovej aplikácie. Tiež obsahuje tag <meta>, ktorý popisuje obsah dokumentu html. V „meta“ tagu sa nachádza atribút „http-equiv“ s hodnotou „Content-Type“ definujúci obsah dokumentu ako html a text a definujúci znakovú sadu ako „iso-8859-1“. Označenie <link> definuje spojenie k externému zdroju, atribút „rel“

špecifikuje externý zdroj ako dokument s určitým štýlom, ktorý bude definovať dizajn html dokumentu, atribút „href“ obsahuje názov tohto dokumentu a atribút „type“ špecifikuje znaky dokumentu ako textové a css znaky.

```
<html>
<head>
<title>Bankove operacie</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<link rel="stylesheet" href="styles.css" type="text/css" />
</head>
```

Dokument pokračuje tagom <body> ktorý tvorí telo alebo obsah celého dokumentu. Tag <h1> ohraničuje nadpis webovej aplikácie a tag s atribútom „id“ a hodnotou „tabs“ vytvára odsek, v ktorom sa nachádza zoznam odkazov odkazujúcich na ďalšie html dokumenty. Dokument CSS, ktorý obsahuje informácie o dizajne webovej aplikácie rozpoznáva tento odsek podľa jeho „id“.

```
<body>
<h1>Vyberte si z ponuky bankovych operacii</h1>
<hr />
<div id="tabs">
  <ul>
    <li><a href="klient1.htm"><span>Vypocet rocneho uroku</span></a></li>
    <li><a href="klient2.htm"><span>Vypocet polrocneho uroku</span></a></li>
    <li><a href="klient3.htm"><span>Vypocet qvartalneho uroku</span></a></li>
    <li><a href="klient4.htm"><span>Vypocet mesacneho uroku</span></a></li>
    <li><a href="klient5.htm"><span>Vypocet tyzdenneho uroku</span></a></li>
    <li><a href="klient6.htm"><span>Vypocet denneho uroku</span></a></li>
  </ul>
</div>
    <br><br>
```

Ďalej je v dokumente definovaný formulár <form> s atribútom “method” a jeho hodnotou “post”. Tento atribút vezme údaje z formulára, ktoré sú klientom zapísané a pošle ich webovej metóde, ktorú špecifikuje ďalší atribút “action”. Tag <input> vytvára spolu s jeho atribútmi „type“, „size“ a „name“ textové políčko s názvom „Rodne_Cislo“ a veľkosťou 10 px. Ďalšie atribúty „type“ a „value“ vytvárajú tlačidlo s názvom „Vypocet_Rocneho_Uroku“, ktoré po stlačení odošlú zadané hodnoty webovej metóde špecifikovanej v atribúte „action“ v tagu <form>. Ďalší html klienti pracujúci s webovou službou sú uvedení v prílohe.

```
    <form method=POST
action='http://localhost:4641/Martin_Huzvar_WebService.asmx/Vypocet_Rocneho_Uroku'
">
```

```
      <br>
       <p>Metoda 'Vypocet Rocneho Uroku'  vypocita rocny urok z Vasho vkladu na
zaklade Vami zadaneho rodneho cisla a cisla Vasho uctu</p>
     <br><br><br><br><br><br><br><br>
       <p> Vlozte Rodne cislo:  </p>
      <input type="text" size="10" name='Rodne_Cislo'>
    <p>  Vlozte Cislo uctu:  </p>
      <input type="text" size="8" name='Cislo_Uctu'>
    <p>  <input type=submit value="Vypocet_Rocneho_Uroku"></p>
  </form>
  </body>
</html>
```

4 Záver a diskusia

Prvý čiastkový cieľ bol naplnený pri skúmaní a analyzovaní súčasného stavu ponúkaných webových služieb v bankovom sektore, ktoré pracujú s úrokovou správou bežného bankového účtu. Bolo zistené, že v súčasnosti na Slovensku ale aj v zahraničí takéto služby nie sú klientom poskytované, nanajvýš banky ponúkajú rôzne jednoduché aplikácie, ktoré sú schopné vypočítať mesačný alebo ročný úrok na základe zvolenej hodnoty úrokovej miery, výšky vkladu a úrokovej periódy. Neposkytujú však žiadne variácie týchto metód, nie sú integrované do bankového systému a nepracujú s klientskymi dátami. Týmto prieskumom sa naplnil čiastkový cieľ jeden.

Druhým cieľom bolo opísať a vysvetliť teoretické východiská týkajúce sa úročenia v bankách a teoretické východiská webových služieb. Tento cieľ bol splnený v kapitole teoretické východiská, ktorá pojednáva práve o základoch úročenia a procesoch pri úrokovaní využívaných v bankovom sektore tak ako aj o základných vzťahoch, typoch a funkciách webových služieb, technológiách a programovacích jazykoch, ktoré s nimi spolupracujú.

Na základe týchto teoretických poznakov bola vyvinutá webová služba, ktorá spája rôzne moderné programovacie jazyky, ako XHTML, CSS, C# a technológie ako ASP.NET platformu spolu s úrokovými procesmi a výpočtami bankového sektora. Táto webová služba poskytuje newebové metódy na overenie klientovej totožnosti a správnosti klientskych údajov, webové metódy na prístup ku klientskym dátam ako je vklad alebo úroková miera uloženým v XML súbore, počítajúce šesť rôznych typov úrokov, menovite ročný úrok, polročný úrok, kvartálny úrok, mesačný úrok, týždenný úrok a denný úrok. Posledná metóda slúži na výpis výsledku na obrazovku a zápis výsledku do súboru klienta so špecifickým názvom.

Aby sa táto aplikácia začala používať v bankovom sektore, potrebovala by ešte niekoľko ďalších úprav a metód, napríklad prepočet daní v závislosti od štátu alebo banky, zavedenie pohyblivej úrokovej miery, funkciu prístupu do skutočnej klientskej bankovej databázy a iné.

5 Použitá literatúra

[1] PIRČ, V. a kol. 2008. Finančná Matematika. Košice. Katedra matematiky FEI Technickej univerzity v Košiciach. ISBN: 978-80-8073-986-7

[2] BOOTH, D. a kol. 2004. Web Service Architecture. [cit. 2014-07-04] Dostupné na internete: < http://www.w3.org/TR/ws-arch/>

[3] LAŠÁK, M. 2010. Verzovanie webových služieb v SOA aplikáciách. Brno. 64 s. [cit. 2012-04-01] Dostupné na internete: <http://is.muni.cz/th/172840/fi_m/dp_lasak.pdf>

[4] CHINNICI, R. a kol. 2007. Web Service Description Language Version 2.0 part 1: Core language. [cit. 2013-11-11]. Dostupné na internete:<http://www.w3.org/TR/wsdl20/#intro_ws>

[5] ERL, T. a kol.2008. Web Service Contract Design and Versioning for SOA.s.l. : Prentice Hall. ISBN: 978-0-13-613517-3.

[6] CHRISTENSEN, E. a kol. 2008. Web Service Description Language. Dostupné na internete: <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>

[7] LIBERTY, J. , HURWITZ, D. 2003. Programming ASP.NET. Beijing. Cit.[2014-06-03] Dostupné na internete: <http://www.google.sk/books?hl=sk&lr=&id=7lf1cC6SXQ0C&oi=fnd&pg=PR9&dq=asp.net&ots=Ei42sIjkft&sig=euz3qhtEjL1WD7bXV15pRKZuqYc&redir_esc=y#v=onepage&q=asp.net&f=false>

[8] MUSCIANO, C. , KENNEDY B. 2007. Creating Effective Web Pages: HTML&XHTML. USA. Cit[2014-08-11]. Dostupné na internete : http://www.oreilly.de/catalog/html6/chapter/ch02.pdf

[9] FREEMANTLE, P. a kol. 2002. Enterprise Services. USA. Cit[2014-08-12]. Dostupné na internete : <http://condor.depaul.edu/dmumaugh/readings/handouts/SE435/cacm-p77-freemantle.pdf>

[10] *BOX, D. A kol. 2000. Simple Object Access Protocol. [cit. 2014-07-04]. Dostupné na internete: <http://www.immagic.com/eLibrary/ARCHIVES/SUPRSDED/W3C/W000520N.pdf >*

[11] ZAIKIN, M. 2009. Sun Certified Developer for Java Web Services 5.0. cit. [2014-21-09]. Dostupné na internete: < http://java.boot.by/scdjws5-guide/index.html>

6 Prílohy

6.1 HTML stránky jednotlivých úrokových metód

Nasledujúce HTML kódy boli použité ako súčasť web služby.

6.1.1 *HTML klient úvodnej web stránky*

```
<html>
<head>
<title>Bankove operacie</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<link rel="stylesheet" href="styles.css" type="text/css" />
</head>
<body>
<h1>Vyberte si z ponuky bankovych operacii</h1>
<hr />
<div id="tabs">
  <ul>
    <li><a href="klient1.htm"><span>Vypocet rocneho uroku</span></a></li>
    <li><a href="klient2.htm"><span>Vypocet polrocneho uroku</span></a></li>
    <li><a href="klient3.htm"><span>Vypocet qvartalneho uroku</span></a></li>
    <li><a href="klient4.htm"><span>Vypocet mesacneho uroku</span></a></li>
    <li><a href="klient5.htm"><span>Vypocet tyzdenneho uroku</span></a></li>
    <li><a href="klient6.htm"><span>Vypocet denneho uroku</span></a></li>

  </ul>
</div>
</body>
</html>
```

6.1.2 *HTML klient pre webovú metódu výpočtu ročného úroku*

```
<html>
<head>
<title>Bankove operacie</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<link rel="stylesheet" href="styles.css" type="text/css" />
</head>
<body>
<h1>Vyberte si z ponuky bankovych operacii</h1>
<hr />
<div id="tabs">
  <ul>
    <li><a href="klient1.htm"><span>Vypocet rocneho uroku</span></a></li>
    <li><a href="klient2.htm"><span>Vypocet polrocneho uroku</span></a></li>
    <li><a href="klient3.htm"><span>Vypocet qvartalneho uroku</span></a></li>
    <li><a href="klient4.htm"><span>Vypocet mesacneho uroku</span></a></li>
    <li><a href="klient5.htm"><span>Vypocet tyzdenneho uroku</span></a></li>
    <li><a href="klient6.htm"><span>Vypocet denneho uroku</span></a></li>

  </ul>
</div>
    <br><br>
```

```
    <form method=POST
action='http://localhost:4641/Martin_Huzvar_WebService.asmx/Vypocet_Rocneho_Uroku'
">
       <br>
        <p>Metoda 'Vypocet Rocneho Uroku'  vypocita rocny urok z Vasho vkladu na
zaklade Vami zadaneho rodneho cisla a cisla Vasho uctu</p>
      <br><br><br><br><br><br><br><br>
        <p> Vlozte Rodne cislo:  </p>
       <input type="text" size="10" name='Rodne_Cislo'>
     <p>  Vlozte Cislo uctu:  </p>
       <input type="text" size="8" name='Cislo_Uctu'>
     <p>  <input type=submit value="Vypocet_Rocneho_Uroku"></p>

   </form>

</body>
</html>
```

6.1.3 *HTML klient výstupnej stránky obsahujúcej výsledok výpočtu*

```
<html>
<head>
<title>Bankove operacie</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<link rel="stylesheet" href="styles.css" type="text/css" />
</head>
<body>
<h1>Vyberte si z ponuky bankovych operacii</h1>
<hr />
<div id="tabs">
  <ul>
    <li><a href="klient1.htm"><span>Vypocet rocneho uroku</span></a></li>
    <li><a href="klient2.htm"><span>Vypocet polrocneho uroku</span></a></li>
    <li><a href="klient3.htm"><span>Vypocet qvartalneho uroku</span></a></li>
    <li><a href="klient4.htm"><span>Vypocet mesacneho uroku</span></a></li>
    <li><a href="klient5.htm"><span>Vypocet tyzdenneho uroku</span></a></li>
    <li><a href="klient6.htm"><span>Vypocet denneho uroku</span></a></li>

  </ul>
</div>
    <form>
<textarea id="result" readonly rows="5" cols="70"> </textarea></form>

    <script type="text/javascript">

        if (typeof XMLHttpRequest === "undefined") {
            XMLHttpRequest = function () {
                try { return new ActiveXObject("Msxml2.XMLHTTP.6.0"); }
                catch (e) { }
                try { return new ActiveXObject("Msxml2.XMLHTTP.3.0"); }
                catch (e) { }
                try { return new ActiveXObject("Microsoft.XMLHTTP"); }
                catch (e) { }
                throw new Error("This browser does not support XMLHttpRequest.");
            };
        }

        function readBOX() {
            function reqListener() {

                document.getElementById("result").innerHTML = this.responseText;
```

```
            }


            var filePath = "http://localhost:4641/method_output.xml";

            var oReq = new XMLHttpRequest();
            oReq.onload = reqListener;
            oReq.open("get", filePath, true);
            oReq.send();
        }

        readBOX();

    </script>

</body>
</html>
```

6.1.4 *HTML klient pre webovú metódu výpočtu polročného úroku*

```
<html>
<head>
<title>Bankove operacie</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<link rel="stylesheet" href="styles.css" type="text/css" />
</head>
<body>
<h1>Vyberte si z ponuky bankovych operacii</h1>
<hr />
<div id="tabs">
  <ul>
    <li><a href="klient1.htm"><span>Vypocet rocneho uroku</span></a></li>
    <li><a href="klient2.htm"><span>Vypocet polrocneho uroku</span></a></li>
    <li><a href="klient3.htm"><span>Vypocet qvartalneho uroku</span></a></li>
    <li><a href="klient4.htm"><span>Vypocet mesacneho uroku</span></a></li>
    <li><a href="klient5.htm"><span>Vypocet tyzdenneho uroku</span></a></li>
    <li><a href="klient6.htm"><span>Vypocet denneho uroku</span></a></li>

  </ul>
</div>
    <br><br>

    <form method=POST
action='http://localhost:4641/Martin_Huzvar_WebService.asmx/Vypocet_Polrocneho_Uroku
' ">
       <br>
        <p>Metoda 'Vypocet Polrocneho Uroku'  vypocita rocny urok z Vasho vkladu na
zaklade Vami zadaneho rodneho cisla a cisla Vasho uctu</p>
      <br><br><br><br><br><br><br><br>
        <p> Vlozte Rodne cislo:  </p>
       <input type="text" size="10" name='Rodne_Cislo'>
     <p>  Vlozte Cislo uctu:  </p>
       <input type="text" size="8" name='Cislo_Uctu'>
     <p>  <input type=submit value="Vypocet_Polrocneho_Uroku"></p>

   </form>

</body>
</html>
```

6.1.5 *HTML klient pre webovú metódu výpočtu kvartálneho úroku*

```
<html>
<head>
<title>Bankove operacie</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<link rel="stylesheet" href="styles.css" type="text/css" />
</head>
<body>
<h1>Vyberte si z ponuky bankovych operacii</h1>
<hr />
<div id="tabs">
  <ul>
    <li><a href="klient1.htm"><span>Vypocet rocneho uroku</span></a></li>
    <li><a href="klient2.htm"><span>Vypocet polrocneho uroku</span></a></li>
    <li><a href="klient3.htm"><span>Vypocet qvartalneho uroku</span></a></li>
    <li><a href="klient4.htm"><span>Vypocet mesacneho uroku</span></a></li>
    <li><a href="klient5.htm"><span>Vypocet tyzdenneho uroku</span></a></li>
    <li><a href="klient6.htm"><span>Vypocet denneho uroku</span></a></li>

  </ul>
</div>
    <br><br>

    <form method=POST
action='http://localhost:4641/Martin_Huzvar_WebService.asmx/Vypocet_Qvartalneho_Urok
u' ">
       <br>
        <p>Metoda 'Vypocet Qvartalneho Uroku'  vypocita rocny urok z Vasho vkladu na
zaklade Vami zadaneho rodneho cisla a cisla Vasho uctu</p>
      <br><br><br><br><br><br><br><br>
        <p> Vlozte Rodne cislo:  </p>
       <input type="text" size="10" name='Rodne_Cislo'>
     <p>  Vlozte Cislo uctu:  </p>
       <input type="text" size="8" name='Cislo_Uctu'>
     <p>  <input type=submit value="Vypocet_Qvartalneho_Uroku"></p>

   </form>

</body>
</html>
```

6.1.6 *HTML klient pre webovú metódu výpočtu mesačného úroku*

```
<html>
<head>
<title>Bankove operacie</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<link rel="stylesheet" href="styles.css" type="text/css" />
</head>
<body>
<h1>Vyberte si z ponuky bankovych operacii</h1>
<hr />
<div id="tabs">
  <ul>
    <li><a href="klient1.htm"><span>Vypocet rocneho uroku</span></a></li>
    <li><a href="klient2.htm"><span>Vypocet polrocneho uroku</span></a></li>
    <li><a href="klient3.htm"><span>Vypocet qvartalneho uroku</span></a></li>
    <li><a href="klient4.htm"><span>Vypocet mesacneho uroku</span></a></li>
    <li><a href="klient5.htm"><span>Vypocet tyzdenneho uroku</span></a></li>
```

```
    <li><a href="klient6.htm"><span>Vypocet denneho uroku</span></a></li>

  </ul>
</div>
    <br><br>

    <form method=POST
action='http://localhost:4641/Martin_Huzvar_WebService.asmx/Vypocet_Mesacneho_Uroku'
">
       <br>
        <p>Metoda 'Vypocet Mesacneho Uroku'  vypocita rocny urok z Vasho vkladu na
zaklade Vami zadaneho rodneho cisla a cisla Vasho uctu</p>
      <br><br><br><br><br><br><br><br>
        <p> Vlozte Rodne cislo:  </p>
       <input type="text" size="10" name='Rodne_Cislo'>
     <p>  Vlozte Cislo uctu:  </p>
       <input type="text" size="8" name='Cislo_Uctu'>
     <p>  <input type=submit value="Vypocet_Mesacneho_Uroku"></p>

   </form>

</body>
</html>
```

6.1.7 *HTML klient pre webovú metódu výpočtu týždenného úroku*

```
<html>
<head>
<title>Bankove operacie</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<link rel="stylesheet" href="styles.css" type="text/css" />
</head>
<body>
<h1>Vyberte si z ponuky bankovych operacii</h1>
<hr />
<div id="tabs">
  <ul>
    <li><a href="klient1.htm"><span>Vypocet rocneho uroku</span></a></li>
    <li><a href="klient2.htm"><span>Vypocet polrocneho uroku</span></a></li>
    <li><a href="klient3.htm"><span>Vypocet qvartalneho uroku</span></a></li>
    <li><a href="klient4.htm"><span>Vypocet mesacneho uroku</span></a></li>
    <li><a href="klient5.htm"><span>Vypocet tyzdenneho uroku</span></a></li>
    <li><a href="klient6.htm"><span>Vypocet denneho uroku</span></a></li>

  </ul>
</div>
    <br><br>

    <form method=POST
action='http://localhost:4641/Martin_Huzvar_WebService.asmx/Vypocet_Tyzdenneho_Uroku
' ">
       <br>
        <p>Metoda 'Vypocet Tyzdenneho Uroku'  vypocita rocny urok z Vasho vkladu na
zaklade Vami zadaneho rodneho cisla a cisla Vasho uctu</p>
      <br><br><br><br><br><br><br><br>
        <p> Vlozte Rodne cislo:  </p>
       <input type="text" size="10" name='Rodne_Cislo'>
     <p>  Vlozte Cislo uctu:  </p>
       <input type="text" size="8" name='Cislo_Uctu'>
     <p>  <input type=submit value="Vypocet_Tyzdenneho_Uroku"></p>
```

```
   </form>

</body>
</html>
```

6.1.8 *HTML klient pre webovú metódu výpočtu denného úroku*

```
<html>
<head>
<title>Bankove operacie</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<link rel="stylesheet" href="styles.css" type="text/css" />
</head>
<body>
<h1>Vyberte si z ponuky bankovych operacii</h1>
<hr />
<div id="tabs">
  <ul>
    <li><a href="klient1.htm"><span>Vypocet rocneho uroku</span></a></li>
    <li><a href="klient2.htm"><span>Vypocet polrocneho uroku</span></a></li>
    <li><a href="klient3.htm"><span>Vypocet qvartalneho uroku</span></a></li>
    <li><a href="klient4.htm"><span>Vypocet mesacneho uroku</span></a></li>
    <li><a href="klient5.htm"><span>Vypocet tyzdenneho uroku</span></a></li>
    <li><a href="klient6.htm"><span>Vypocet denneho uroku</span></a></li>

  </ul>
</div>
    <br><br>

    <form method=POST
action='http://localhost:4641/Martin_Huzvar_WebService.asmx/Vypocet_Denneho_Uroku'
">
       <br>
        <p>Metoda 'Vypocet Denneho Uroku'  vypocita rocny urok z Vasho vkladu na
zaklade Vami zadaneho rodneho cisla a cisla Vasho uctu</p>
      <br><br><br><br><br><br><br><br>
        <p> Vlozte Rodne cislo:  </p>
       <input type="text" size="10" name='Rodne_Cislo'>
     <p>  Vlozte Cislo uctu:  </p>
       <input type="text" size="8" name='Cislo_Uctu'>
     <p>  <input type=submit value="Vypocet_Denneho_Uroku"></p>

   </form>

</body>
</html>
```

6.2 CSS dokument obsahujúci dizajn HTML klienta

```
body {
       font: bold 11px/1.5em Verdana;
       text-align: center;
       background-image: url("bank_icon.png");
       background-repeat: no-repeat;
    background-attachment: fixed;
    background-position: center;
    vertical-align: middle;
       }
```

```
h1 {
        font-family:Verdana, Arial, Helvetica, sans-serif;
        font-size:20px;
        font-weight:bold;
        margin:0;
        padding:0;
     color: white;
        }

hr {
        border:none;
        border-top:1px solid #CCCCCC;
        height:1px;
        margin-bottom:25px;
        }

#tabs {
        float:left;
        width:100%;
        font-size:93%;
        line-height:normal;
        border-bottom:1px solid #84776B;
        }

#tabs ul {
        margin:0;
        padding:15px 15px 0 70px;
        list-style:none;
     color: black;

        }

#tabs li {
        display:inline;
        margin:0;
        padding:0;

        }

#tabs a {
        float:left;
        background:url("tableft.gif") no-repeat left top;
        margin:0;
        padding:0 0 0 4px;
        text-decoration:none;

        }

#tabs a span {
        float:left;
        display:block;
        background:url("tabright.gif") no-repeat right top;
        padding:5px 15px 4px 6px;
        color:black;
        }
        textarea {
  position: absolute;
  left: 350px;
  top: 300px;
  right: 450px;
  bottom: 100px;
}
```

6.3 XML dokument obsahujúci klientskú databázu

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<databaza>

<ucet banka="Business Bank">
  <cislo_u>24448391</cislo_u>
  <meno>Martin</meno>
  <priezvisko>Huzvar</priezvisko>
  <vklad>10000</vklad>
  <rod_c>9002048888</rod_c>
  <ucet_id>5749205471</ucet_id>
  <mesac_urok>0.0009</mesac_urok>
</ucet>

<ucet banka="Private Bank">
  <cislo_u>14003978</cislo_u>
  <meno>Juraj</meno>
  <priezvisko>Huzvar</priezvisko>
  <vklad>50000</vklad>
  <rod_c>9002039999</rod_c>
  <ucet_id>6949205471</ucet_id>
  <mesac_urok>0.0009</mesac_urok>
</ucet>

<ucet banka="Public Bank">
  <cislo_u>59482745</cislo_u>
  <meno>Jozef</meno>
  <priezvisko>Zvolinsky</priezvisko>
  <vklad>100</vklad>
  <rod_c>9006041898</rod_c>
  <ucet_id>5741005471</ucet_id>
  <mesac_urok>0.0007</mesac_urok>
</ucet>

<ucet banka="Small Bank">
  <cislo_u>23452345</cislo_u>
  <meno>Peter</meno>
  <priezvisko>Vyskocil</priezvisko>
  <vklad>10000</vklad>
  <rod_c>9004048562</rod_c>
  <ucet_id>5749205491</ucet_id>
  <mesac_urok>0.0012</mesac_urok>
</ucet>

<ucet banka="Public Bank">
  <cislo_u>98087678</cislo_u>
  <meno>Jakub</meno>
  <priezvisko>Zeleznik</priezvisko>
  <vklad>200</vklad>
  <rod_c>9001038848</rod_c>
  <ucet_id>5775205471</ucet_id>
  <mesac_urok>0.0011</mesac_urok>
</ucet>

<ucet banka="Private Bank">
  <cislo_u>48205620</cislo_u>
  <meno>Ondrej</meno>
  <priezvisko>Latinak</priezvisko>
  <vklad>900000</vklad>
```

```
  <rod_c>9004044888</rod_c>
  <ucet_id>5749205431</ucet_id>
  <mesac_urok>0.0011</mesac_urok>
</ucet>

</databaza>
```