UNIVERSITATEA ALEXANDRU IOAN CUZA

FACULTATEA DE INFORMATICA

LUCRARE DE LICENTA

Descoperirea dinamică a serviciilor web

student

Bâgu Alexandru Bogdan

profesor coordonator

Pistol Ionuț

Declaraţie privind originalitate şi respectarea drepturilor de autor

Prin prezenta declar că Lucrarea de licenţă cu titlul "Descoperirea dinamica a serviciilor web " este scrisă de mine şi nu a mai fost prezentată niciodată la o altă facultate sau instituţie de învăţământ superior din ţară sau străinătate. De asemenea, declar că toate sursele utilizate, inclusiv cele preluate de pe Internet, sunt indicate în lucrare, cu respectarea regulilor de evitare a plagiatului:

- toate fragmentele de text reproduse exact, chiar şi în traducere proprie din altă limbă, sunt scrise între ghilimele şi deţin referinţa precisă a sursei;
- reformularea în cuvinte proprii a textelor scrise de către alţi autori deţine referinţa precisă;
- codul sursă, imagini etc. preluate din proiecte open-source sau alte surse sunt utilizate cu repsectarea drepturilor de autor şi deţin referinţe precise;
- rezumarea ideilor altor autori precizează referinţa precisă la textul original.

Data **Bâgu Alexandru Bogdan**

_____

Declaraţie de consimţământ

Prin prezenta declar că sunt de acord ca Lucrarea de licenţă cu titlul "Descoperirea dinamica a serviciilor web", codul sursă al programelor şi celelalte continuturi (grafice, multimedia, date de test etc.) care însoţesc această lucrare să fie utilizate în cadrul Facultăţii de Informatică.
De asemenea, sunt de acord ca Alexandru Ioan Cuza Iaşi să utilizeze, modifice, reproducă şi să distribuie în scopuri necomerciale programele-calculator, format executabil şi sursă, realizate de mine în cadrul prezentei lucrări de licenţă.

Bâgu Alexandru Bogdan

_____

Cuprins

1. Abstract
2. Introducere
3. Terminologie
4. Tehnologii folosite
    - 4.1 IP Multicasting
    - 4.2 ORM - Object Relational Mapping
    - 4.3 Grupul www-data și fișierul /etc/sudoers
    - 4.4 Executarea în background a comenzilor în PHP
    - 4.5 Domenii de nume în XML
5. Descoperirea dinamică a serviciilor web
    - 5.1 Descrierea protocolului
    - 5.2 Implementare
        - 5.2.1 Web Interface
        - 5.2.2 Client Service
        - 5.2.3 Server Service
6. Concluzii

2. Introducere

Un serviciu web este o colecție de protocoale și standarde folosite pentru schimbul de date între aplicații sau sisteme. Aplicațiile software scrise în limbaje de programare diferite și care rulează pe diverse platforme pot folosi serviciile web pentru a face schimb de informații într-o rețea locala sau pe Internet, într-o manieră oarecum asemănătoare comunicării între procesele de pe un singur calculator.

Majoritatea serviciile web folosesc următoarele tehnologii:

- XML-ul - pentru structurarea informațiilor
- SOAP/REST/HTTP - pentru transferul de date
- WSDL - pentru a descrie serviciile
- UDDI - pentru a lista ce servici sunt disponibile.

Serviciile web permit aplicațiilor din diferite surse să comunice unele cu altele fără a modifica codul în mod particular și deoarece comunicarea se realizează prin XML, serviciile web nu sunt dependente de sistemul de operare sau limbajul de programare. În prima faza, serviciile web, au fost folosite pentru doar pentru a comunica cu clienții dar în ultima vreme ideea de serviciu web a luat o amploare deosebită în rîndul site-urilor web, care oferă din ce în ce mai multe protocoale pentru trimiterea diverselor date către diferite tipuri și categorii de utilizatori, de la simple protocoale de RPC (destinate actualizării blog-urilor), RSS / Atom, motoare sociale, statistică și analiză, canale de feed-uri și multe alte servicii. Numărul acestor servicii fiind este în continuă creștere.

Pentru face posibilă comunicarea cu un serviciu web trebuie sa ii știm adresa prin care poate fi apelat și ce tip de serviciu este, de aceea este nevoie de un protocol pentru descoperirea serviciilor web. Fără un protocol bine stabilit este imposibilă detectarea serviciilor web de pe Internet. Protocolul prin care se face descoperirea serviciilor web se numește “WS Discovery”. Pentru acesta IANA (Internet Assigned Numbers Authority) asociază portul 3702 atît pe UDP cât și pe tcp (http://www.iana.org/assignments/port-numbers).

Protocolul a fost inițial dezvoltat de către BEA Systems, Canon, Intel, Microsoft și WebMethods care a fost aprobat ca un standard de OASIS începând cu 1 iulie 2009. Diverse componente ale sistemului de operare Microsoft Windows Vista utilizează WS-Discovery, de exemplu: "People near me". WS-Discovery este o parte integrantă a tehnologiilor Windows Rally.

3. Terminologie

- Serviciu țintă

  Un nod dintr-o rețea care poate fi descoperit
- Client

  Un nod dintr-o rețea care cauta servicii web
- Discovery Proxy

  Un nod nod dintr-o rețea care facilitează descoperirea serviciilor țintă de către client . Practic acestea au rolul de a face cache, stocînd astfel informațiile despre serviciile țintă
- Tip

  Tipul serviciului web
- Scop

  Scopul serviciului țintă după care se face căutarea
- Host

  Un PC care dispune de o adresa IPv4
- XML (Extensible Markup Language)

  Este un meta-limbaj de marcare pentru crearea de alte limbaje de marcare, cum ar fi ATOM, RDF, RSS și altele. Meta-limbajul XML este o simplificare a limbajului SGML (din care se trage și HTML) și a fost proiectat în scopul transferului de date între aplicații pe internet asigurînd o descriere pentru structura de date.XML este acum și un model de stocare a datelor nestructurate și semi-structurate în cadrul bazelor de date native XML.
- URI (Universal Resouce Indentifier)

  Reprezintă o abstractizare a unui URL
- URN (Uniform Resource Name)

  Este un URI ce folosește o schema urn iar resursa pe care o reprezinta nu este întotdeauna rezolvabilă
- UUID (Universally Unique IDentifier)

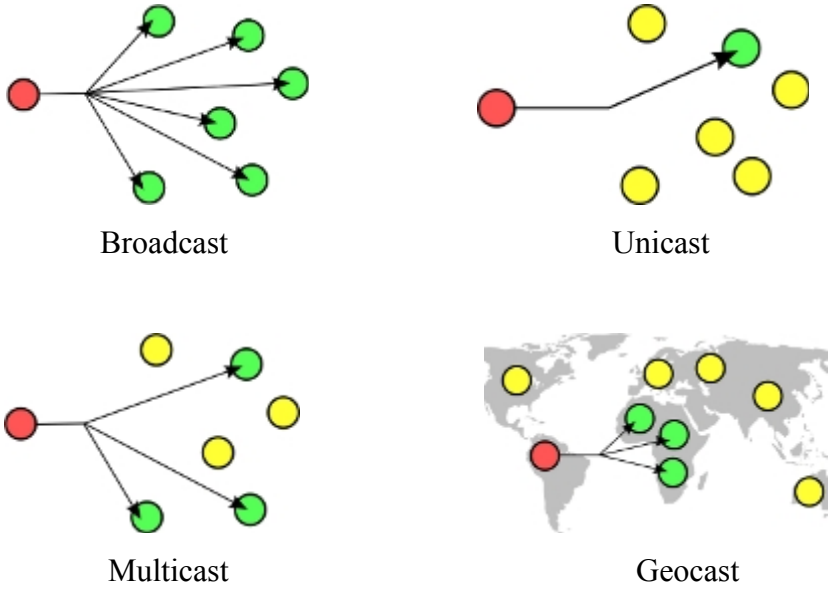
  Reprezinta un identificator unic folosit în software pentru diferite resurse. Scopul UUID-ului este de a permite sistemelor distribuite sa identifice în mod unic informații fără o coordonare specificata la nivel central.

4. Tehnologii folosite

4.1 IP Multicating

Sunt trei tipuri de adrese IPv4 : unicast, brodcast, multicast [geocast]. O adresă unicast este făcută pentru a transmite pachete către o destinație unică. Adresa de tip broadcast este folosită pentru a trimite pachete peste întreaga rețea iar cea multicast este utilizata pentru a transmite datagrame către un set de hosturi care s-au asociat în prealabil unui grup de tip multicasting. Diferența dintre adresa de tip multicast și cea unicast este prezenta unui grup de tip multicast în locul destinației.

Fig. 1: Tipuri de adrese IPv4



Nu sunt restricții atunci când un host dorește să se alăture unui grup de timp multicasting sau sa-l părăsească. De asemenea nu sunt restricții în ceea ce privește numărul hosturilor ce se asociază unui grup.

Spre deosebire de unicast și broadcast în care sunt folosite clasele de IP-uri A,B sau C, Multicastingul folosește clasa D de IP-uri, conform IANA (Internet Assigned Numbers Authority).

Clasa A	1.0.0.1 - 126.255.255.255
Clasa B	128.1.0.1 - 191.255.255.254
Clasa C	192.0.1.1 - 223.255.254.254
Clasa D	224.0.0.0 - 239.255.255.255

IP Multicastingul este folosit ca TV Multicasting încă din 2004 în USA înregistrîndu-se 1700 de stații care emiteau multicast.

4.2 ORM - Object Relational Mapping

După două decenii, industria software a ajuns la un punct stabil în evoluția tehnologiilor orientate-obiect. Privind la generaţia actuală și viitoare a tehnologiilor, a devenit evident faptul că următoarea provocare în industria software este de a reduce complexitatea de accesare și de integrare a informațiilor care nu au nativ definită tehnologia OO. Cele două mai utilizate resurse non-OO de stocare informații sunt bazele de date relaționale și XML. ORM-ul are ca scop salvarea tabelelor din bazele de date în clase adăugînd relații de legătura între acestea. Astfel putem manipula obiectele instantiate fără a fi nevoie sa mai scriem comenzi SQL.

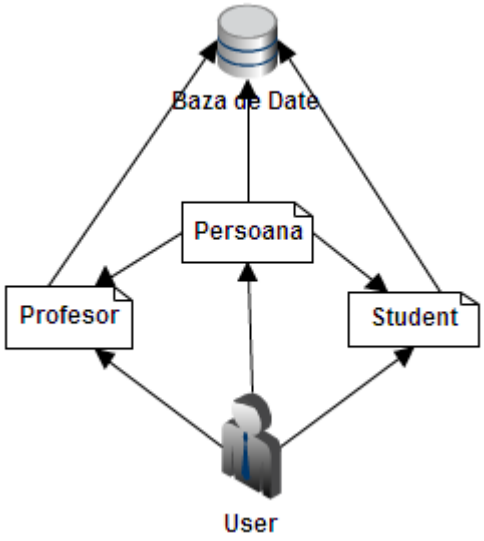


Figura 2: Legatura intre obiecte si baza de date

În schema de mai sus Profesor, Persoana și Student sunt clase ce pointează către tabele în baza de date. Între aceste obiecte se pot fi relații de legătura dacă au fost definite și constrîngeri între tabele.

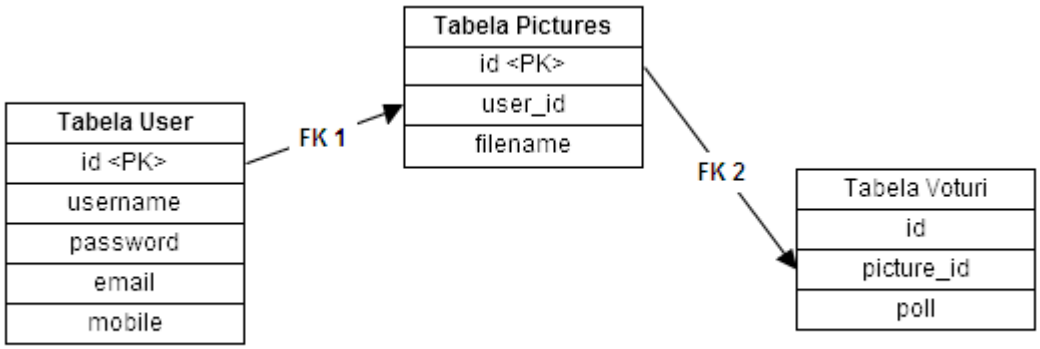


Fig. 3: Tabele si relatiile de legatura dintre ele

În imaginea de mai sus avem trei tabele și relațiile de legătura între ele. PK reprezentînd cheia primară iar FK constrîngerea intre doua tabele. După generarea claselor, care poată fi făcuta automat sau manual, putem merge în tabela Pictures din tabela User folosind relația FK1.

4.3 Grupul/userul www-data și fișierul /etc/sudoers

www-data reprezintă grupul/userul în care pornește serverul de Apache. Acest grup este limitat din punctul de vedere al comenzilor pe care le poate executa. Din aceasta cauză putem crea un grup/user nou iar apoi vom modifica fișierul /etc/apache2/envvars specificînd userul și grupul:

```
export APACHE_RUN_USER=alexandrubagu
export APACHE_RUN_GROUP=alexandrubagu
```

Pentru executarea unor comenzi speciale (terminarea unui proces, executarea unui proces în background) fără a fi nevoiți sa introducem parola pentru root putem modifica fișierul /etc/sudoers. Acest fișier reprezintă o lista de reguli peste utilizatorii sistemului. Fișierul este descris printr-o gramatică de tip EBNF (Extended Backus-Naur Form) și este alcătuit din doua

tipuri de înregistrări: aliasuri pentru comenzi și specificațiile pentru utilizatori, care descriu ce comenzi pot fi executate. Aliasurile sunt de patru tipuri: User_Alias , Runas_Alias, Host_Alias și Cmnd_Aliast.

Exemple pentru specificarea aliasurilor:

```
# Aliasuri pentru useri
User_Alias  FULLTIMERS = john, mike
User_Alias  PARTTIMERS = fox, david



# Aliasuri pentru utilizatori
Runas_Alias OP = root, operator
Runas_Alias DB = oracle

# Aliasuri pentru hosturi
Host_Alias  NETS = 128.138.0.0/255.255.0.0
Host_Alias  SERVERS =  mail, www

# Aliasuri pentru comenzi
Cmnd_Alias  KILL = /usr/bin/kill
Cmnd_Alias  PRINTING = /usr/sbin/lpc, /usr/bin/lprm
Cmnd_Alias  SHUTDOWN = /usr/sbin/shutdown
Cmnd_Alias  REBOOT = /usr/sbin/reboot
```

După specificarea aliasurilor de mai sus, putem să asociem pentru fiecare user unul sau mai multe aliasuri ca în exemplele de mai jos:

```
#Userii din grupul FULLTIMERS pot executa orice comandă fără să le
#fie cerută parola
FULLTIMERS  ALL = NOPASSWD: ALL

#Utilizatorul operator poate executa comenzile definite de aliasurile
#de mai jos doar dacă s-a autentificat
operator    ALL = KILL, SHUTDOWN, REBOOT, PRINTING

#Pentru orice mașina dintre hosturile NET, utilizatorul marian
#poate executa orice comandă din directorul /usr/bin
marian            NETS = /usr/bin/
```

4.4 Executarea în background a comenzilor în PHP

Pentru a executa comenzi în PHP putem folosi următoarele funcții :

```
string shell_exec ( string $cmd )
string system ( string $command [, int &$return_var ] )
string exec ( string $command [, array &$output [, int &$return_var ]] )
void pcntl_exec ( string $path [, array $args [, array $envs ]] )
resource popen ( string $command , string $mode )
```

Primele patru funcții nu fac altceva decît să execute în aceeași proces comanda specificată ca argument. O atenție deosebită ii vom acorda ultimii funcții. Funcția popen realizează următoarele operații:

- creează un pipe în cadrul procesului curent
- instantiaza un proces copil;
- închide capetele pipe-ului conform celui de-al doilea parametru - "r" sau "w" , ce definește operația ce va fi posibilă asupra pipe-ului din cadrul procesului părinte, scriere sau citire;
- redirectează intrarea sau ieșirea standard după caz, spre capătul pipe-ului rămas deschis din cadrul procesului copil
- asociază un stream de tip FILE descriptorului pipe-ului din cadrul procesului părinte pe care îl și returnează

În urma apelului popen() se creează astfel un pipe prin care se pot transmite date, sau primi date, din procesul curent către unul extern, prin intermediul descriptorului de tip FILE. Apelul pclose() pur și simplu închide capătul pipe-ului din procesul curent accesibil prin pointerul amintit, ce este transmis ca parametru, în plus așteptînd și finalizarea procesului copil (extern).

4.5 Domenii de nume în XML

Domeniile de nume în XML furnizează o metoda de reutilizare a marcajelor în documente, eliminânad conflictele între nume elementelor folosite în orice limbaje de marcare. Prin conflict înțelegem un element sau o combinație dintre un element și un atribut care a mai fost definit ulterior.

Vom considera următorul XML ca exemplu:

```
<?xml version="1.0" ?>
<facultate>
      <nume>Facultatea de Informatica</nume>
      <profesor>
            <nume>Popescu</nume>
            <prenume>Vasile</prenume>
            <materii>
                  <materie>M1</materie>
                  <materie>M2</materie>
            </materii>
      </profesor>
      ...
</facultate>
```

Putem observa, în exemplul de mai sus, ca elementul nume este folosit cu doua semnificații diferite : prima dată este folosit în denumirea facultății iar a doua oara este folosit pentru a specifica numele profesorului. Pentru a rezolva astfel de probleme au fost introduse domeniile de nume (namespaces). Domeniile de nume identifică care elemente din document aparțin aceluiași vocabular XML.

Pentru a face distincție între elemente diferite le vom încadra pe fiecare într-un spațiu de nume. Fiecărui domeniu ii vom asocia un URI unic.

Un URI (Universal Resouce Indentifier) reprezintă o abstractizare a unui URL, în timp ce URL identifică o resursă. De exemplu, putem să specificăm un URI pentru un profesor specificînd acestuia numele și prenumele dar în schimb nu putem vizualiza profesorul într-un browser spcificând numele și prenumele.

Definirea domeniilor de nume se realizează folosind atributul xmlns. Încadrarea unui element într-un domeniu de nume se face printr-un prefix urmat de doua puncte. Asocierea dintre prefix și domeniu de nume se face folosind atributul xmlns în rădăcina elementului ce introduce conceptul. Pentru a înțelege mai bine cum se declara aceste domenii de nume vom folosi exemplul anterior, căruia îi vom adăuga namespace-uri.

```
<?xml version="1.0" ?>
  <facultate xmlns:f=”http://www.example.ro/xml/facultate”>
        <f:nume>Facultatea de Informatica</f:nume>
        <profesor xmlns:p=”http://www.example.ro/xml/profesor”>
              <p:nume>Popescu</p:nume>
              <p:prenume>Vasile</p:prenume>
              <p:materii>
                    <p:materie>M1</p:materie>
                    <p:materie>M2</p:materie>
              </p:materii>
        </p:profesor>
        ...
  </facultate>
```

5. Descoperirea dinamică a serviciilor web

5.1 Descrierea protocolului

În prima parte vom descrie un protocol pentru descoperirea dinamică a serviciilor web. Principalul mod de descoperire a serviciilor web este căutarea de către client a serviciilor ținta, pentru aceasta vom defini următoarele tipuri de mesaje, specificînd și rolul acestora:

Probe reprezintă un mesaj trimis de Client atunci când dorește căutarea unor servicii ținta după un anumit tip sau scop.

```
<s:Envelope
      xmlns:a="http://schemas.xmlsoap.org/ws/2004/08/addressing"
      xmlns:d="http://schemas.xmlsoap.org/ws/2005/04/discovery"
      xmlns:s="http://www.w3.org/2003/05/soap-envelope" >
<s:Header>
      <!-- Acțiunea ce urmează a avea loc -->
      <a:Action>
            http://schemas.xmlsoap.org/ws/2005/04/discovery/Probe
      </a:Action>
      <!-- Identificator mesaj ( UUID ) -->
      <a:MessageID>
            uuid:0a6dc791-2be6-4991-9af1-454778a1917a
      </a:MessageID>
      <!-- Cui îi este adresat mesajul -->
      <a:To>urn:schemas-xmlsoap-org:ws:2005:04:discovery</a:To>
</s:Header>
<s:Body>
      <!-- Mesaj de tipul Probe -->
      <d:Probe>
            <!-- Tipul probei -->
            <d:Types>PrintBasic</d:Types>
      </d:Probe>
</s:Body>
</s:Envelope>
```

Probe Match este mesajul trimis de server dacă tipul serviciului ținta corespunde cu tipul pe care îl caută clientul. În caz contrar nu se trimite nici un mesaj de acest tip.

```
<s:Envelope
      xmlns:a="http://schemas.xmlsoap.org/ws/2004/08/addressing"
      xmlns:d="http://schemas.xmlsoap.org/ws/2005/04/discovery"
      xmlns:s="http://www.w3.org/2003/05/soap-envelope" >
      <s:Header>
      <!-- Acțiunea ce urmează a avea loc -->
      <a:Action>
            http://schemas.xmlsoap.org/ws/2005/04/discovery/ProbeMatches
      </a:Action>
      <!-- Identificator mesaj ( UUID ) -->
      <a:MessageID>
            uuid:e32e6863-ea5e-4ee4-997e-69539d1ff2cc
      </a:MessageID>
      <!-- Identificatorul mesajului pentru care urmează sa se răspundă -->
      <a:RelatesTo>
            uuid:0a6dc791-2be6-4991-9af1-454778a1917a
      </a:RelatesTo>
      <!-- Cui îi este adresat mesajul -->
      <a:To>
            http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
      </a:To>
</s:Header>
<s:Body>
      <!-- Mesaj de tipul ProbeMatch -->
      <d:ProbeMatch>
            <!-- referința de tip EndPointReference -->
            <a:EndpointReference>
                  <!-- UUID-ul serviciului web -->
                  <a:Address>
                        uuid:98190dc2-0890-4ef8-ac9a-5940995e6119
                  </a:Address>
            </a:EndpointReference>
            <!-- Tipul serverului -->
            <d:Types>PrintBasic/d:Types>
      </d:ProbeMatch>
</s:Body>
</s:Envelope>
```

Hallo reprezintă mesajul trimis de un serviciu ținta atunci când se alătura rețelei de noduri. Acestă conține informații cheie despre serviciul ținta.

```
<s:Envelope
      xmlns:a="http://schemas.xmlsoap.org/ws/2004/08/addressing"
      xmlns:d="http://schemas.xmlsoap.org/ws/2005/04/discovery"
      xmlns:s="http://www.w3.org/2003/05/soap-envelope" >
<s:Header>
      <!-- Acțiunea ce urmează a avea loc -->
      <a:Action>
            http://schemas.xmlsoap.org/ws/2005/04/discovery/Hello
      </a:Action>
      <!-- Identificator mesaj ( UUID ) -->
      <a:MessageID>
            uuid:73948edc-3204-4455-bae2-7c7d0ff6c37c
      </a:MessageID>
      <!-- Cui îi este adresat mesajul -->
      <a:To>urn:schemas-xmlsoap-org:ws:2005:04:discovery</a:To>
</s:Header>
<s:Body>
      <!-- Mesaj de tipul Hallo -->
      <d:Hello>
            <!-- referința de tip EndPointReference -->
            <a:EndpointReference>
                  <!-- UUID-ul serviciului web -->
                  <a:Address>
                        uuid:98190dc2-0890-4ef8-ac9a-5940995e6119
                  </a:Address>
            </a:EndpointReference>
      </d:Hello>
</s:Body>
</s:Envelope>
```

Bye este un mesaj trimis de un serviciu ținta atunci când părăsește rețeaua. Acesta conține aceleași informații ca și mesajul Hallo.

```
<s:Envelope
      xmlns:a="http://schemas.xmlsoap.org/ws/2004/08/addressing"
      xmlns:d="http://schemas.xmlsoap.org/ws/2005/04/discovery"
      xmlns:s="http://www.w3.org/2003/05/soap-envelope" >
```

```
<s:Header>
      <!-- Acțiunea ce urmează a avea loc -->
      <a:Action>
            http://schemas.xmlsoap.org/ws/2005/04/discovery/Bye
      </a:Action>
      <!-- Identificator mesaj ( UUID ) -->
      <a:MessageID>
            uuid:337497fa-3b10-43a5-95c2-186461d72c9e
      </a:MessageID>
      <!-- Cui îi este adresat mesajul -->
      <a:To>urn:schemas-xmlsoap-org:ws:2005:04:discovery</a:To>
</s:Header>
<s:Body>
      <!-- Mesaj de tipul Bye -->
      <d:Bye>
            <!-- referința de tip EndPointReference -->
            <a:EndpointReference>
                  <!-- UUID-ul serviciului web -->
                  <a:Address>
                        uuid:98190dc2-0890-4ef8-ac9a-5940995e6119
                  </a:Address>
            </a:EndpointReference>
      </d:Bye>
</s:Body>
</s:Envelope>
```

Resolve este mesajul trimis de Client atunci când acesta dorește să descopere numele și adresa serviciului web

```
<s:Envelope
      xmlns:s="http://www.w3.org/2003/05/soap-envelope"
      xmlns:a="http://schemas.xmlsoap.org/ws/2004/08/addressing"
      xmlns:d="http://schemas.xmlsoap.org/ws/2005/04/discovery">
<s:Header>
      <!-- Acțiunea ce urmează a avea loc -->
      <a:Action>
            http://schemas.xmlsoap.org/ws/2005/04/discovery/Resolve
      </a:Action>
      <!-- Identificator mesaj ( UUID ) -->
      <a:MessageID>
```

```
            urn:326B9C36-9996-11E0-8EB4-72983CD85658
      </a:MessageID>
      <!-- Cui îi este adresat mesajul -->
      <a:To>
            http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
      </a:To>
</s:Header>
<s:Body>
      <!-- Mesaj de tipul Resolve -->
      <d:Resolve>
            <!-- Tipul pentru serviciului -->
            <d:Types>PrinterBasic</d:Types>
            <!-- adresa de transport folosită pentru comunicarea cu un -->
            <!-- serviciu ținta sau proxy -->
            <d:XAddrs>192.168.1.100:3752</d:XAddrs>
      </d:Resolve>
</s:Body>
</s:Envelope>
```

Resolve Match după recepționarea acestui mesaj clientul va ști ce adresa de transport va avea serviciu țintă.

```
<s:Envelope
      xmlns:s="http://www.w3.org/2003/05/soap-envelope"
      xmlns:a="http://schemas.xmlsoap.org/ws/2004/08/addressing"
      xmlns:d="http://schemas.xmlsoap.org/ws/2005/04/discovery">
<s:Header>
      <!-- Acțiunea ce urmează a avea loc -->
      <a:Action>
            http://schemas.xmlsoap.org/ws/2005/04/discovery/ResolveMatch
      </a:Action>
      <!-- Identificator mesaj ( UUID ) -->
      <a:MessageID>
            urn:C414F658-99C0-11E0-8E07-0BA03CD85658
      </a:MessageID>
      <!-- Cui îi este adresat mesajul -->
      <a:To>
            http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
      </a:To>
</s:Header>
```

```
<s:Body>
      <!-- Mesaj de tipul Resolve -->
      <d:Resolve>
            <d:Types>Printer Basic</d:Types>
      <a:ReplyTo>
            <!-- referința de tip EndPointReference -->
            <a:EndpointReference>
                  <a:Address>
                        84381DB8-80E1-3B58-B6EA-BA3713D202CA
                  </a:Address>
            </a:EndpointReference>
      </a:ReplyTo>
      <!-- adresa de transport folosită pentru comunicarea cu un -->
      <!-- serviciu ținta sau proxy -->
      <d:XAddrs>http://ww.webservice.ro/transportaddress/apiCall</d:XAddrs>
      </d:Resolve>
</s:Body>
</s:Envelope>
```

După cum putem vedea în mesajele de mai sus, apar anumite cîmpuri, de aceea vom explica rolul fiecăruia:

- **<a:Action>...</a:Action>**

  Reprezintă acțiunea ce urmează a avea loc

- **<a:MessageID>...</a:MessageID>**

  Este un identificator unic (UUID) pentru fiecare mesaj

- **<a:To>...</a:To>**

  Cîmp ce reprezintă cui i se adresează mesajul. Dacă este trimis către un serviciu ținta atunci se folosește `urn:schemas-xmlsoap-org:ws:2005:04:discovery`

- **<a:RelatesTo>...</a:RelatesTo>**

  Acest cîmp reprezintă ID-ul mesajului pentru care se răspunde

- **<d:XAddrs>...</d:XAddrs>**

  Reprezintă adresa de transport folosită pentru comunicarea cu un serviciu ținta sau proxy

- **<a:EndpointReference><a:Address>...</a:Address></a:EndpointReference>**

Serviciile ținta prezente în rețea au un UUID unic, aceasta informație este inclusă într-un element de tipul EndpointReference [WS Adressing]. Cîmpul `<a:Address>` trebuie sa reprezinte o adresa logică sau de identificare și nu neapărat una rezolvabilă, deci putem defini un UUID în aceasta adresa. În cazul în care această adresa nu este una rezolvabilă se va defini un cîmp `<d:XAddrs>`.

Modul în care funcționează protocolul pentru descoperirea serviciilor web este următorul:

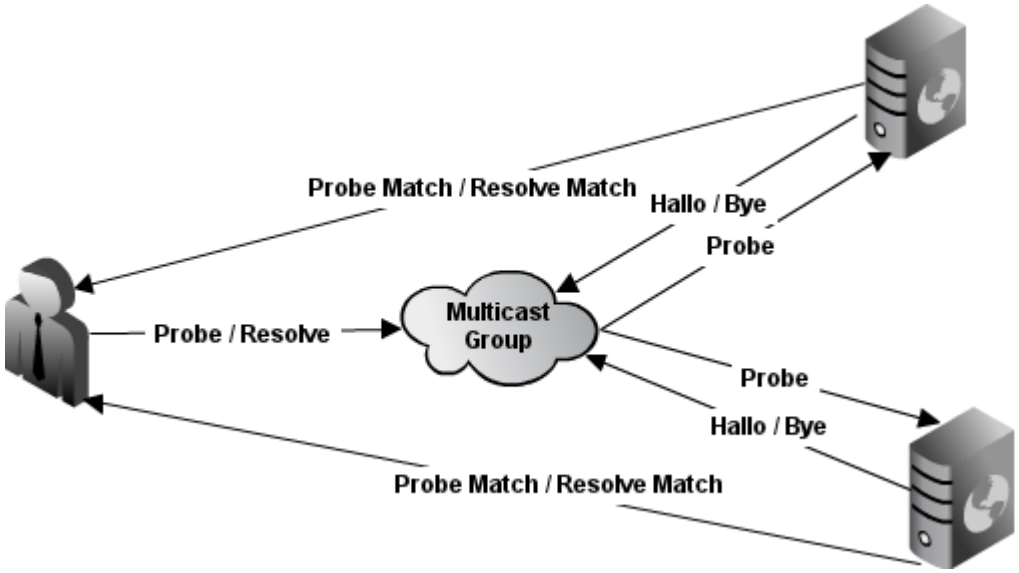


Fig. 4: Modul de functionare al protocolului (fara proxy)

1. Clientul trimite o probă, ce reprezintă mesajul de tip **Probe,** către un grupul de tip multicast.
2. Serverul primește această probă, deoarece s-a alăturat în prealabil grupului, iar dacă tipul serviciului ținta corespunde cu tipul pe care îl căuta clientul atunci serverul trimite unicast, un mesaj de tipul **Probe Match**
3. Clientul primește acest mesaj și creează un mesaj de tip **Resolve** după care îl trimite grupului folosind multicastingul.
4. După ce mesajul ajunge la server, acesta este verificat pentru a vedea dacă i se adresează serverului în căzut. În caz afirmativ serverul îi răspunde clientului unicast cu un mesaj de tip **Resolve Match** care va conține adresa serviciului web.
5. Atunci când un serviciu ținta se alătură rețelei acesta trimite un mesaj de tip **Hallo**, iar cazul în care va părăsi rețeaua va trimite un mesaj de tip **Bye**.

Pentru că acest protocol să fie scalabil, în cazul în care avem foarte multe noduri cu servicii țința, putem defini și un nod cu numele de Proxy. Dacă Proxy-ul detecteaza un mesaj de tip Probe sau Resolve ii va trimite un răspuns clientului prin care se va identifica. Astfel clientul poate alege între a comunica numai cu Proxy-ul sau să urmeze pașii protocolului descris mai sus. Rolul acestui Proxy este de a face caching (de a salva informații).

În prima etapa el se comportă ca un Client după care salvează datele despre fiecare servicii țința în parte.

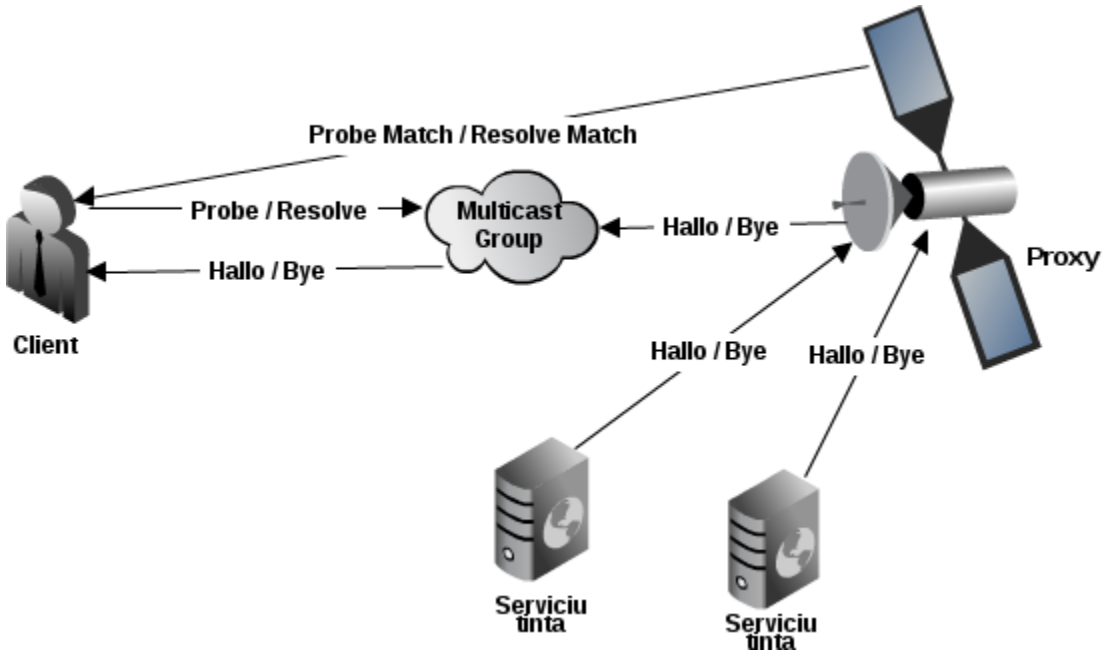


Fig.4: Modul de functionare al protocolului (cu proxy)

5.2 Implementare

Pentru implementarea protocolului vom folosi trei module care vor funcționa independent:

1. *Web Interface* - interfața web ce va fi conectată cu modulul Client Service
2. *Client Service* - modul ce va fi pornit din interfața web și se va participa activ la descoperirea serviciilor web
3. *Server Service* - modulul care creează servicii țintă ce urmează a fi descoperite.

Aceste module vor fi descrise detaliat în cele ce urmează.

5.2.1 Web Interface

Acest modul reprezintă interfața pentru utilizator care este compusă dintr-o serie de butoane și ferestre realizate cu ajutorul librăriei de JavaScript jQuery UI. Ferestrele prezente în interfața se pot închide, după necesitățile clientului, putînd fi redeschise apăsînd butoanele din partea de sus.

Modulul conține și doua butoane cu funcții speciale: “Start ClientService” și “Stop ClientService”. Prin apăsarea primului buton, clientul va apela un script PHP care va executa comanda de pornire a modului ClientService printr-o comandă bash folosind funcția popen, descrisa în subcapitolul “Executarea în background a comenzilor în PHP”. Acesta cerere se realizează folosind AJAX. După apăsarea acestui buton putem accesa lista de procese, folosind comanda *ps -G alexandrubagu | grep perl* și putem observa că modulul Client Service rulează în background.

Pentru a începe căutarea tuturor serviciilor țintă va trebui sa apăsam pe butonul “Search All” iar pentru căutarea după un anumit tip vom apăsa pe butonul “Custom Search” după ce am selectat o opțiune din dropdown. Prin apăsarea unuia dintre butoane se apelează un script PERL care creează mesajul de tip Probe și-l va scrie în fișierul messagesFromWebInterface.dat. Modul de Client Service va copia în fișierul messagesToSend.dat mesajele create de utilizatori. Aceste mesaje vor fi trimise de către modulul Client Service către grupul de tip multicast.

Modulul Web Interface mai conține și o fereastră pentru vizualizarea logurilor create de Client Service. Fiecărui thread din modulul Client Service îi este asociat un fișier de loguri iar imediat după ce serviciul a fost deschis se citesc toate fișierele de loguri, se sortează descrescător după dată, și se afișează ultimile 30 de loguri. Acest lucru este realizat printr-o cerere de tip AJAX către fișierul _geLogs.php din directorul Ajax.

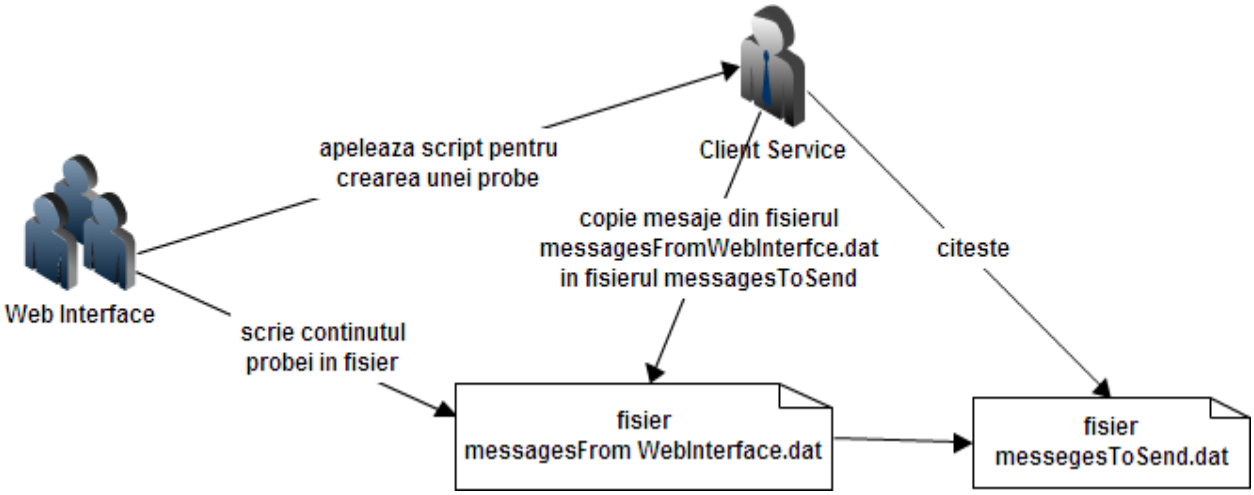


Fig.5: Modul de funcționare al componentelor modulului Web Interface

5.2.2 Client Service

Acest modul este alcătuit din trei componente. Aceste componente sunt pornite imediat după pornirea modulului, fiecărei componente îi vom asocia un thread, astfel avem :

1. threadMulticastIn - primește mesaje de tipul Hallo sau Bye
2. threadMulticastOut - trimite mesaje de tipul Probe sau Resolve
3. threadUnicastIn - primește unicast mesaje de tipul Probe Match sau Resolve Match

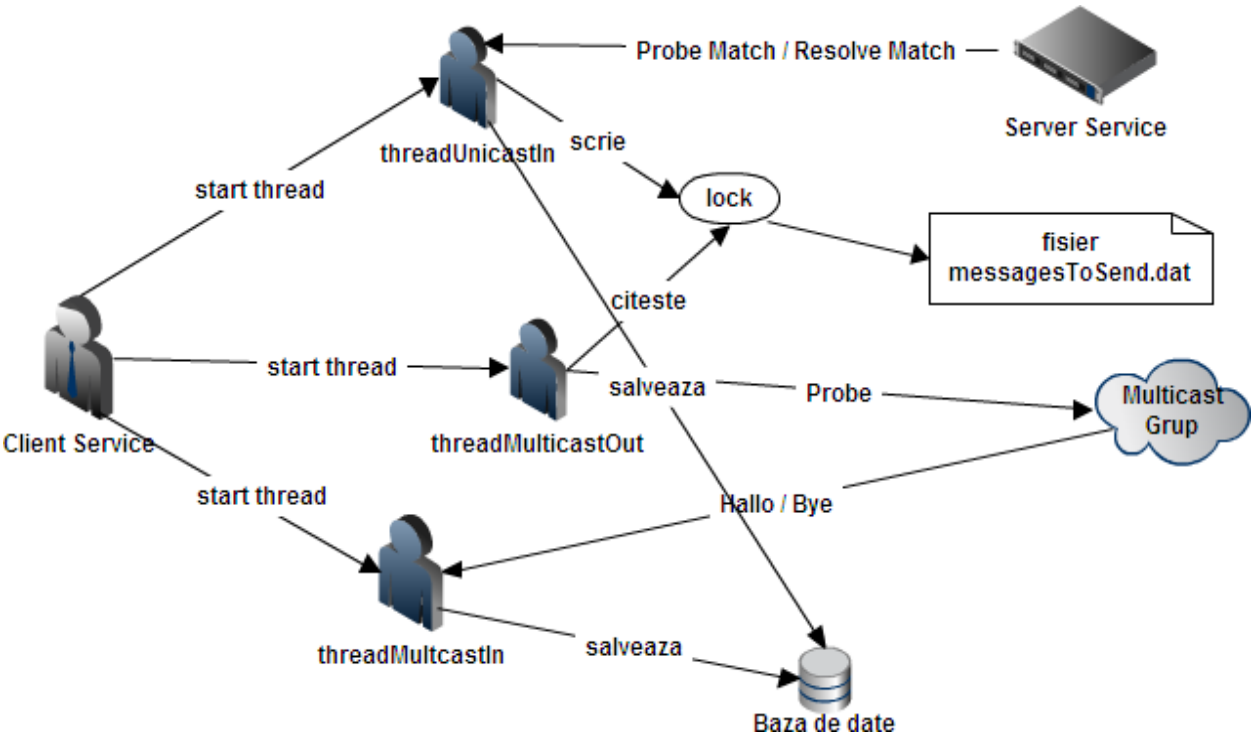


Fig. 6 : Modul de funcționare al componentelor modului ClientService

Prima componenta este asociată cu grupul de tip multicast astfel va primi mesaje de tipul Hallo sau Bye de la serviciile țintă. După ce este mesajul este primit, se extrage din mesaj UUID-ul serviciului țintă și cu ajutorul ORM-ului se modifică statusul, online sau offline, în baza de date. Statusurile serviciile web vor putea fi vizualizate în modulul Web Interface în fereastra de servicii descoperite.

A doua componenentă, threadMulticastOut, are ca scop trimiterea către grupul de tip multicast a mesajelor din fișierul messagesToSend.dat. Înainte de a trimite aceste mesaje, sunt copiate în fișierul messagesToSend.dat mesajele care provin de la utilizatorii modulului Web Interface.

Cea de a treia componenta primește unicast mesaje de tipul Probe Match sau Resolve Match. În funcție de tipul mesajul thread-ul realizează următoarele operații:

- dacă mesajul primit este de tip Probe Match atunci se va crea un mesaj de tipul Resolve. După ce mesajul este creat se verifică dacă exista lock pe fișier. În cazul în care fișierul este blocat se așteaptă pâna ce fișierul nu mai este utilizat. În cazul în care fișierul messagesToSend.dat nu este utilizat, se pune lock pe fișier și se depune mesajul creat. Acest lock este creat cu scopul de a permite doar una din cele două operații, scriere și citire. Astfel nu se poate citi și scrie simultan.
- dacă mesajul este de tipul Resolve atunci se extrag informațiile necesare din mesaj, adresa de transport și UUID-ul serviciului țintă, și cu ajutorul ORM-ului se introduc în baza de date.

5.2.3 Server Service

Acest modul simulează o fermă de n servicii țintă pe care le poate descoperi clientul. Fiecare serviciu țintă are un tip, un UUID și o adresă de transport, fiecare fiind unică. Fiecare serviciu țintă reprezintă un thread care este alcătuit din alte doua thread-uri: send și recv. Deci în total vom avea 3*n thread-uri.

Prin thread-ul send, serviciul țintă trimite către grupul de tip multicast mesaje de tipul Hallo sau Bye. Aceste mesaje sunt trimise pe rând la intervale specificate la pornirea serviciului țintă.

Thread-ul recv se va comporta în felul următor:

- După primirea unui mesaj de tip Probe îi va trimite unicast clientului un mesaj de tip Probe Match dacă tipul pe care l-a specificat clientul în proba coincide cu tipul serviciului țintă
- Dacă mesajul este de tip Resolve atunci serviciul țintă îi va trimite unicast clientului un mesaj de tipul Resolve Match în care îi va specifica adresa de transport și UUID-ul

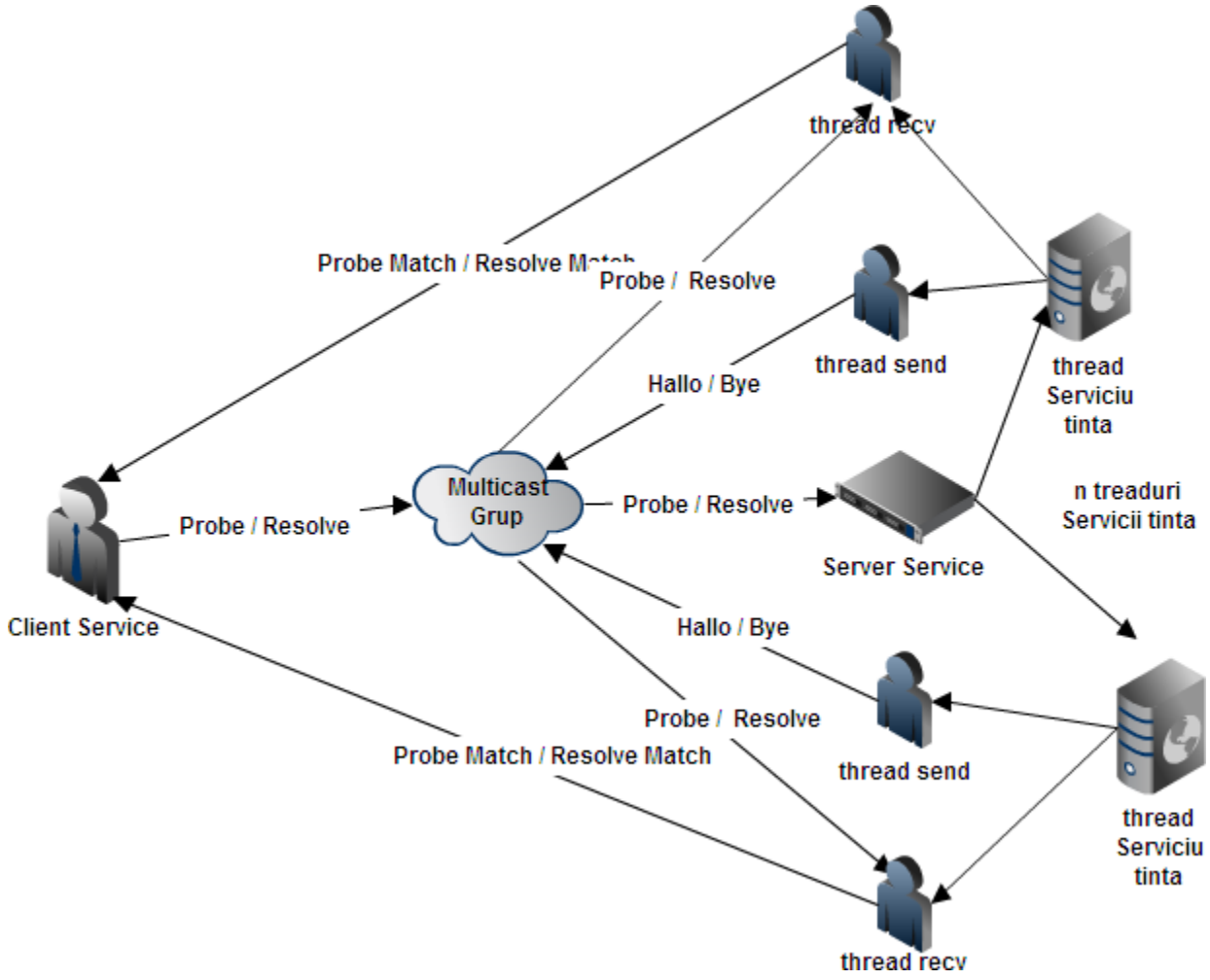


Fig. 7: Modul de functionare al modulului ServerService

6. Concluzii

Avînd în vedere că ideea de serviciu web a luat o amploare în rîndul site-urilor web, nu putem avea o evidenta asupra lor. De aceea e necesar un protocol pentru descoperirea acestora. Protocolul prezentat reprezintă o modalitate de descoperire dinamică atît a serviciilor aflate într-o rețea cât și pe Internet, indiferent de limbajul de programare sau sistemul de operare folosit, el fiind implementat în JAVA, .NET framework versiunea 4.

Acest protocol nu excelează din motive de securitate pentru că mesajele sunt supuse la o varietate mare de atacuri. Prin urmare, comunicarea ar trebui sa implementeze următoarele mecanisme: WS-Security , WS-SecureConversation și WS-Trust .

Protocolul este vulnerabil la următoarele tipuri de atacuri:

- **Alterarea mesajelor** - Conținutul mesajului poate fi schimbat de către un atacator. Pentru a preveni acest lucru, mesajul ar trebui să fie criptat sau sa se facă schimb de chei în prealabil.
- **Disponibilitatea (Denial of Service)** - Un atacator poate trimite mesaje care consumă resurse. Pentru a preveni acest lucru, o semnătură asigură că un mesaj este original. Pentru a evita prelucrarea inutilă, semnarea ar trebui să fie valida înainte de efectuarea oricărei prelucrări.
- **Retrimitere de mesaje (Replay)** - Un atacator poate retrimite un mesaj valabil și poate cauza prelucrări duplicate.
- **Spoofing** - Un atacator trimite un mesaj care pretinde ca mesajul este autentic. Pentru a preveni acest lucru, semnarea ar trebui să fie unica pentru fiecare client.

Bibliografie

[PHP] Zend PHP Study Guide [consultat - 1/04/2008]

http://www.zend.com/en/services/certification/

[Perl] Perl Threads Shared [consultat - 1/11/2010]

http://perldoc.perl.org/threads/shared.html

[JavaScript] Javascript jQuery UI Framework [consultat - 1/04/2009]

http://jqueryui.com/

[JavaScript] Javascript Jquery Framework [consultat - 1/05/2010]

http://jquery.com

[Linux] Man /etc/sudoers [consultat - 21/05/2011]

http://www.gratisoft.us/sudo/sudoers.man.html

[Linux] Bash Alias [consultat - 21/05/2011]

http://ss64.com/bash/alias.html

[PHP] Popen function [consultat - 21/05/2011]

http://php.net/manual/en/function.popen.php

http://profs.info.uaic.ro/~eonica/rc/

[ORM] Object Relation Object [consultat - 21/05/2011]

http://en.wikipedia.org/wiki/Object-relational_mapping

[ORM Patterns] 10 orm patterns: Components of a object-relational mapper [consultat - 21/05/2011]

http://giorgiosironi.blogspot.com/2009/08/10-orm-patterns-components-of-object.html

[WS-Discovery] Web Service Discovery Protocol [consultat - 14/02/2011]

http://specs.xmlsoap.org/ws/2005/04/discovery/ws-discovery.pdf

[IANA] Port Numbers," February 2005 [consultat - 14/02/2011]

http://www.iana.org/assignments/port-numbers

[Namespaces in XML 1.1] Namespaces in XML [consultat - 17/04/2011]

http://www.w3.org/TR/2004/REC-xml-names11-20040204

[SOAP 1.1] Simple Object Access Protocol (SOAP) [consultat - 17/04/2011]

http://www.w3.org/TR/2000/NOTE-SOAP-20000508

[SOAP1.2] SOAP Version 1.2 Part 1: Messaging Framework [consultat - 17/04/2011]

http://www.w3.org/TR/2003/REC-soap12-part1-20030624

[UUID] A UUID URN Namespace [consultat - 17/04/2011]

http://www.ietf.org/internet-drafts/draft-mealling-uuid-urn-05.txt

[WS-Addressing] Web Services Addressing (WS-Addressing) [consultat - 19/04/2011]

http://www.w3.org/Submission/2004/SUBM-ws-addressing-20040810

[WS-Trust] Web Services Trust Language (WS-Trust) [consultat - 28/05/2011]

http://schemas.xmlsoap.org/ws/2005/02/trust

[WSDL 1.1] Web Services Description Language (WSDL) [consultat - 18/03/2011]

http://www.w3.org/TR/2001/NOTE-wsdl-20010315

[WS-Security] Web Services Security: SOAP Message Security [consultat - 18/03/2011]

http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf