UNIVERSITATEA POLITEHNICA DIN BUCUREȘTI
FACULTATEA DE ELECTRONICĂ, TELECOMUNICAȚII ȘI TEHNOLOGIA INFORMAȚIEI

TEMĂ DE CASĂ

Aplicaţii RIA

Profesor coordonator:
Conf. dr. ing. Ștefan Stăncescu

Studentă:
Angelica Negrilă
IISC – anul II

București
2016

Cuprins

1. Introducere/Caracteristici generale

Termenul “Rich Internet Application (RIA – Aplicatii pe internet imbogatite)” a fost utilizat prima dată în martie 2002 într-o carte redactată de organizaţia Macromedia (în prezent parte componentă a Adobe). RIA se bazează pe o serie de concepte precum: Remote Scripting, utilizat de Microsoft din 1998; X Internet, utilizat de Forrester Research din 2000; Rich (Web) clients; Rich Web application.

O formă tipică de aplicaţie de tip RIA este o aplicaţie client care are un nivel separat de servicii pe componenta back-end. Termenii de front-end şi back-end sunt des utilizaţi şi se referă la faza iniţială şi respectiv la faza finală a unui proces. Componenta front-end este responsabilă de colectarea de la utilizator a intrărilor în diferite forme (fiind un fel de interfaţă între utilizator şi componenta back-end) şi de preluarea acestora în conformitate cu o specificaţie pe care o utilizează componenta back-end.

Aplicaţiile de tip RIA se află în faza de început în ceea ce priveste dezvoltarea lor şi adaptarea de către utilizatori. Pentru implementarea cu succes a unei aplicaţii de tip RIA trebuie rezolvate restricţiile şi cerinţele următoare:

- Adaptarea navigatoarelor: majoritatea aplicaţiilor de tip RIA necesită navigatoare Web moderne pentru a fi executate
- Standardele Web: existenţa unor diferenţe între navigatoarelor Web poate face dificilă crearea de aplicaţii de tip RIA operaţionale pe marea majoritate a acestor navigatoare
- Instrumentele de dezvoltare: anumite medii de lucru Ajax, produse precum Curl, Adobe Flex şi Microsoft Silverlight şi platforme de executie precum WebORB furnizează un mediu integrat în care se pot crea aplicaţii de tip RIA
- Accesibilitatea: poate fi restricţionată din cauza unor abordări tehnice care ar putea limita accesul la aplicaţii
- Acceptarea limitată de către utilizatori: anumite funcţionalităţi ale navigatorului în cazul aplicaţiilor Web standard pot fi diferite în cazul aplicaţiilor de tip RIA.

O aplicaţie de tip RIA se execută prin intermediul unui navigator Web şi nu necesită instalarea niciunui software special sau local într-un mediu protejat (“sandbox”) de către un mecanism de Securitate care asigură execuţia în siguranţă a programelor, acest mecanism este utilizat pentru rularea de cod netestat sau considerat nesigur provenit de la furnizori necunoscuţi.

Aplicaţiile de tip RIA includ o serie de caracteristici, cele mai importante dintre acestea fiind prezentate în continuare:

- Sensibilitate şi interactivitate;
- Interfaţă îmbunătăţită cu utilizatorul;
- Utilizare pe scară largă;
- Comunicare în timp real;
- Existenţa unor interfeţe cu utilizatorul din ce în ce mai generoase asigură creşterea gradului de reuşită al proceselor de business;
- Existenţa unor interfeţe mai interactive şi astfel mai intersante pentru utilizatori
- Creşterea satisfacţiei clienţilor şi a productivităţii utilizatorilor

- Utilizarea de limbaje de programare şi de modele de proiectare standard
- Reducerea costurilor de dezvoltare a aplicaţiilor prin diminuarea ponderii proceselor complexe şi utilizarea unei dezvoltări bazate pe autoinstruire
- Sprijinirea utilizatorilor pentru a se adapta cu uşurinţă la schimbările de conţinut sau tehnologice
- Posibilitatea realizării de aplicaţii proprii prin abordarea "look abd feel" utilizând modele de "skinning" (măsti de prezentare) şi stil bazate pe CSS.

2. Tehnologii/protocoale folosite in dezvoltarea aplicatiilor pe Internet îmbogăţite (RIA)

Tehnologiile RIA oferă, în plus faţă de aplicaţiile tradiţionale Web caracterizate prin mod de implementare, capacitate de administrare şi disponibilitate, o serie de facilităţi noi precum sunt utilizabile, capacitatea de răspuns şi reutilizarea aplicaţiilor client/server. Aplicaţiile de tip RIA oferă flexibilitatea în utilizare a unei aplicaţii inteligente de tip desktop şi adaugă capabilităţi largi ale aplicaţiilor Web tradiţionale la un nou tip de experienţă Web care este angajantă, interactivă, de tip "lightweight" şi flexibilă prin tehnologiile suport prezentate în continuare.

2.1. Protocolul TCP/IP

Comunicaţia între două terminale implică o conexiune Internet realizată prin rutere IP. Acest serviciu de conectare de terminale este de tip serviciu fără conexiune şi utilizează protocoalele TCP/IP.

Nivelul fizic are rolul de a transporta datele prin LAN, MAN sau WAN. Nivelul fizic utilizează o gamă largă de protocoale pentru conectarea la reţea: Ethernet, linii închiriate, ATM, etc. Adaptarea IP la această mare varietate de reţele este unul dintre motivele dezvoltării deosebite a Internetului. Nivelul reţea include protocolul IP care face posibilă comunicaţia între reţele. Acesta include funcţii de rutare şi adresare. IP furnizează un serviciu de transfer de unităţi de date – datagrame – între calculatoarele gazdă şi rutere sau între rutere. Sursa şi destinaţia sunt identificate prin adrese IP de lungime fixă. Protocolul IP este de asemenea responsabil cu segmentarea şi recombinarea datagramelor. La acest nivel nu se realizează controlul transmisiei corecte, dar se realizează o verificare a antetului pentru a detecta datagramele transmise eronat, care se pierd.

Protocolul IP trebuie să fie implementat în fiecare calculator gazdă care are acces la Internet precum şi în fiecare gateway sau ruter care interconectează diferite reţele.

TCP este un protocol de comunicaţie între procese şi este orientat pe conexiune logică între procesele reţelei.

Funcţiile de bază asigurate de TCP sunt următoarele:

– transferul datelor de bază sub formă de şir continuu de biţi ai datelor în ambele direcţii, obţinut prin împachetarea unui număr de octeţi în segmente care pot fi furnizate nivelelor inferioare

– adresarea şi multiplexarea – TCP poate fi folosit simultan de mai multe procese, care utilizează numerele de porturi (care identifică utilizatorul) şi adrese IP diferite

– TCP atribuie numere de secvenţe la fiecare datagramă, astfel că TCP poate tolera

pierderea sau duplicarea datagramelor; numărul de secvenţă permite corecţia erorilor şi împachetarea datagramelor; receptorul va transmite către emiţător confirmări ale recepţiilor corecte, lipsa confirmării indicând emiţătorului că trebuie realizată o retransmisie

– controlul fluxului este realizat prin confirmările recepţionate, care sunt însoţite de numărul de octeţi pe care transmiţătorul le poate transmite înaintea recepţiei următoarei confirmări

– nivelul transport realizează conexiunea între procesele care comunică, iar aceasta rămâne realizată până la terminarea comunicaţiei

– procesele care utilizează TCP pot specifica prioritatea şi securitatea asociate conexiunii

TCP poate fi folosit pentru numeroase protocoale de aplicaţii, ca SMTP (Simple Mail Transfer Protocol), FTP (File Transfer Protocol), HTTP (HyperText Transfer Protocol).

Pentru setarea unei conexiuni TCP una din părţi transmite un pachet în care flagul SYN este setat. Dacă partea cealaltă acceptă conexiunea, el transmite un pachet care are flagurile ACK şi SYN setate. Partea chemătoare confirmă (ACK) recepţia pachetului. Procedura de conexiune TCP este finalizată şi poate începe comunicaţia în ambele sensuri. La terminarea comunicaţiei, cele două părţi se informează reciproc prin cerere de deconectare.

2.2. Protocolul HTTP

Protocolul de Transfer Hipertext (HyperText Transfer Protocol – HTTP) constituie modalitatea de accesare a paginilor web. Acest protocol este cel care permite unui client să realizeze o conexiune la un server web.

A) http/1.1

Versiunea 1.1 a specificaţiei HTTP a fost publicată de către IETF în iunie 1999. Îndeplinind cerinţele de performanţă şi scalabilitate, aceasta a rămas specificaţia finală care reglementează modul de implementare protocolului HTTP.

Practic, o aplicaţie client (ex. browserul web) adresează o cerere aplicaţiei serverului (ex. serverul web). Cererea de acces permite identificarea resursei solicitate. Cererile şi răspunsurile conţin un antet cu informaţii despre resursă şi un corp care conţine resursa sau documentul solicitat.

Sunt folosite curent următoarele cereri:

– GET – clientul cere o resursă specificată

– OPTIONS – asocierea de cerinţe sau de posibilităţi suplimentare

– HEAD – similară cu GET, dar nu conţine un corp al mesajului (ex. nu se cere un document, ci numai data ultimei actualizări)

– POST – informaţii suplimentare referitoare la resursa specificată (ex. date dintr-un formular)

– PUT – o cale de scriere de pagini web şi punere a lor pe un server la distanţă

– DELETE – indică serverului îndepărtarea resursei specificate

– TRACE – permite clientului să urmărească ruta (calea) prin care a fost transmis mesajul

Răspunsul la cereri conţine coduri de stare de 3 octeţi:

– 1xx: informaţie – cerere recepţionată, procesarea în curs de realizare
– 2xx: succes – cerere recepţionată, înţeleasă şi acceptată
– 3xx: rerutare – cererea necesită rutare pentru obţinerea de date suplimentare
– 4xx: eroare client – cererea conţine erori
– 5xx: eroare server – serverul nu poate răspunde la cererea validate

Protocolul HTTP este astfel folosit cu succes pentru multiple aplicaţii, dintre care se remarcă în mod deosebit accesarea paginilor web. Totuşi, acest protocol are un dezavantaj deosebit, acela că transmisia datelor de la server la client şi invers se face în clar. Astfel, informaţiile pot fi extrase de terţe persoane. În cazul unor informaţii confidenţiale, cum ar fi datele cardului de credit, o astfel de scurgere de informaţii poate produce pierderi deosebite.

Pentru a rezolva această problemă există o extensie a protocolului – Secure HTTP.

B) Secure HTTP (S-HTTP/HTTPS)

HTTP este ideal pentru comunicaţii deschise, dar nu oferă facilităţi de autentificare şi criptare. S-HTTP a fost dezvoltat pentru a lucra împreună cu HTTP pentru a permite clienţilor şi serverelor să se angajeze în tranzacţii private şi sigure. S-HTTP este în mod deosebit folositor pentru criptarea informaţiilor din formulare atunci când sunt transmise de la client la server.

S-HTTP este folosit de o serie de produse. Suportă o varietate de algoritmi de criptare şi moduri de operare. Mesajele pot fi protejate folosind semnături digitale, autentificare şi criptare. După primul contact, emiţătorul şi receptorul stabilesc preferinţele pentru criptarea şi manipularea mesajelor sigure.

Un număr de algoritmi şi tehnici de securitate pot fi folosiţi, incluzând criptarea DES şi RC2, sau semnătura cu chei publice RSA. În plus, utilizatorii pot alege să folosească un anumit tip de certificat sau nici un certificat. În cazuri în care certificate conţinând cheia publică nu sunt disponibile, este posibil ca emiţătorul şi receptorul să folosească o cheie de sesiune pe care au schimbat-o anterior.

Totuşi, S-HTTP nu a fost niciodată acceptat în întregime de producătorii de browsere cum ar fi Microsoft şi Netscape. În schimb, un protocol similar numit SSL (Secure Sockets Layer) a devenit mai popular.

C) SSL (Secure Sockets Layer)

Protocolul SSL rulează deasupra TCP/IP şi sub protocoalele de nivel mai înalt cum ar fi HTTP sau IMAP. El foloseşte TCP/IP în numele protocoalelor de nivel mai înalt, iar în timpul procesului permite unui server SSL-enabled să se autentifice pentru un client SSL-enabled, permite clientului să se autentifice faţă de server şi permite ambelor sisteme să stabilească o conexiune criptată.

Aceste posibilităţi se vin în întâmpinarea unor griji fundamentale privind comunicaţia în Internet şi alte reţele TCP/IP:

– **autentificarea SSL a serverului** permite unui utilizator să confirme identitatea serverului; clientul SSL poate folosi tehnica standard a criptografiei cu cheie publică pentru a verifica dacă certificatul este valid şi a fost emis de o autoritate de certificare recunoscută de client
– **autentificarea SSL a clientului** permite serverului să confirme identitatea unui

utilizator; folosind aceeaşi tehnică, software-ul SSL-enabled al serverului poate verifica certificatul clientului

– **o conexiune SSL criptată** necesită ca toate informaţiile trimise între un client şi un server să fie criptate de software-ul care le trimite şi decriptate de cel care le primeşte, oferind astfel un grad mare de confidenţialitate; în plus, toate datele transmise prin conexiuni SSL criptate sunt protejate cu un mecanism de detectare a modificărilor suferite de pachete pe parcurs.

Protocolul SSL include două sub-protocoale: protocolul SSL înregistrare (*SSL record protocol*) şi protocolul SSL transfer (*SSL handshake protocol*). Protocolul SSL înregistrare defineşte formatul folosit pentru transmisia de date. Protocolul SSL transfer implică folosirea protocolului SSL înregistrare pentru a schimba o serie de mesaje între client şi server atunci când se stabileşte iniţial conexiunea SSL. Acest schimb de mesaje este destinat să faciliteze următoarele acţiuni:

– autentificarea serverul faţă de client
– selectarea de către client şi server a algoritmilor criptografici pe care le ambii le suportă
– opţional, autentificarea clientului faţă de server
– folosirea unor tehnici de criptare cu cheie publică pentru a genera secrete partajate
– stabilirea unei conexiuni SSL criptate

D) Serverul Web Apache

Protocolul HTTP face, aşadar, legătura între un client şi un server . Client poate fi orice calculator care are un browser web (ex. Internet Explorer, Netscape Navigator / Communicator, Opera, etc.). La fel, server poate fi orice calculator cu un software special destinat. Deşi există mai multe opţiuni pentru un astfel de soft, se poate spune că Apache s-a impus ca un standard de facto. Conform unui *studiu statistic efectuat pe Internet**[3]***, în luna mai 2004, 33.892.817 de servere rulau Apache, adică un procent de 67,05 din totalul serverelor web.

În februarie 1995, cel mai popular software pentru server web era *HTTP daemon*, program gratuit, dezvoltat de Rob McCool la NCSA (National Center for Supercomputing Applications) din Universitatea din Illinois. Dezvoltarea *httpd* a încetase totuşi de la mijlocul anului 1994, când Rob McCool a părăsit NCSA. Astfel că mulţi webmasteri şi-au dezvoltat propriile extensii şi corecţii de erori care erau necesare. Un mic grup de astfel de webmasteri, contactaţi prin intermediul adreselor de email, s-au strâns pentru a coordona schimbările aduse de ei – sub forma de *„petece”* (*„patches”*). S-a pornit astfel un mailing list, un loc pentru informaţia partajată şi conturi pentru dezvoltatorii principali. Până la sfârşitul lunii februarie, opt dintre aceştia au format Grupul Apache iniţial.

Folosind httpd 1.3 al NCSA ca bază, ei au adăugat toate corecţiile de erori şi îmbunătăţirile semnificative care au putut fi găsite, au testat rezultatele pe propriile servere şi au lansat prima versiune publică (0.6.2) a serverului Apache în aprilie 1995.

Serverul Apache de la început a reprezentat un mare succes, dar dezvoltatorii săi erau conştienţi că era necesar un redesign complet al codului de bază. În timp ce majoritatea dezvoltatorilor se ocupau cu implementarea de noi facilităţi pentru versiunile 0.7.x şi cu sprijinul acordat comunităţii utilizatorilor Apache, aflată în plină ascensiune, Robert Thau a realizat o nouă arhitectură pentru server. Aceasta avea o structură modulară şi interfaţă pentru

aplicaţii (API) pentru o mai bună extensibilitate, precum caracteristici care îmbunătăţeau performanţele sale. Grupul a trecut la această nouă arhitectură în iulie şi a adăugat facilităţilor de la 0.7.x, rezultând Apache 0.8.8 în August.

După teste extensive, portarea pe platforme din cele mai diverse, o nouă documentaţie şi adăugarea multor facilităţi sub forma modulelor standard, Apache 1.0 a fost lansat pe 1 decembrie 1995.

La mai puţin de un an de la formarea grupului, Apache a devenit serverul web numărul unu pe Internet, poziţie pe care şi-o menţine şi în prezent.

Puterea Apache stă tocmai în numărul mare de utilizatori. Fiind un software gratuit, distribuit şi în forma de cod sursă ce poate fi modificat de utilizatori, aceştia pot deveni astfel şi dezvoltatori. De asemenea, tot datorită gratuităţii sale, utilizatorii sunt mult mai dispuşi să anunţe dezvoltatorilor eventualele erori sau lipsuri de securitate.

Pe lângă aceasta, structura modulară şi API-ul său permit adăugarea de noi facilităţi de către terţe părţi sau folosirea sa în comun cu alte aplicaţii. Un exemplu în acest sens îl poate constitui OpenSSL, un proiect ce urmăreşte printre altele implementarea protocolului SSL prezentat anterior. Similar cu serverul Apache, acesta este un proiect open-source, dezvoltat de o comunitate mondială de voluntari care folosesc Internetul pentru a comunica, planifica şi dezvolta aplicaţia propriu-zisă şi documentaţia aferentă.

2.3. Adobe Flash

O altă tehnologie foarte utilizată este Adobe Flash, care nu se confruntă cu probleme de compatibilitate a navigatoarelor, însă pentru dezvoltatori este dificil de realizat integrarea cu .NET.

Adobe Flash este o platformă multimedia creată de organizaţia Macromedia iar, în prezent, este dezvoltată şi distribuită de Adobe Systems. De la apariţia sa în anul 1996, Flash a devenit o metodă foarte utilizată pentru adaugarea de animaţii şi interactivitate în cadrul paginilor Web.

Adobe Flash permite manipularea de vectori şi grafică în format raster şi susţine fluxul bidirecţional de audio şi video. Adobe Flash include un limbaj de scriptare denumit ActionScript. Există mai multe produse software, sisteme şi dispozitive care pot crea sau afişa conţinut Flash, incluzând Adobe Flash Player, care este oferit în mod gratuit şi poate fi utilizat în cadrul celor mai utilizate navigatoare Web.

3. Integrarea tehnologiilor

Pentru a ilustra în mod concret modul în care tehnologiile prezentate conlucrează pentru ca serverul să comunice cu clientul, să îi ofere acestuia unele informaţii şi în acelaşi timp să culeagă de la el alte informaţii, considerăm un caz destul de des întâlnit în prezent pe Internet – crearea unui cont pe un sit. Astfel, situl *www.testez.ro* (domeniul este fictiv) doreşte să le ofere vizitatorilor săi fideli posibilitatea de a accesa informaţii personalizate, în funcţie de preferinţele fiecăruia. Pentru aceasta se foloseşte un sistem de conturi personalizate. Pentru a avea acces la un astfel de cont, fiecare vizitator trebuie să se înscrie, folosind modalitatea practică care va fi prezentată în continuare.

3.1. Datele problemei

Pentru a rezolva această problemă este necesar să pornim de la identificarea concretă a aspectelor de care trebuie să ţinem seama.

În primul rând, care ar fi datele necesare într-un astfel de proces de înscriere? Acestea sunt prenumele şi numele vizitatorului, eventual formula de adresare (domnul, doamna, etc.) sau sexul, data naşterii, etc. şi adresa de email. Aceasta din urmă se foloseşte pentru a-i oferi vizitatorului informaţii despre starea contului său, pentru a-i trimite alte date solicitate, etc. De asemenea, în procesul de înscriere se mai solicită de obicei un nume de utilizator şi o parolă care vor fi folosite ulterior pentru accesarea contului personal.

În cazul de faţă vom culege în formular următoarele date:

– numele de utilizator – text, maxim 20 de caractere
– parola – text (parolă), maxim 20 de caractere
– prenumele – text, maxim 30 de caractere
– nume – text, maxim 20 de caractere
– sexul – masculin/feminin, ceea ce poate fi reprezentat printr-un singur bit (1/0)
– data naşterii – dată calendaristică
– adresa email – text, maxim 40 de caractere

Pentru a evita o serie de neplăceri ulterioare, pentru aceste date este bine să se mai ia în seamă câteva restricţii. În primul rând, pentru a putea vorbi într-adevăr de conturi personalizate, trebuie să avem un indicator unic prin care să fie denumite acestea. Acest rol îl poate îndeplini numele de utilizator, cu condiţia să nu existe două nume utilizator identice. Pentru a respecta această condiţie este necesar ca în procesul de înscriere să fie verificată existenţa în baza de date a numelui dorit de un anumit vizitator. Dacă numele de utilizator dorit există, vizitatorul trebuie să-şi aleagă altul.

Pentru o mai bună securitate a conturilor, este bine ca parolele corespunzătoare să nu fie foarte mici. Pentru aceasta este bine să se seteze o dimensiune minimă pentru parole – în cazul de faţă stabilim această valoare minimă de 4 caractere.

Multe din facilităţile oferite de sit depind de adresa de email a fiecărui utilizator. Aceasta poate fi greşită foarte uşor, de aceea este bine să fie verificată. O adresă de email trebuie să aibă o anumită sintaxă: *nume@server.com*. În cadrul unei adrese pot exista numai anumite caractere (a-z, 0-9, „.”, ş.a.), nu există spaţii. Adresa trebuie să conţină numai un @, şi cel puţin un punct, dar care nu are voie să fie la mai puţin de două caractere depărtare de finalul adresei. Aceste verificări la nivelul adresei se fac automat, eliminând astfel încă de la început o serie de greşeli destul de uzuale. Totuşi, numai aceasta nu este suficient, deoarece o adresă poate avea o sintaxă validă şi totuşi să nu existe. Pentru a corecta aceasta, după completarea procesului de înscriere se trimite un email de confirmare pe adresa trecută în formular. Înainte de aceasta, utilizatorul este avertizat că dacă nu primeşte în circa 24 de ore un email s-a produs o greşeală în procesul de înscriere şi este sfătuit ca în acest caz să reia procesul. Dacă adresa a fost corectă şi totul a decurs fără erori, vizitatorul primeşte un email cu instrucţiunile de confirmare a înscrierii. În caz contrar, în termen de câteva zile, datele folosite pentru o înscriere eşuată se şterg de obicei din baza de date sau sunt corectate de vizitatorul care revine. Această metodă exclude şi posibilitatea ca un rău-voitor să se înscrie cu adresa unei alte persoane într-un astfel de cont.

3.2. Realizarea aplicaţiei pe partea de client

Următorul pas este acela de a stabili cum vor fi culese informaţiile de la client şi a realiza practic interfaţa necesară. Aceasta se referă de fapt la o pagină HTML destul de simplă. Creăm fişierul *formular.html* după cum se vede în continuare.

Culegerea datelor în paginile web se face în general folosind formularele HTML. În cazul de faţă vom avea un formular cu următoarele câmpuri:

– nume_utilizator – de tipul *text*
– parola – *password*
– prenume – *text*
– nume – *text*
– sex – *butoane radio*
– data_n – *text*
– email – *text*

Pe lângă aceste câmpuri, în formular vor mai fi prezente două butoane unul tip *submit*, pentru trimiterea datelor formularului, iar celălalt tip *reset*, pentru golirea informaţiilor scrise în formular.

După cum am precizat anterior, este necesar ca datele introduse de utilizator să fie verificate înainte de fi înscrise în baza de date. Cel mai potrivit nivel pentru a face acest lucru este acela al clientului, pentru cât mai multe câmpuri posibil. Prin aceasta se reduce traficul inutil care ar putea fi generat de trimiterea unor informaţii greşite la server.

Pentru a realiza aceste verificări, vom folosi JavaScript. Pentru aceasta avem două posibilităţi: includerea codului în cadrul fişierului html sau realizarea unui fişier separat. Pentru o mai bună organizare a lucrului, optăm pentru a doua variantă, creând fişierul *verificare.js*.

Prima verificare pe care o facem este aceea ca toate câmpurile necesare să fie completate. Pentru aceasta vom folosi pentru fiecare câmp funcţia *emptyField(camp)*, care verifică dacă acesta este gol. Dacă un câmp nu este gol, dar conţine numai spaţii, funcţia îl consideră gol, deoarece nu conţine informaţiile cerute.

Mai departe vom verifica lungimea parolei, care am stabilit că trebuie să aibă minim patru caractere. Pentru aceasta vom folosi funcţia *vf_pass(camp)*. Nu putem verifica şi dacă numele de utilizator ales mai există, această verificare poate fi realizată doar pe server, unde există informaţiile despre utilizatorii deja înscrişi.

Următoare verificare pe care o facem are ca obiectiv corectarea eventualelor greşeli la scrierea adresei de email. Pentru aceasta vom folosi funcţia *invalidemail(camp)*. Aceasta face toate verificările pe care le-am precizat anterior.

Toate aceste funcţii sunt apelate de către *verificare(formular)*, funcţie care este lansată la trimiterea formularului (se apasă pe butonul de submit, *Trimite*). Dacă în urma verificării se relevă vreo eroare, se afişează un mesaj iar datele din formular nu se mai trimit. Altfel, până acum am cules datele de la utilizator şi le-am verificat în prima instanţă. Dar este necesar ca aceste date să fie prelucrate şi stocate pe server. Pentru aceasta, setăm atributul *action* al formularului să indice spre scriptul care se va ocupa de aceasta – *inscriere.php*.

3.3. Realizarea aplicaţiei pe partea de server

Am stabilit că vom folosi fişierul *inscriere.php* pentru a prelucra datele transmise de la client. Acesta are mai multe funcţii de îndeplinit.

În primul rând, scriptul trebuie să verifice datele primite. Chiar dacă se face o verificare iniţială la nivelul clientului, autorul paginii nu are control asupra acestuia, aşa încât nu se poate pune baza numai pe această verificare. Aşadar, trebuie reluate toate verificările realizate deja. În plus, aici se execută şi alte verificări care nu sunt posibile pe partea de client, cum ar fi verificarea existenţei numelui de utilizator.

Această verificare se face prin conectarea la baza de date şi căutarea numelui de utilizator. Dacă acesta mai există în baza de date, utilizatorul primeşte un mesaj de eroare potrivit căruia trebuie să aleagă un alt nume de utilizator. Ca un mecanism de siguranţă suplimentar, câmpul *nume_utilizator* are atributul *unic*, ceea ce împiedică două inserarea unei a doua înregistrări cu aceeaşi valoare.

Dacă în urma verificării datelor se constată una sau mai multe erori, aceasta sau acestea sunt afişate, iar utilizatorul se întoarce prin intermediul unei legături la pagina anterioară pentru a corecta ce este necesar.

Dacă verificările nu au relevat nici o eroare, informaţiile din formular se introduc în baza de date. Pe lângă aceasta, un email este trimis pe adresa utilizatorului, conţinând un cod unic de activare. Acesta este trecut şi într-un câmp *cod* din tabel.

Dacă emailul oferit a fost într-adevăr corect, utilizatorul va primi un mesaj conţinând un url pentru activarea contului. Acest url conţine adresa scriptului şi codul de activare. Detectând existenţa codului, scriptul verifică valoarea acestuia în baza de date. Dacă această valoare există, contul corespunzător este activat (practic câmpul *cod* primeşte o valoare nulă). Dacă nu, este afişat un mesaj de eroare, oferind posibilitatea corectării eventualelor greşeli în procesul de accesare a url-ului respectiv.

Acesta este doar un exemplu simplu de pagină web generată dinamic, în care mai multe tehnologii conlucrează pentru obţinerea efectului final. Deşi simplu, chiar şi acest script necesită destul de multe operaţiuni, cele mai numeroase fiind cele de verificare. În exemplul prezentat am verificat datele din mai multe puncte de vedere, iar rezultatul final pare a fi în regulă. Folosind date de intrare aşteptate, nu apare nici o eroare. Dar dacă modificăm de exemplu câmpul *data_n* în formular, scriind un text oarecare, nu va fi transmisă o dată calendaristică la baza de date, deşi totul pare să funcţioneze în continuare corect. Doar vizualizând valoarea care apare în baza de date putem sesiza eroarea. Eroare pentru evitarea căreia trebuie să introducem încă o procedură de verificare, care să valideze data introdusă.

De data aceasta, eroarea a fost sesizată destul de uşor. Dar în practică apar greşeli asemănătoare care se detectează destul de greu. Modalitatea prin care astfel de erori sunt îndepărtate într-o măsură cât mai mare este aceea de a testa intensiv aplicaţiile realizate, de preferinţă folosind utilizatori neavizaţi.

4. Criterii de evaluare a tehnologiilor support pentru dezvoltarea RIA

Criteriile care trebuie avute în vedere în procesul de evaluare a tehnologiilor suport pentru dezvoltarea de aplicaţii RIA sunt prezentate în cele ce urmează:

a) Bogăţia interfeţei cu utilizatorul

Pentru evaluarea bogăţiei interfeţei cu utilizatorul este important să se răspundă la o serie de întrebări precum: câte widget-uri sunt disponibile în cel mai scurt timp pentru dezvoltarea unei interfeţe cu utilizatorul, cum se poate efectua legarea datelor şi legarea evenimentelor utilizând controale identificate?

Unele tehnologii RIA asigură moduri simple de a se adăuga bogăţiei şi un numar sporit de informaţii vizuale prezentate într-un format mai intuitiv, precum furnizarea de interfeţe de programare a aplicaţiei pentru animaţie la nivel de pagină Web.

b) Complexitatea

Mult timp dezvoltatorii au utilizat modelele bazate pe pagini deoarece acest mod de dezvoltare este uşor şi simplu, chiar dacă se poate părea rudimentar. Cu toate că tehnologia RIA are un grad sporit de complexitate, este uşor de învăţat, de creat şi de dezvoltat. De asemenea, poate să interopereze cu tehnologiile Web existente.

c) Interoperabilitatea flexibilă a componentelor

Este important să existe o interoperabilitate flexibilă cu diferitele componente middlware. Interoperabilitatea ar trebui să fie uşor de realizat şi extensibilă în scopul creării de noi widget-uri personalizate. Bibliotecile de widget-uri personalizate, nou create, pot fi reutilizate în dezvoltarea de noi aplicaţii de tip RIA.

d) Actualizarea unui bloc dintr-o pagină

Actualizarea unui bloc dintr-o pagină în locul actualizării unei pagini întregi este un avantaj important deoarece influenţează în mod direct traficul reţelei. Prin actualizarea unui bloc dintr-o pagină, aplicaţia devine mai rapidă, mai uşor de folosit şi se asigură utilizatorilor o prezentare vizuală mult mai intuitivă.

e) Securitatea

Atunci când se doreşte dezvoltarea unei aplicaţii de tip RIA, un factor important care trebuie avut în vedere este securitatea acesteia. Astfel, ameninţările la adresa securităţii nu trebuie să fie mai mari decât cele existente în cazul aplicaţiilor convenţionale.

f) Instrumentele disponibile

Instrumentele de tip medii de dezvoltare integrate (IDE) disponibile pentru dezvoltatori trebuie să fie analizate, luându-se în considerare şi testarea unităţilor şi susţinerea depanării. Instrumentele analizate ar putea fi plug-in-uri dedicate unor editoare sau acceptate.

g) Utilizabilitatea

Din punctul de vedere al utilizatorului, este ideal ca aplicaţia pentru navigator să lucreze cu facilităţile cu care este deja obisnuit. Trebuie avut în vedere faptul că unele facilităţi, cum sunt: salvarea de imagini, utilizarea combinaţiei de taste “Ctrl+F” pentru căutarea de conţinut

pe o pagină şi facilitatea “copy-paste”, nu funcţionează în soluţiile bazate pe Flash.

5. Dezvoltarea de aplicatii pe Internet imbogatite (RIA)

RIA sunt aplicaţii Web ce utiliyeaya date ce pot fi prelucrate atât pe partea de server cât şi de client. Pe partea de client,aplicaţiile sophisticate de tip RIA au o abordare similara cu cea a aplicaţiilor de tip desktop. Aplicaţiile pe Internet îmbogăţite se caracterizează prin asigurarea unei game largi de controale de operare interactive, prin posibilitatea de utiliyare on-line / off-line a aplicaţiei şi prin utilizarea transparentă a puterii de calcul atât pe partea de client, cât şi de server, precum şi a conexiunii reţea.

5.1. Principalele platforme utilizate în dezvoltarea de RIA

În informatică, prin platformă se înţelege un anumit tip de arhitectură hardware şi software care asigură execuţia aplicaţiilor software. Platforma include arhitectura hardware, sistemul de operare, limbajele de programare şi interfaţa cu utilizatorul.

Cele mai performante platforme utilizate în dezvoltarea de RIA sunt:

a) Platforma Adobe AIR : este o componentă cheie a platformei Adobe Flash, este o platformă destinată procesului de execuţie independentă de sistemul de operare care o găzduieşte. Adobe AIR permite implementarea / instalarea aplicaţiilor Flash Player şi Ajax pe un desktop al utilizatorului într-un mod similar unei aplicaţii de tip desktop.

Platforma Adobe AIR asigură un mediu de deyvoltare coerent şi flexibil pentru livrarea de aplicaţii pe dispozitive şi platforme. Platforma asigură suport pentru sistemul de operare Android, sistemul de operare dedicat tabletelor BlackBerrz, precum şi pentru alte sisteme de operare care permit accesul la Internet de pe o serie de dispozitive mobile.

b) Platforma Curl : oferă un mediu puternic şi eficient pentru deyvoltarea rapidă de aplicaţii Web complexe de tip RIA pentru întreprindere. Se caracterizează prin faptul că nu este necesară o component pe partea de server şi poate fi utiliyat orice tip de server Web.

Prin utilizarea platformei Curl, dezvoltatorii pot crea o nouă clasă de aplicaţii Web cu caracteristici îmbogăţite care beneficiază de interactivitate, funcţionalitate, precum şi de de performanţele aplicaţiilor client-server.

Utilizarea platformei Curl este gratuită pentru scopuri necomerciale şi anumite scopuri comerciale prevăzute în contractul de licenţiere. De asemenea, este disponibilă şi o versiune Pro care oferă caracteristici suplimentare la nivel de clase de întreprindere.

Platforma Curl asigură ingineria software destinată unor aplicaţii complexe de mari dimensiuni, diminuând astfel volumul de cod necesar pentru scrierea de aplicaţii. Aplicaţiile create pe baya platformei Curl necesită cu aproape o treime mai puţin cod decât aplicaţiile create pe baya platformelor Adobe Flex şi Ajax.

c) Platforma JavaFX : include o serie de produse bazate pe tehnologia Java care oferă o experienţă consistentă pentru o gamă largă de dispositive cum sunt desktop-urile, dispozitivele mobile şi player-ele Blu-Ray. La origine, platforma JavaFX includea limbajele JavaFX Script şi JavaFX Mobile. JavaFX Script asigură dezvoltarea rapidă de interfeţe 2D bogate utilizând o sintaxă declarative asemănătoare cu SVG.

d) Platforma Ajax : este un grup de metode interdependente de dezvoltare folosite pe partea de client pentru crearea de aplicaţii Web interactive. Prin intermediul Ajax, aplicaţiile Web

pot transmite şi obţine date la / de la server, în mod asincron, fără a afecta afişarea şi comportamentul paginii Web. Ajax se execută pe calculatorul client, permite integrarea mai profundă a celor două medii, client şi server, şi face ca programul să fie mai eficient şi mai uşor de gestionat.

e) Platforma Microsoft Silverlight : este o platformă puternică de dezvoltare destinată creării de aplicaţii pentru Web, desktop şi aplicaţii mobile care asigură utlizatorilor experienţe captivante şi interactive atunci când lucrează on-line sau off-line.

Platforma poate fi considerate un subset al Windows Presentation Foundation (WPF) şi utilizează limbajul XAML. Pe maşinile client trebuie instalat un plugin (Silverlight Runtime), gratuity şi de mici dimensiuni (aprox 2MB), creat pe baya cadrului de lucru .NET, compatibil cu principalele navigatoare, dispoyitive şi sisteme de operare. Microsoft oferă Silverlight pe partea de client pentru Windows şi Mac OS X.

f) Platforma Mozilla Prism : integrează aplicaţii Web cu aplicaţii desktop, permiţând lansarea aplicaţiilor Web de pe desktop şi configurarea independentă de navigatorul Web implicit. Platforma este bazată pe un concept denumit "navigator specific unui site". Acest navigator este proiectat să funcţioneze exclusiv cu o aplicaţie Web, neavând meniurile, barele de instrumente şi alte caracteristici ale unui navigator Web traditional.

Platforma se bazează pe un motor de navigare foarte performant similar cu Firefox pentru a asigura compatibilitatea maximă cu gama de aplicaţii disponibile pe Web.

5.2. Viitorul aplicaţiilor pe Internet îmbogăţite

Popularitatea în continua creştere a aplicaţiilor pe Internet îmbogăţite conduce în mod obligatoriu la necesitatea stabilirii unei strategii şi a unor direcţii de dezvoltare.

Ca punct de plecare este necesar să se analizeze produsele folosite pentru dezvoltarea acestor aplicaţii în scopul determinării acelor produse care vor supravieţui competiţiei în viitorul apropiat. De asemenea, trebuie identificate şi evaluate modificările ce pot apărea în cee ace priveşte platformele de dezvoltare de aplicaţii de tip RIA. Nu în ultimul rând, trebuie evaluate beneficiile care vor fi aduse de aceste aplicaţii companiilor utilizatoare.

În ceea ce priveşte platformele de dezvoltare de aplicaţii de tip RIA, una dintre cele mai bune platforme existente este Adobe Flash Player, o platformă scalabilă de cel mai înalt nivel.

Deşi, în prezent, Microsoft se concentrează în principal pe îmbunătăţirea Internet Explorer, este de aşteptat ca în viitor să deyvolte o platformă de tip RIA mult mai avansată decât Silverlight. Platforma Ajax este, de asemenea, o platformă competitivă, însă nu asigură nivelul înalt de scalabilitate şi de compatibilitate oferit de Adobe, iar dezvoltarea de aplicaţii pe Internet îmbogăţite folosind Ajax este adesea consumatoare de timp.

Limbajele utlizate pentru dezvoltarea de aplicaţii de tip RIA trebuie să fie mai puţin complicate pentru a facilita deyvoltarea de aplicaţii sofisticate. Acest lucru poate fi realiyat prin includerea caracteristicilor de bază ale limbajelor Java şi C++ pentru a crea aplicaţii uşor de modificat, extrem de scalabile şi simplu de utilizat.

În prezent, aplicaţiile de tip RIA adoptate cel mai uşor sunt aplicaţiile de tip video, se preconizează ca, pe viitor, o serie de aplicţii RIA să devină la fel de populare precum sunt în prezent cele de tip video. Astfel, dezvoltatorul Curl se axează pe aplicaţii pentru întreprindere bazate pe Web care asigură scalabilitate, fiabilitate, Securitate, performanţă şi predictibilitate de înalt nivel. Obiectivul său principal este să realiyeye trecerea de la aplicaţiile client-server

la o arhitectură bazată pe Web prin care se va reduce foarte mult costul total de întreţinere.

Succesul platformei JavaFx se datorează faptului ca oferă dezvoltatorului posibilitatea să-şi creeze propriul său player video şi să îl ruleze într-un applet. Din punctul de vedere al unora dintre cei mai importanţi dezvoltatori care utilizează platforma JavaFX, în viitor se va pune un accent mai mare pe aplicaţiile audio, jocuri şi cele destinate utilizatorilor mobili.

Dezvoltatorul Silverlight consideră că, în prezent, doar o mică parte a conţinutului video oferit este de înaltă definiţie. Ca urmare, Silverlight îşi propune să creeze aplicaţii inovative destinate spaţiului video. Dezvoltatorii din cadrul Mozzila Corporation, creatoarea primului client de e-mail bazat pe Web, consideră că în viitor se vor concentra pe realizarea de editoare de cod pe Web îmbogăţite astfel încât momentul când vor apărea aplicaţii de tip editoare video sau Photoshop care pot fi rulate direct în Web nu este foarte îndepărtat.

6. Avantajele/Beneficiile RIA

RIA (Rich Internet Application) este o tehnologie revoluţionară care permite dezvoltarea de aplicaţii ce rulează într-un browser de Internet păstrându-şi totuşi caracteristicile şi funcţionalităţile aplicaţiilor desktop.

Ca **avantaje** RIA sunt cunoscute:

- Accesibilitatea crescută datorită utilizării Internetului: aplicaţiile RIA pot rula de oriunde (chiar şi de pe telefonul mobil) şi la orice oră;
- Costurile reduse de licenţiere şi implementare deoarece nu este necesară instalarea aplicaţiei pe fiecare calculator pe care rulează. Mai mult, versiunile noi se updatează direct pe server şi utilizatorii beneficiază în mod automat de ultima versiune a aplicaţiei.
- Securitatea sporită;
- Funcţionalităţile de tipul Drag and Drop pentru utilizatori şi timpi de răspuns mai buni o dată cu eliminarea comunicaţiilor client-server la fiecare apel făcut de utilizator;
- Lucrul în timp real;
- Numărul pasilor de instalare este redus, procesul de actualizare şi distribuire a aplicaţiei fiind destul de facil sau minimizat semnificativ în comparaţie cu o aplicaţie de tip desktop sau o aplicaţie nativă “open source”;
- Aplicaţiile bazate pe Web sunt, în general, mai puţin vulnerabile la atacuri informatice în comparaţie cu cele executabile.

7. Concluzii

Aplicaţiile RIA realizate prin diversele tehnologii existente produc o interfaţă de utilizator foarte bogată, animată şi interactivă. Astfel site-urile devin mai atractive şi se pot dezvolta mai uşor.

Experienţa îmbogăţită câstigată de utilizatorii care folosesc aplicaţii de tip RIA are implicaţii pe termen lung în ceea ce priveşte afacerile unei interprinderi. Prin extinderea modului de interacţiune dintre utilizatorii finali şi aplicaţii, aplicaţiile de tip RIA asigură modalităţi noi îmbogăţite prin intermediul cărora întreprinderile pot adăuga valoare produselor şi serviciilor oferite.

Domeniul RIA este unul relativ nou şi în continuă expansiune. Se caută noi metode de dezvoltare a aplicaţiilor, care să fie mai uşor de utilizat şi de integrat în sistemele de dezvoltare existente.

Dezvoltarea accentuata a aplicatiilor de tip RIA a condus la aparitia a numeroase platforme de dezvoltare, care sa tina pas cu tendintele moderne si care sa ofere un grad de interactivitate si continut multimedia similar cu cel ce se regaseste in aplicatiile traditionale de tip desktop. Dintre noile alternative de dezvoltare a aplicatiilor Web, prezinta un interes mai ridicat setul de instrumente pus la dispozitie de catre Google, ce permite crearea de aplicatii fara cunostine prea avansate de scripting HTML sau JavaScript. Acest lucru reprezinta o prioritate pentru framework-urile noi aparute, intrucat acestea incearca sa ofere pe langa functionalitatile deja existente pe platformele clasice, si o usurinta a mediului de dezvoltare, pastrand in acelasi timp acelasi nivel de interactivitate si continut ca in cazul aplicatiilor traditionale. In ceea ce priveste incapsularea datelor in pagini web, principalele standarde ce intervin in interschimbul de informatii sunt XML si SOAP.

8. Bibliografie

[1] https://en.wikipedia.org/wiki/Rich_Internet_application

[2] Rossi, G., O. Pastor, D. Schwabe, L. Olsina: Web Engineering: Modelling and Implementing Web Applications, Springer, 2008.

[3] http://rria.ici.ro/ria2010_4/art08.pdf

[4] http://www.asw.ro/rich-internet-application

[5] http://staff.cs.upt.ro/~dan/curs/pm/ria.html

[6] http://www.rria.ici.ro/ria2011_3/art03.pdf

[7] http://www.ietf.org/rfc/rfc2616.txt