**Universitatea Politehnică Bucureşti**

**Facultatea de Electronică, Telecomunicaţii si Tehnologia Informaţiei**

**Arhitectura Three-Tier. AJAX.**

**Student: Oprea Cristinel**

**Grupa: 443A**

**Disciplina: Inginerie Software**

**Profesor: Ştefan Stăncescu**

**Cuprins:**

1. **Context**
	1. **Arhitectura software distribuita**
	2. **Arhitectura Client-Server**

**2. Arhitectura Three-Tier
   2.1. User Interface Tier
   2.2. Application Logic Tier
   2.3. Database Tier**

 **3. AJAX**

* 1. **Prezentare. Implementare.**
	2. **Aplicatii**
	3. **Etapele de parcurgere**

 **4. Bibliografie**

1. **Context**
	1. **Arhitecturi software distribuite**

Sistemele informationale sunt compuse din componente software distribuite pe mai multe masini de calcul si in mai multe zone geografice. De exemplu, datele unei organizatii financiare pot fi stocate pe un calculator mainframe aflat in biroul central. Alte calculatoare midrange aflte in birouri regionale pot generate raporte cu datele stocate in calculatorul mainframe. Prin urmare, cand se proiecteaza un sistem IT, designerii trebuie sa creeeza in mod normal arhitecturi care sunt capabile sa suporte un numar mare de statii de lucru care trebuie sa aiba acces rapid la datele organizatiei.

* 1. **Arhitectura Client-Server**

Arhitectura Client-Server este o metoda de a organiza software-ul astfel incat sa furnizeze acces informational distribuit si resurse de calcul. Software-ul se divide in doua parti: partea de server si partea client. Server-ul gestioneaza resursele sistemului si furnizeaza acces la aceste resurse printr-o interfata de comunicatie bine definata. Partea client utilizeaza interfata de comunicatii ca sa face cereri la resursele serverului, iar apoi serverul trebuie sa raspunda la aceste cereri. Serverele in mod normal sunt disponibile tot timpul la cererile clientului.

Cel mai familiar exemplu al utilizarii tehnologiei client server este modelul de server Web-browser-Web folosit in intranet si in internet. Cea mai simpla forma este modelul de arhitectura two-tier. Acesta inseamna ca sunt doua componente implicate in serviciu. Componentele acestei arhitecturi sunt urmatoarele: calculatorul client care ruleaza un browser Web, un server Web de aplicatii, o conexiune intre ele, si un set de protocoale ce ruleaza pe masina server, in acest caz HTTP, pentru comunicatia intre aplicatii web, HTML pentru reprezentrea datelor, si de obicei un protocol TCP/IP care sa faciliteze conexiunea de retea.

In cea mai simpla forma, un browser cere o pagina o pagina Web care este stocat intr-un fisier HTML pre-creat pe server. In mod normal utilizatorul cauta informatii specifice insemnand ca o pagina personalizata trebuie creata automat, utilizand un program ce cauta informatii in baza de date, proceseza informatiile daca este necesar, si formeaza pagina dorita in mod dinamic.

Problema. Cu toate ca este posibil sa se mentina o baza de date, sa se execute procese aditionale pe aceasta si sa se creeze pagini pe acelasi calculator in rolul de server Web, acesta poate face parte din organizatie Internet si ar putea raspunde la mii de cereri simultan. Pentru ca timpul de raspuns este considerat caracteristica principala de catre useri, este adesea mai practic sa se separeu baza de date si procesarea paginilor intr-un sitem tert de calculatoare.

Rezultatul este numit arhitectura three-tier. Trebuie notat ca, in acest caz, serverul Web joaca rolul unui client in aplicatia de baza de date. CGI (Common Gateway Interface) este un protocol ce face comunicarea intre masina server Web si baza de date. In unele cazuri procesul este dus mai departe, si pentru a simplifica upgradarea sistemului sau maximiza scalabilitatea, se pot utiliza arhitecturi n-tier care separa diferite aplicatii si procese si care rezulta intr-un control mai bun.

**2. Arhitectura Three-Tier**

**2.1. User Inteface Tier**

 Primul nivel este cel al interfetei grafice. Acest nivel gestioneaza intrarile si iesirile datelor cat si afisarea lor. Interfata utilizator contine componente HTML necesare colectarii datelor de la nivelul logic de aplicati si afisarii lor. Limbaje de web-scripting care sunt exploatate la acest nivel de prezenare pot fi Java Server Pages(JSP) sau JavaScript.

**2.2. Application Logic Tier**

 Nivelul logic de aplicatie este nivelul de mijloc, care aduce impreuna interfata cu utilizatorul si baza de date, ascunzand detaliile tehnice fata de utilizator. Operatiile sunt realizate de un server de aplicatii, care contine in container un server Web. Serverul Web raspunde evenimentelor ca receptionarea de date, translatarea, trimiterea si mentinerea job-urilor. Componentele de la acest nivel primesc cereri de la nivelul interfata client. Cererile sunt interpretate in grupuri de actiuni si controlate de job-uri definite in conformitate cu anumite reguli. Java Bean-urile pot executa conexiunile si calculele adecvate, cum ar fi receptionarea/trimiterea informatiilor de la baza de date si efectuarea de procese aferente modelor definite.

**2.3. Database Tier**

 Nivelul bazelor de date este responsabil pentru mdelarea si stocarea informatiilor necesare sistemului si pentru optimizarea accesului la informatii. Datele de la acest nivel sunt preluate de nivelul de aplicatie, care efectueaza operatii de prelucrare si le stocheaza inapoi in baza de date. Pentru ca informatia este un aspect complex in cadrul sistemelor, este esential sa fie structurata si sa isi mentina integritatea. De aceea se folosesc modele dedicate pentru gestionarea lor, unul din ele, foarte folosit, modelul relational.

**3. AJAX**

1. **Prezentare. Implementare.**

Tehnologiile care stau la baza aplicatiilor web clasice(HTML) sunt simple si usor de implentat, insa nu dispun de suport inteligent sau de comportament dinamic si interactiv.

Ajax, Asynchronous JavaScript and XML, rezolva aceste probleme. AJAX nu este un produs specifc, ci mai degraba o suita de tehnologii care permit paginilor web sa fie interactive la fel ca aplicatiile desktop. Poate fi privit ca o imbunatatire a limbajului JavaScript pentru ca in principiu ofera o metoda pe partea client de a obtine date de la server sau de updatare a unor portiuni din pagina fara a reincarca pagina web.

Ajax pune in balanta parte client si partea de server, astfel incat acestea sa nu mai functioneze separat. In acest mod utilizatorul poate manipula pagina web in timp ce datele de la client si de la server comunica intre ele pe fundal.

Pentru a explica acest concept consideram un formular web in care utilizatorul i se cere sa completeze niste campuri(nume, adresa de email, etc.) care trebuie sa fie validate inainte de nivelul logic al aplicatiei. Fara AJAX, sunt posibile doua metode de a realiza acesta validare.

* Prima metoda este ca user-ul sa completeze datele, sa submita pagina, iar validarea sa se petreaca pe server. In acest scenariu utilizatorul experimenteaza timpi morti pana cand noua pagina se incarca
* A doua metoda, de validare pe masina utilizatorului, nu este fezabila pentru ca implica i

In scenariul utilizarii AJAX, aplicatia web poate valida datele introduse in fundal, in timp ce user-ul continua sa tasteze. Ca exemplu, in Google Suggest, user-ul este inampinat de sugestii relevante in timp ce tasteaza.



Concluzie. AJAX ajuta la crearea unor aplicatii web mai versatile si mai interactive posibilitatea de a folosi aplelari asincrone la server in mod transparent, in timp ce utilizatorul lucreaza. AJAX este un tool prin care paginile web devin inteligente si se comporta mai bine la interactiunea cu omul.

Implementare

Tehnologia AJAX este deja implementata in toate browser-erele moderne, ca Mozilla Firefox, Internet Explorer, Chrome, astfel incat utilizatorul nu trebuie sa mai instaleze alte module. Motorul AJAX ruleaza in interiorul browser-ului web, si actioneaza ca un intermediar intre interfata utilizatorului si server

**3.2. Aplicatii**

* Validarea datelor in timp real: datele dintr-un formular(serii de carduri, coduri postale, coduri de cupoane) ce necesita validare pot fi verificate inainte ca acestea sa fie submise
* Autocompletare: Portiuni specificedin date precum adrese de email, nume, orase pot fi completate automat in timp ce utilizatorul tasteaza
* Interfata utilizator specializata: Comenzi de control ca meniuri, arbori, tabele, editoare de text, bara de progres care creaza o mai buna interactiuni pot fi folosite fara reincarcarea paginii
* Mashup-uri: paginile HTML pot obtine date utilizand un proxy de server sau incluzand un script care sa combine date externe cu date din propria aplicatie

**3.3. Etapele de parcurgere al unui AJAX**

* Declansarea unui eveniment: Functiile JavaScipt sunt apelate ca urmare a unui eveniment. In acest caz functia validate() este mapata la evenimentul onkeyup, cand user-ul ridica mana de pe tasta.



<input type="text" onkeyup="validate();">

* Un obiect XMLHttpRequest este creat si configurat: Functia necesita trei argumente: metoda http care poate fi GET sau POST, url-ul serverului, si o variabila booleana care indica daca apelarea este sau nu asincrona. Daca la o iteratie variabila asincron este setata la true atunci trebuie specificata o functie de „callback”.

 request.open("GET", url, true);

 request.onreadystatechange = callback;

* Obiectul XMLHttpRequest face o apelare: cand instructiunea „request.send(null);” este executata, apelarea va fi facuta catre url specific. In cazul metodei GET, data sunt incluse in url ca parametrii.
* Cererea este procesata de server. Un servlet mapat la url-ul validat verifica daca data sunt corecte in baza de date. Un servlet proceseaza un XMLHttpRequest ca orice alta cerere HTTP.
* Returnarea rezultatelor sub forma de XML. Programatorul trebuie sa fie constient de doua lucruri: Content – Type sa fie text/xml si Cache Control setat pe no-cache in cazul in care acelasi request la un server ar returna valori diferite.

response.setContentType("text/xml");

response.setHeader("Cache-Control", "no-cache");

* Obiectul XMLHttpRequest apeleaza functia callback() si proceseaza rezultatul. Browserele retin un obiect al documentelor ce sunt afisate(DOM – document object model). Javascript este o tehnologie in HTML ce are acces la DOM, atfel in cat pagina poate fi modificata dupa ce a fost incarcata.

**3. Bibliografie**

1. THE ARCHITECTURE OF COMPUTER HARDWARE, SYSTEM SOFTWARE, AND NETWORKING( Irv Englander)
2. http://www.oracle.com
3. http://www.openajax.org