**TITLUL LUCRĂRII DE LICENȚĂ**

LUCRARE DE LICENŢĂ

|  |  |  |
| --- | --- | --- |
|  | Absolvent: | **Szabolcs BENE** |
|  |  |  |
|  | Coordonator ştiinţific: | **Assist. Prof. Eng. Cosmina IVAN** |

**2016**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | |  |
| DECAN, |  | | DIRECTOR DEPARTAMENT, |
| **Prof. dr. ing. Liviu MICLEA** |  | **Prof. dr. ing. Rodica POTOLEA** | |

Absolvent: **Szabolcs BENE**

**TITLUL LUCRĂRII DE LICENŢĂ**

1. **Enunţul temei:** *Scurtă descriere a temei lucrării de licenţă și datele inițiale*
2. **Conţinutul lucrării:** *(enumerarea părţilor componente) Exemplu: Pagina de prezentare, aprecierile coordonatorului de lucrare, titlul capitolului 1, titlul capitolului 2,… titlul capitolului n, bibliografie, anexe.*
3. **Locul documentării**: *Exemplu*: Universitatea Tehnică din Cluj-Napoca, Departamentul Calculatoare
4. **Consultanţi**:
5. **Data emiterii temei:** 1 noiembrie 2013
6. **Data predării:** 28 Iunie 2014 (*se va completa data predării*)

|  |  |
| --- | --- |
| Absolvent: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  |  |
| Coordonator ştiinţific: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

**Declaraţie pe proprie răspundere privind**

**autenticitatea lucrării de licenţă**

Subsemnatul(a)**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**, legitimat(ă) cu \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ seria \_\_\_\_\_\_\_ nr. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
CNP \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, autorul lucrării \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_elaborată în vederea susţinerii examenului de finalizare a studiilor de licență la Facultatea de Automatică și Calculatoare, Specializarea \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ din cadrul Universităţii Tehnice din Cluj-Napoca, sesiunea \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ a anului universitar \_\_\_\_\_\_\_\_\_\_, declar pe proprie răspundere, că această lucrare este rezultatul propriei activităţi intelectuale, pe baza cercetărilor mele şi pe baza informaţiilor obţinute din surse care au fost citate, în textul lucrării, și în bibliografie.

Declar, că această lucrare nu conţine porţiuni plagiate, iar sursele bibliografice au fost folosite cu respectarea legislaţiei române și a convenţiilor internaţionale privind drepturile de autor.

Declar, de asemenea, că această lucrare nu a mai fost prezentată în faţa unei alte comisii de examen de licenţă.

In cazul constatării ulterioare a unor declaraţii false, voi suporta sancţiunile administrative, respectiv, *anularea examenului de licenţă*.

|  |  |  |
| --- | --- | --- |
| Data  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  | Nume, Prenume  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  |  |  |
|  |  | Semnătura |

**De citit înainte** (această pagină se va elimina din versiunea finală):

1. Cele trei pagini anterioare (foaie de capăt, foaie sumar, declaraţie) se vor lista pe foi separate (nu faţă-verso), fiind incluse în lucrarea listată. Foaia de sumar (a doua) necesită semnătura absolventului, respectiv a coordonatorului. Pe declaraţie se trece data când se predă lucrarea la secretarii de comisie.
2. Pe foaia de capăt, se va trece corect titulatura cadrului didactic îndrumător (consultaţi pagina de unde aţi descărcat acest document pentru lista cadrelor didactice cu titulaturile lor).
3. Documentul curent a fost creat în **MS Office 2007.** Dacă folosiţi alte versiuni e posibil sa fie mici diferenţe de formatare, care se corectează (textul conţine descrieri privind fonturi, dimensiuni etc.).
4. **Cuprinsul** începe pe pagina nouă, impară (dacă se face listare faţă-verso), prima pagina din capitolul **Introducere** tot aşa, fiind numerotată cu 1. Pentru actualizarea cuprinsului, click dreapta pe cuprins (zona cuprinsului va apare cu gri), Update field->Update entire table.
5. Vizualizaţi (recomandabil și în timpul editării) acest document după ce activaţi vizualizarea simbolurilor ascunse de formatare (apăsaţi simbolul **π** din *Home/Paragraph*).
6. Fiecare capitol începe pe pagină nouă, datorită simbolului ascuns Section Break (Next Page) care este deja introdus la capitolul precedent. Dacă ştergeţi din greşeală simbolul, se reintroduce (*Page Layout -> Breaks*).
7. Folosiţi stilurile predefinite (Headings, Figura, Tabel, Normal, etc.)
8. Marginile la pagini nu se modifică (Office 2003 default).
9. Respectaţi restul instrucţiunilor din fiecare capitol.

Cuprins

[Capitolul 1. Introducere – Contextul proiectului (Heading 1 style) 9](#_Toc452928065)

[1.1. Contextul proiectului (Heading 2 style) 9](#_Toc452928066)

[1.1.1. (Heading 3 style) 9](#_Toc452928067)

[Capitolul 2. Obiectivele Proiectului 11](#_Toc452928068)

[Capitolul 3. Studiu Bibliografic 12](#_Toc452928069)

[3.1. Dezvoltarea aplicatiilor web 12](#_Toc452928070)

[3.2. Comunicare in timp real 12](#_Toc452928071)

[3.3. WebRTC 14](#_Toc452928072)

[3.4. WebRTC vs WebSockets 16](#_Toc452928073)

[3.5. Sisteme similare 17](#_Toc452928074)

[3.5.1. Apizee 17](#_Toc452928075)

[3.5.2. Client Bee 18](#_Toc452928076)

[3.5.3. eFace2Face 18](#_Toc452928077)

[3.5.4. FACEmeeting 18](#_Toc452928078)

[3.5.5. Gearcloud Labs - Mixology 18](#_Toc452928079)

[3.5.6. Net Medical Xpress – RTC Conference Switch 19](#_Toc452928080)

[3.5.7. Proplogic Software Inc - Tawk 19](#_Toc452928081)

[Capitolul 4. Analiză și Fundamentare Teoretică 20](#_Toc452928082)

[4.1. Tehnologii și unelte utilizate pentru dezvoltarea aplicației web 20](#_Toc452928083)

[4.1.1. PHP 20](#_Toc452928084)

[4.1.2. JavaScript 21](#_Toc452928085)

[4.1.3. HTML 21](#_Toc452928086)

[4.1.4. CSS 22](#_Toc452928087)

[4.1.5. GitHub 22](#_Toc452928088)

[4.1.6. PHP 22](#_Toc452928089)

[4.1.7. Apache Server 22](#_Toc452928090)

[4.2. Cerințele sistemului 22](#_Toc452928091)

[4.2.1. Cerinte functionale 23](#_Toc452928092)

[4.2.2. Cerinte functionale 24](#_Toc452928093)

[4.3. Cazuri de utilizare 25](#_Toc452928094)

[Capitolul 5. Proiectare de Detaliu și Implementare 29](#_Toc452928095)

[Capitolul 6. Testare şi Validare 30](#_Toc452928096)

[Capitolul 7. Manual de Instalare și Utilizare 31](#_Toc452928097)

[Capitolul 8. Concluzii 32](#_Toc452928098)

[Bibliografie 33](#_Toc452928099)

[Anexa 1 (dacă este necesar) 34](#_Toc452928100)

**Cuprins**

**Capitolul 1. Introducere – Contextul proiectului (Heading 1 style)**

1.1. Contextul proiectului (Heading 2 style)

1.1.1. (Heading 3 style)

**Capitolul 2. Obiectivele proiectului**

**Capitolul 3. Studiu bibliografic**

**Capitolul 4. Analiză și fundamentare teoretică**

**Capitolul 5. Proiectare de detaliu și implementare**

**Capitolul 6. Testare și validare**

**Capitolul 7. Manual de instalare și utilizare**

**Capitolul 8. Concluzii**

**Bibliografie**

[**Anexa 1 (dacă este necesar)**](#h.lnxbz9)

# Introducere – Contextul proiectului (Heading 1 style)

Titlul capitolului se bazează pe Heading 1 style, numerotat cu o cifra (x. Nume capitol), font Times New Roman de 14, Bold.

Ce se scrie aici:

* Contextul
* Conturarea domeniului exact al temei
* reprezintă cca. 5% din lucrare

## Contextul proiectului (Heading 2 style)

Fontul folosit implicit în acest document este Times New Roman, dimensiune de 12, conform *Normal style*, cu spaţiere la 1 rând (Paragraph, Line spacing de 1.0) şi *Justify*.

Pentru prima linie din fiecare paragraf se foloseşte indentare (implicit in *Normal Style*), iar între paragrafe succesive nu se lasă distanţă suplimentară.

### (Heading 3 style)

Fiecare tabel introdus în lucrare este numerotat astfel: Tabel x.y, unde x reprezintă numărul capitolului iar y numărul tabelului din capitol. Se lasă un rând liber între tabel și paragraful anterior, respectiv posterior.

Tabel 1.1 (Insert caption->Tabel)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Times new roman ( 12) | xxxx | xxxx | xxxx |  |
|  |  |  |  |  |

Fiecare figură introdusă în text este citată (de ex: în figura x.y este prezentată ... ) şi numerotată. Numerotarea se face astfel Figura x.y unde x reprezintă numărul capitolului iar y numărul figurii în acel capitol. Folosiţi (Insert caption->Figura).

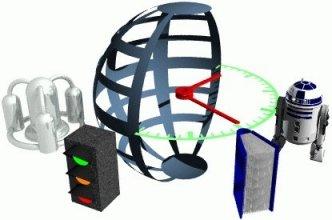


Figura 1.1 Numele figurii (insert->reference->caption->Figura)

Fiecare capitol începe pe pagină nouă.

# Obiectivele Proiectului

În acest capitol se prezintă tema propriu zisă (sub forma unei teme de proiectare/cercetare, formulată exact, cu obiective clare – 2-3 pagini și eventuale figuri explicative).

Reprezintă cca. 10% din lucrare.

# Studiu Bibliografic

În acest capitol se realizează o analiză și evaluare atat a comunicarii in timp real, cat și a unui set de aplicatii care folosesc tehnologii de comunicara in timp real. Se va descrie ce inseamna o aplicatie web, și se vor diferentia solutiile posibile pentru a realiza comnuicare in timp real.

## Dezvoltarea aplicatiilor web

In dezvoltarea a unei aplicatii web, scopurile și functionalitatile aplicatiei sunt separate de mediul in care sunt rulate. In cazul ideal, codul aplicatiilor web nu ar trebuie sa fie dependenta de platforma pe care este rulat, dar din cauza interpretarilor browserelor, pot aparea diferente in functionare, care trebuie tratate separat. O aplicatie web nu este altceva decat un mediu de prezentare și de acces la resurse, din care fac parte atat paginile aplicatiei web cat și resursele de tip imagine, audio sau video. Pentru accesarea acestor resurse se face cu ajutorul unei adrese URI, prin protocolul HTTP. Acest protocol este utilizat pentru a sustine cererile de acces la resursele identificate prin URI.

O aplicatie web reprezinta o colecție interconectată de pagini web cu conținut generat dinamic, menită să ofere utilizatorilor o funcționalitate specifică. Interacțiunea dintre aplicație și utilizatori are loc printr-o interfață web. Orice aplicatie web are la baza arhitectura Client- Server, iar maparea peste aceste componente se face extrem de simplu dacă considerăm că orice browser prin care dorim sa obținem accesul la resursele expuse de aplicație reprezintă Clientul, iar Serverul este cel care răspunde interogărilor pe care clientul le trimite.

Pentru transmiterea interogarilor si a mesajelor, browserul deschide o conexiune catre server prin care ii va trimite un mesaj in care se descrie operatia care se doreste a fi efectuata. In urma trimiterii acestui mesaj, serverul descrie operatia care se doreste a fi efectuata. In urma trimiterii acestui mesaj, serverul verifica daca resursa este disponibila si daca poate sa fie oferita spre afisare. Chiar daca pe server se gaseste resursa respectiva sau nu, clientul va primi un mesaj, ori de eroare, ori de success, impreuna cu continutul resursei accesate de catre utilizator.

Cererile importante si folosite cel mai des in protocolul HTTP sunt GET si POST. GET este adesea utilizat pentru a obține resursele daca acestea sunt disponibile, iar POST oferă posibilitatea introducerii de date in conținutul mesajului transmis serverului.

## Comunicare in timp real

Comunicarea in timp inseamna o telecomunicatie in orice mod, in care utilizatorii pot interschimba orice fel de informatii, text, audio sau video, instant sau cu intarzieri foarte mici.

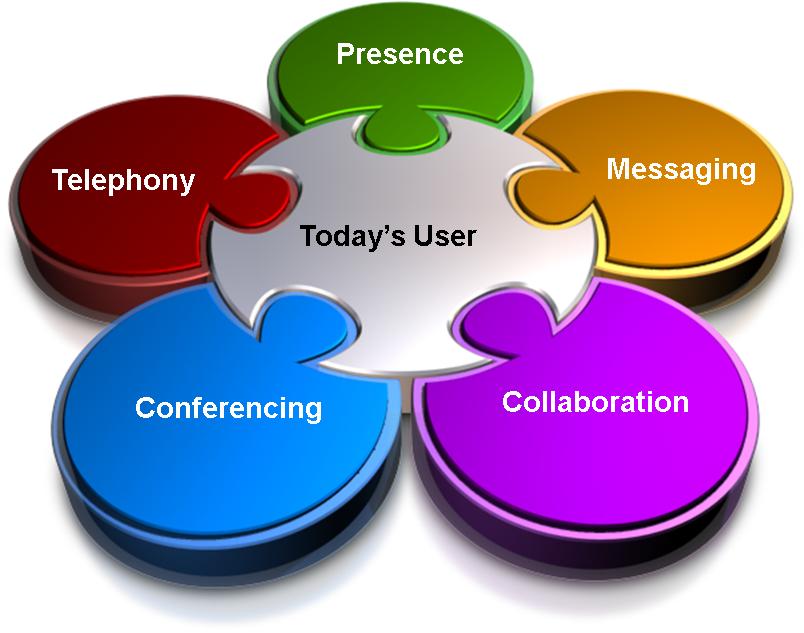
In figura 3.1, se poate observa elementele care construiesc comunicarea in timp real, si cum sunt conectate la utilizator.

figura 3.1

Comunicarea reala se poate imparti in doua moduri: comunicare half-duplex si comunicare full duplex. In comunicare half-duplex, informatiile pot fi transmise in ambele directii, dar nu in acelasi timp, in schimb comunicarea full-duplex, informatiile se pot transmite in ambele directii, in acelasi timp. In general cand vorbim despre RTC, atunci intelegem comunicare peer-to-peer, nu broadcast sau multicast.

In comunicare in timp real, exista intotdeauna o cale directa intre sursa si destinatie, fara ca informatiile sa fie stocate undeva pe durata transmiterii informatiilor. In comparatie cu comunicarile de tip *timeshifting*, informatiile sunt intotdeauna stocate undeva pe un server. In aceasta categorie de timeshifting intra comunicarile prin mail sau comunicare prin commenturi ale retelei de socializare Facebook, unde informatiile sunt salvate pe server, iar intre transmiterea si receptia informatiilor exista o anumita intarziere. In categoria comunicarii in timp real intra urmatoarele: comunicare prin telefon, prin mesaje instante, VoIP, conferinte audio/video, si multe altele.

## WebRTC

In acest subcapitol se va descrie ce inseamna tehnologia WebRTC, si cum s-a ajuns sa existe comunicare in timp real pe web, fara a folosi librarii externe sau pluginuri.

WebRTC este un proiect utilizabil de toata lumea ce asigura comunicare in timp real prin browsere de web si aplicatii mobile. Misiunea de la inceput a fost de a oferi functionalitati de calitate RTC, si care sa se poata integra in diferite platforme mobile sau web, care permite comunicare prin un set de protocoale. WebRTC este sustinut de diferite platforme cum ar fi Chrome, Mozilla sau Opera, si este intretinut de echipele de la Google Chrome.

In Mai, 2011, Google a pornit un proiect open-source, pentru RTC pe web numit WebRTC.[2] Dupa acest proiect, s-a pornit si o serie de lucruri pentru standardizarea protocoalerol relevante in IETF[3] si pentru API-ul respectiv W3C.[4] API-ul a fost facut dupa un proiect anterior facut de WHATWG[6] si a fost referit ca si ConnectionPeer API, iar implementarea conceptului pre-sandardelor a fost facuta la Ericsson Labs.[7]

WebRTC vine cu o gama mare de functionalitati ce permite comunicarea prin text, audio si video, doar prin folosirea browserului. WebRTC-ul permite si partajarea fisierelor sau a ecranului utilizatorului, dar si multe altele.

Tehnologia vine cu avantaje mari, cand este vorba de RTC. In primul rand, functionalitatile oferite sunt usor folosite si implementate, si nu este nevoie folosirea unei liberarii externe sau pluginuri de browsere. Tot ce este nevoie este folosirea unui protocol de instantiere a semnalelor, ce are rolul de a transmite informatii de conexiune de la un client la altul. Al doilea avantaj este ca forteaza programatorii de web sa implementeze propria lor securitate si incriptarea datelor.

Avantajele acestei tehnologii se raspandesc nu numai in afara companiilor, ci si in interiorul lor. Reduce costul telefoanelor interne prin inlocuirea lor cu aplicatie WebRTC, si poate oferi comunicare bogata si neintrerupta intre clienti si angajati.

Arhitectura WebRTC-ului este una complexa, dar folosirea functionalitatilor oferite de aceasta tehnologia este usoara. Arhitectura a fost creata astfel, incat programatorii sa poata folosi si integra cu usurinta functionalitatile WebRTC-ului, iar utilizatorii sa poata folosi aplicatiile web fara descarcarea a unei librarii externe sau pluginuri pentru browseri.

Arhitectura totala se poate observa in figura urmatoare:

.



Fig

La prima vedere se pot observa doua layeri distincte:

1. Programatorii de browseri sunt interesati in API-ul de WebRTC C++
2. Programatorii de aplicatii web sunt interesanti de API-ul de Web

Aplicatiile web vor folosi cabalititatiile de comunicare in timp real oferite de API-ul de Web, oferit de tehnologia WebRTC.

Multe componente sunt abstractizate, oferind creatorilor de aplicatii web sa implementeze propriile lor modalitati de gestiune a datelor cum ar fi componenta de transport is sesiuni folosit pentru diferite protocoale de transport, componenta de STUN/ICE care este folosit pentru stabilirea conexiunii sau componenta de gestionare a sesiunilor.

## WebRTC vs WebSockets

Pentru a implementa o aplicatie web, exista mai multe tehnologii, care sa ofere posibilitatea de a comunica intr-o retea, sau in afara retelei, in timp real. Aceste tehnologii se pot folosi atat separat, cat si combinat, si din cauza aceasta se pot folosi avantajele unor tehnologii fara a se gandi la dezavantajele oferite de celelalte tehnologii.

Una dintre aceste tehnologii este WebSockets. WebSockets este o tehnologie web care ofera comunicare bidirectionala prin protocolul TCP. Este folosita pentru comunicare client-server si server-client, la aplicatii web. Protocolul WebSocket a fost standardizat prima data de IETF, dupa care a fost standardizat API-ul de WebSocket de catre W3C. [x]

Conexiunea este stabilita folosind porturile 80 si 443 in mod implicit. WebSockets fiin bazat pe protocolul TCP, relatia intre protocolul WebSocket si HTTP e de tip handshake, iar potrivit standarului RFC6455, protocolul WebSocket este alcatuit din 2 parti: partea de handshake si transferul de date.[x]

Ca si functionalitate, prima data trebuie stabilita conexiunea intre client si server. Acest lucru se face prin trimiterea mesajelor de handshake intre client si server. Daca handshakeul dintre client si server s-a stabilit cu succes, atunci atat clientul, cat si serverul, poate sa trimita date sau mesaje, deoarece conexiunea este bidirectionala. Dupa ce s-au trimis datele si mesajele, conexiunea intre client si server si poate inchide.

In urma acestor informatii, s-a facut o comparatie intre tehnologia de WebRTC si WebSockets, iar rezultatul se poate vedea in urmatorul tabel:

|  |  |
| --- | --- |
| WebRTC | WebScockets |
| Peer-to-Peer  Acesta este principalul avantaj folosind WebRTC, deoarece nu este nevoie de un webserver pentru transmiterea datelor | WebSockets are nevoie de un webserver centralizat pentru a trimite datele, mesajele sau imagini de video. |
| Comunicarea este bidirecta intre client-client;  Are nevoie de un serviciu extern, numit signalling, prin care se transmit informatii de retea pentru stabilirea conexiunii.  Pentru implementarea unui serviciu de signalling se poate folosi WebSockets, sau diferite librarii care au la baza lor implementare de socketuri (ex. Socket.io) | Comunicarea este bidirecta intre client-server-client, reducand astfel viteza de a trimite date de la un client la altul |
| Suport pentru WebRTC este oferit numai de ultimele versiuni a majoritatea browserelor. Ex: Chrome, Firefox, Opera | Suport pentru socketuri este oferit de toate browserele, fara exceptie. |
| Este browser-browser, iar “camerele” pentru  conversatii sunt create in procesul de signaling | Foloseste handshakeuri HTTP compatibile si porturi default pentru combinarea mai usoara a infrastructurii web existente si a proxyurilor/firewall-urilor |
| Layerul de transport este configurabil | Un API de browser mai usor folosit si implementat |
| Majoritatea aplicatiilor care foloseste implementeaza functionalitatiile de WebRTC folosesc protocolul UDP | Aplicatiile care folosesc socketuri folosesc protocolul TCP |
| Folosind functionalitatea de DataChannel, se pot trimite date neprelucrate de la un client la altul, fara a folosi vreun server de comunicare | Clientii trebuie sa foloseasca neaparat ajutorul unui server pentru a putea comunica intre ei. |
| Datele nu sunt transmise pe cai de “incredere” | Datele sunt transmise pe cai de “incredere” |
| Foloseste servere STUN si TURN pentru a obtine conexiuni cu client care au IP-uri private (clasele A,B,C) | Pentru comunicarea intre clienti nu este necesar sa fie folosit servere de STUN si TURN, deoarece serverul de web decide unde vor ajunge datele de la un client la altul. |
| Pentru a se putea face broadcast (comunicare intre mai mult de 2 clienti in aceasi timp), solutia este aceasi la ambele tehnologii: este nevoie de un server centralizat. Dezavantajul este ca datele mari ce trebuie transmise pot ajunge mai tarziu la client. | |

In urma acestei comparatii facuta, s-a ajuns la decizia folosirii tehnologiei WebRTC, din cauza solutiei si avantajelor oferite de aceasta. Avantajele principale ale aceste tehnologii sunt urmatoarele: comunicare peer-to-peer fara folosirea pluginurilor sau altor librarii externe, schimb de date text/audio/video fara ajutorul unui server intermediar, combinarea cu alte tehnologii pentru procesul de signalling.

## Sisteme similare

Produsele alternative existente pe piata, sunt sisteme care se concentreaza pe calitatea oferita pentru realizarea comunicarii in timp real, dar exista si alte aplicatii care folosesc tehnologia WebRTC pentru alte scopuri, nu numai comunicare. Aplicatiile se concentreaza atat pe calitate, cat si pentru experienta de utilizator. Aceste aplicatii trebuie sa fie atat usor folosibile, cat si sa ofere calitate. In urmatoarele subcapitole vor fi prezentate unele aplicatii care folosesc pentru comunicare intre utilizatori tehnologia WebRTC.

### Apizee

Este o platforma Saas (Software as a Service) pentru comunicare de web si mobil, in timp real. Solutia oferita de ei ajuta la discutiile de afaceri si imbunatateste colaborarea intrea clientii digitali.

In spatele acestei aplicatii este apirtc-ul, prin care aplicatia realizeaza conectarea si comunicarea intre client. Functionalitatile principale ale aplicatiei sunt: schimb de mesaje de text, audio/video call, colaborare vizuala, multi-dispozitiv, 100% fara pluginuri, integrare si conectare cu alte aplicatii

Aplicatia a castigat mai multe premii la diverse conferinte cum ar fi “Stratégie Clients”, “Oscars of economy Côtes d’Armor” si altele. Pentru mai multe informatii se poate accesa linkul <https://apizee.com/>

### Client Bee

Aceasta aplicatia are la baza tot technologia webRTC, dar este folosita pentru un total alt scop decat schimbul de mesaje sau conversatiile prin web.

Aceasta aplicatie este folosita pentru: plati online, facturare automata, inregistrare video, video in-browser, reminder automat. Pentru mai multe informatii se poate accesa linkul <https://clientbee.com/>

### eFace2Face

Aceasta aplicatie ofera solutie pentru vanzarea de video bazate pe web si solutii de e-signing.

Imbunatateste productivitatea si efficient cu converatii video, partajarea informatiilor si semnaturilor electornice. Ofera posibilitatea sa te conectezi la mai multi clienti in aceasi timp si este usor de integrat in infrastructure deja existente online. Pentru mai multe informatii se poate accesa linkul <https://eface2face.com/>

### FACEmeeting

O aplicatie simple de “1 click” pentru intalniri online cu conversatii HD audio/video, schimb de mesaje de text, si partajarea fisierelor. Pentru ca foloseste webRTC, nu trebuie downloadat si instalat niciun plugin, iar informatiile de media si de semnalare intre clienti sunt criptate si encodate.

Aplicatia momentan merge numai pe Google Chrome, dar este gratis, in comparatie cu majoritatea aplicatiilor. Pentru mai multe informatii se poate accesa linkul <https://facemeeting.com/>

### Gearcloud Labs - Mixology

Aceasta companie dezvolta o technologie care lasa oamenii si utiliatorii sa impartaseasca experientele noi si unice legat de grafica si streaming de video. Ei o numesc technologia asta “Mixology” si poate fi folosit pe o varietate de aplicatii ce ofera: distractie, jocuri, evenimente live si multe altele.

Technologiile care le folosesc sunt webRTC pentru solutiile de procesare video si OpenCV pentru imaginile si informatiile grafice. Pentru mai multe informatii se poate accesa linkul [www.gearcloudlabs.com](http://www.gearcloudlabs.com/)

### Net Medical Xpress – RTC Conference Switch

Aceast companie folosesc webRTC-ul pentru dezvoltarea aplicatiei lor interne numit “RTC Conference Switch”, pentru a lasa utilizatorul sa-si creeze propria organizatie, grup, si subgroup pentru conferinte de video. Pentru mai multe informatii se poate accesa linkul <http://www.nmxs.com/>

### Proplogic Software Inc - Tawk

Aceasta companie a creat aplicatia de chat numit “Tawk”.

Functionalitatiile principale ale aplicatiei sunt urmatoarele: integrarea aplicatiei in alte aplicatii folosind doar un simplu snippet de cod, chat audio/video, impartasire de monitor, transfer de date securizat si criptat, impartasirea textelor si a fisierelor, optiune fullscreen, camera de conversatie pana la 4 utliziatori. Pentru mai multe informatii se poate accesa linkul <https://tawk.com/>

Dupa cum se observa fiecare aplicatie are la baza technologia WebRTC, si ca si interfata de UI este folosit HTML5. Diferentele intre aceste aplicatii sunt functionalitatile, si serverele pe care ruleaza aplcatia web. Fiind WebRTC-ul raspandit in lume, exista mult mai multe aplicatii care folosesc aceasta technologie, iar mai sus sunt prezentate doar o parte mica din setul de aplicatii care exista peste tot.

# Analiză și Fundamentare Teoretică

În acest capitol vor fi prezentate tehnologiile și uneltele care au contribuit la implementarea aplicației web. Printre acestea se numără: WebRTC, PHP, JavaScript, HTML, CSS, XAMPP, Github etc. Se descriu detaliile considerate necesare în vederea întelegerii codului sursă, cu referire către literatura de specialitate, care detaliaza tehnologiile enumerate.

## Tehnologii și unelte utilizate pentru dezvoltarea aplicației web

### PHP

PHP este un limbaj de programare de tip script, folosit pe partea de server a aplicațiilor web. Numele original provenit de la *Personal Home Page* a fost ulterior schimbat in *PHP: Hypertext Preprocessor*.

Codul PHP poate fi integrat cu codul de HTML și poate fi folosit in differite combinații pentru diferite sisteme web. Codul este procesat de un interpretor ca un modul in partea de server sau ca un executabil CGI. Serverul de web combină rezultatele codului PHP interpretat și executat, incluzând imagini și pagini de web generate. Codul PHP poate fi executat și din CLI, și poate fi folosit și pentru implementarea aplicațiilor grafice singulare.[6]

Sintaxa pentru folosirea codului PHP este *<?php…?>* și poate fi combinat cu HTML-ul în urmatorul mod.

*<body>*

*<?php echo ‘<p>Hello World</p>’;?>*

*</body>*

Deși se poate combina cu HTML-ul, nu exista nicio regulă să combini aceste două limbaje, iar dacă fișierul conține numai cod PHP, atunci este indicat sa omitem etichetele de închidere.

*<? ’Hello world’;*

Variabilele trebuie anotate cu caracterul “*$”,* iar tipul variabilei nu trebuie specificat. Expresiile trebuie să se termine cu caracterul “*;”*, altfel codul PHP nu poate fi interpretat corect și ne va da mesaje de eroare. PHP-ul este similar cu limbajele de programare C/C++/Java, avand atât sintaxe și cuvinte cheie similare cât și logica expresiilor *if* sau *for*.

De la versiuna PHP5, sunt introduse variabile *private* și *protected*, *metode*, împreuna cu clase *abstracte și finale*. A fost introdusă și o modalitate standard de a declara *constructori* și *destructori*, similar limbajelor de programare C++.

### JavaScript

*JavaScript este un limbaj de programare interpretabil, netipizat, dinamic și de nivel înalt.*[1] *A fost standardizat în specificația de limbaje de la ECMAScript.*[2] Este una dintre tehnologiile de bază pentru producția aplicațiilor web. *Marea majoritate a websiturilor folosesc această tehnologie și este întretinută de toate browserele moderne*.[1] Prefixul *Java* sugerează incorect că are legatură cu limbajul de programare Java, iar sufixul *Script* sugerează ca nu este un limbaj de programare “*real*”.

JavaScript-ul rulează pe partea de client, de obicei pe browser, și datorită acestui lucru poate să raspundă rapid la interacțiunile utilizatorului, și chiar dacă are defectele lui de design și arhitectură, este foarte puternic și popular, fiind întretinut de toate browserele.

Una din motivele pentru care JavaScript-ul este atât de popular, este implementarea execuțiilor de cod asincron, și chiar dacă nu ajută la citirea codului, asigură că paginile web sunt rapide și sunt executate in timp minim. Pentru executarea codului în mod asincron trebuie folosite funcții de callback. *O funcție callback este o bucată de cod executabilă, care poate fi plasată ca și argument la o altă funcție, care va fi executat la un moment dat. Executarea funcției poate fi imediată, care se numește callback sincron, sau poate fi întarziată, care se numește callback asincron.*[3]

JavaScript-ul este folosit și in alte medii, ce nu au legatură cu aplicații web, cum ar fi documente PDF, widgeturi, jocuri de calculator și aplicații mobile. Existp implementpri și pe parte de server, nu numai client, care ruleazpă în medii cum ar fi *Node.js*.

### HTML

HyperText Markup Language, sau prescurtat HTML, este limbajul de marcare standard pentru crearea paginilor web. *Pe lângă CSS și JS, HTML-ul este una dintre tehnologiile necesare pentru a creea pagini web, dar și pentru a creea interfețe de utilizare pentru aplicatii mobile*.[2]

Elementele de HTML construiesc blocurile de HTML pentru paginile web. HTML permite folosirea atât a imaginilor, cât și a formelor interactive, și asigură că documentele create să fie structurate. Elementele de HTML sunt folosite prin etichete cum are fi ***<img />*** sau ***<input />*** introducând astfel direct conținut în pagina de web. Există și alte etichete cum ar fi ***<p>…</p>*** ce poate contine informații de text și care poate să conțina sub-elemente. Browsere nu afișeaza etichetele de HTML ci le folosesc ca să interpreteze conținutul paginii web.

HTML-ul poate încorpora scripturi cum ar fi JavaScript, care poate afecta comportamentul paginilor. Se mai poate combina și cu CSS pentru a da formă și culoare elementelor de HTML.

HTML5 este a 5-a si, momentan, ultima versiune a limbajului de marcaj HTML. A fost publicată de World Wide Web Consortium ca să îmbunătățească aplicațiile multimedia de ultima generație. HTML5 introduce un API pentru aplicații web complexe, îmbunătățind și extinzând astfel funcționalitățile deja existente.

HTML5 introduce funcționalități noi precum folosirea conținutului de tip multimedia și elementelor grafice. Exemple de etichete multimedia și etichete grafice sunt urmatoarele: ***<video>, <audio>, <canvas>*.**

### CSS

### GitHub

GitHub este o aplicație web folosită pentru salvarea și stocarea aplicațiilor prin servicii Git. Precum Git, GitHub oferă funcționalităti cum ar fi controlul versiunilor și gestionarea codului sursă, însă putem întâlni și funcționalități noi. [3] Proiectele pot fi accesate și manipulate folosind linii de comandă standard. Spre deosebire de Git care este o aplicatie folosită strict din linia de comandă, GitHub oferă atât o interfață grafică web, cât și o interfață grafica desktop. O altă funcționalitate întalnită ar fi integrarea cu aplicațiile mobile.

Utilizatorii aplicațiilor GitHub au posibilitatea de a se înregistra pentru a discuta despre proiecte și pentru a creea depozite. Utilizatorii neînregistrați au posibilitatea de a vizualiza și de a salva proiectele, însă nu pot creea și modifica proiectele existente. Alte funcționalități ar fi vizualizarea bug-urilor, ale funcționalităților existente, dar și gestionarea noilor task-uri.

### PHP

### Apache Server

## Cerințele sistemului

Cerințele unui sistem pot fi clasificate între cerințe funcționale și cerințe non-funcționale. Cerințele funcționale prezintă o descriere completă a funcționalităților pe care sistemul trebuie să le îndeplinească, ce să se poată realiza utilizând sistemul. Cerințele non-funcționale dictează proprietăți și constrângeri asupra sistemului și sunt specificate atribute de calitate pe care sistemul trebuie să le dețină.

### Cerinte functionale

Capacitățile funcționale definesc ce acțiuni specifice poate sistemul să ofere. Aceste funcționalități sunt urmatoarele: login, logout, register, integrare cu rețele de socializare, adăugare prieteni, chat în formă de text, chat audio/video, trimiterea fișierelor între utilizatori, salvarea conversațiilor.

**Login/Logout/Register**

Utilizatorul va putea sa se înregistreze cu un mail valid și să-și creeze un nume de utilizator unic. După toate acestea utilizatorul va putea să intre în aplicație introducând numele de utilizator și parola setată, și odată intrat in aplicație, va putea ieși din aplicație apasând butonul de logout.

**Integrare cu rețele de socializare**

Utilizatorul isi va putea sincroniza contul de chat cu contul de Facebook, dupa care va putea sa intre aplicatie folosind acel cont de socializare.

**Adăugare prieteni**

Utilizatorul va putea să-și adauge în lista de prieteni alți utilizatori, introducând numele de utilizator al prietenului.

**Chat in formă de text**

Utilizatorul va putea sa comunice cu un alt utilizator din lista de prieteni prin mesaje de text.

**Chat audio/video**

Utilizatorul va putea sa comunice cu un alt utilizator din lista de prieteni prin convorbiri audio și video.

**Trimiterea fișierelor între clienți**

Utilizatorul va putea sa trimită fișiere cu o mărime maxima setată la un alt utilizator din lista de prieteni.

**Salvarea conversațiilor**

Aplicatia va salva conversațiile de mesaje între utilizatori, ca aceștia să poată vedea conversțtiile mai vechi cu alți utilizatori.

### Cerinte functionale

În orice sistem informatic, cerințele non-funcționale sunt identificatorii de calitate ai sistemului. Dacă în subcapitolul precedent am vazut ce funcționalități trebuie să pună la dispoziție sistemul, în următoarele paragrafe vom decide și analliza care sunt principalale calități și constrângeri impuse sistemului.

**Accesibilitate**

Aplicația web este ușor accesibilă dintr-un web browser. Utilizatorul trebuie să introducă URL-ul de baza a aplicației, după care va putea naviga între paginile aplicației. Linkurile pentru înregistrare și logare vor fi accesibile în pagina de bază a aplicației și utilizatorul va fi redirecționat întotdeauna către paginile dorite, fără a mai da clickuri suplimentare.

**Deploiare**

Instalarea aplicatiei pe server este ușor realizată, prin copierea codului sursă pe serverul de aplicații. Pentru a vedea modificări pe partea de client, tot ce trebuie sa facă utlizatorul este să reîncarce pagina web, iar modificarile de server se vor putea vedea instant, fără reîncarcarea paginilor.

**Extindere**

Extindarea aplicației este usor realizabilă datorită structurii codului sursă, care contine două părți majore: partea de server și partea de client. Partea de server are o arhitectura MVC, care ușureaza adăugarea noilor pagini web. Logica din spatele paginilor web se poate schimba cu ușurință deoarece interfața de utilizator este decuplat de controllerul ei corespunzător.

**Performanță**

Performanța aplicației este ridicată, deoarece serverul este folosit pentru semnalarea și transmiterea informatiilor de conexiune de la un peer la altul. Transmiterea mesajelor de chat și a frame-urilor video se face rapid, deoarece utilizatorii sunt conectați direct între ei.

**Compatibilitate**

Aplicația este compatibilă atât pe windows, cât și pe linux sau alte sisteme de operare, deoarece este o aplicație web. Majoritatea browserelor majore, Opera, Chrome, Firefox, chiar și ultima versiune de Internet Explorer, au implementată tehnologia WebRTC, și din cauza aceasta aplicația web poate fi folosită pe oricare din browserele menționate mai sus.

**Securitate**

Pentru folosirea aplicației și a funcționalităților majore oferite de aplicație, serverul de aplicație trebuie sa fie securizat cu SSL. Astfel informațiile trimise de la server-client și client-server vor fi încriptate. Pentru a naviga în aplicație folosind *https,* serverul de aplicații trebuie să conțină certificatul de încredere împreună cu cheile de încriptare, iar browserul trebuie să accepte aceste certificate, și să adauge aplicația în lista de aplicații acceptate.

## Cazuri de utilizare

Cazurile de utilizare au menirea să ofere o perspectivă globală asupra comportamentului și funcționalităților puse la dispoziție utilizatorilor. Cerințele funcționale ale sistemului nu reprezintă altceva decât cazuri de utilizare posibile, însă nu toate cerințele funcționale trebuie tratate ca și cazuri de utilizare.

Use CaseCazurile de utilizator se poate observa in figura de mai jos:

**Fig. 4.1 Cazuri de utilizare pentru user**

**CU1. Titlu:** Login

**Descriere**: utilizatorul simplu acceseaza aplicatia și se autentifica

**Actor primar**: utilizator simplu

**Precondiţii**: utilizatorul are un cont care consta intr-un nume de utilizator și o parola

**Postconditii**: utilizatorul vizualizează cu success pagina principala

**Principiu**: utilizatorul selectează pagina de autentificare

**Scenariu de success**: Sistemul afişează pagina dorită

**Scenariu de eşec**:

a. utilizatorul introduce un nume de utilizator incorect

b. utilizatorul introduce o parola incorecta

**CU2. Titlu**: Login folosind Facebook

**Descriere**: utilizatorul simplu acceseaza aplicatia și se autentifica folosind Facebook

**Actor** **primar**: utilizator simplu

**Precondiţii**: utilizatorul are un cont pe reteaua de socializare Facebook care consta intr-un nume de utilizator și o parola

**Postconditii**: utilizatorul vizualizează cu success pagina principala

**Principiu**: utilizatorul selectează pagina de autentificare folosind Facebook

**Scenariu** **de** **success**: Sistemul afişează pagina dorită

**Scenariu** **de** **eşec**:

a. utilizatorul introduce un nume de utilizator incorect

b. utilizatorul introduce o parola incorecta

**CU3**. **Titlu**: Logout

**Descriere**: utilizatorul simplu acceseaza aplicatia și se deconecteaza

**Actor** **primar**: utilizator simplu

**Precondiţii**: utilizatorul este inregistrat in aplicatie cu un nume de utilizator și o parola

**Postconditii**: utilizatorul vizualizează pagina de start

**Principiu**: utilizatorul selectează optiunea de deconectare

**Scenariu** **de** **success**: Sistemul afişează pagina de start

**Scenariu** **de** **eşec**:

a. utilizatorul nu este redirectionat catre pagina de start a aplicatiei

b. numele de utilizator și parola utilizatorului nu sunt resetate

**CU4**. **Titlu**: Inregistrare

**Descriere**: utilizatorul simplu acceseaza aplicatia și se inregistreaza

**Actor** **primar**: utilizator simplu

**Precondiţii**: utilizatorul nu este inregistrat in aplicatie

**Postconditii**: utilizatorul vizualizează pagina de start

**Principiu**: utilizatorul selectează optiunea de inregistrare

**Scenariu** **de** **success**: sistemul creeaza un cont de utilizator și afişează pagina de start

**Scenariu** **de** **eşec**:

a. numele de utilizator exista in baza de date, iar contul nu este creat

b. utilizatorul nu este redirectionat catre pagina de start a aplicatiei

**CU5**. **Titlu**: Conversatie de tip text

**Descriere**: utilizatorul simplu initiaza o conversatie de tip text cu un alt utilizator

**Actor** **primar**: utilizator simplu

**Precondiţii**: utilizatorul este autentificat in aplicatie și conexiunea cu un alt utilizator s-a efectuat

**Postconditii**: utilizatorul trimite mesaje de tip text catre un alt utilizator

**Principiu**: utilizatorul initiaza o conversatie de tip text

**Scenariu** **de** **success**: sistemul trimite mesaje de tip text catre un alt utilizator

**Scenariu** **de** **eşec**:

a. conexiunea cu cel de-al doilea utilizator a esuat

b. mesajul de tip tip text nu este transmis

c. mesajul de tip tip text nu este transmis correct sau in totalitate

**CU6**. **Titlu**: Conversatie audio/video

**Descriere**: utilizatorul simplu initiaza o conversatie audio/video cu un alt utilizator

**Actor** **primar**: utilizator simplu

**Precondiţii**: utilizatorul este autentificat in aplicatie și conexiunea cu un alt utilizator s-a efectuat

**Postconditii**: utilizatorul converseaza prin camera web și microfon

**Principiu**: utilizatorul initiaza o conversatie audio/video

**Scenariu** **de** **success**: utilizatorii inregistrati in conversatia audio/video poarta o conversatie folosind microfonul și camera web

**Scenariu** **de** **eşec**:

a. conexiunea cu cel de-al doilea utilizator a esuat

b. imaginea utilizatorului folosind camera web nu este afisata

c. sunetul nu se aude

**CU7**. **Titlu**: Transfer de fisiere

**Descriere**: utilizatorul simplu transmite fisiere unui alt utilizator

**Actor** **primar**: utilizator simplu

**Precondiţii**: utilizatorul este autentificat in aplicatie și conexiunea cu un alt utilizator s-a efectuat

**Postconditii**: utilizatorul transfera fisiere cu succes

**Principiu**: utilizatorul transmite fisiere unui alt utilizator

**Scenariu** **de** **success**: utilizatorii conectati pot face schimb de fisiere

**Scenariu** **de** **eşec**:

a. conexiunea cu cel de-al doilea utilizator a esuat

b. fisierul nu este transmis

c. fisierul nu este transmis in totalitate

**CU8**. **Titlu**: Istoria conversatiilor de tip text

**Descriere**: utilizatorul simplu vizualizeaza istoria conversatiilor

**Actor** **primar**: utilizator simplu

**Precondiţii**: utilizatorul este autentificat in aplicatie

**Postconditii**: utilizatorul vizualizeaza istoria unei conversatii

**Principiu**: utilizatorul deshide și vizualizeaza istoria unei conversatii

**Scenariu** **de** **success**: utilizatorul poate vizualiza cu succes istoria conversatiilor de tip text

**Scenariu** **de** **eşec**:

a. conexiunea cu cel de-al doilea utilizator a esuat

b. istoria unei conversatii nu poate fi vizualizata

# Proiectare de Detaliu și Implementare

Împreună cu capitolul precedent reprezintă aproximativ 60% din total.

Scopul acestui capitol este de a documenta aplicaţia dezvoltată în aşa fel încât dezvoltarea şi întreţinerea ulterioară să fie posibilă. Cititorul trebuie să identifice funcţiile principale ale aplicaţiei din ceea ce este scris aici.

Capitolul ar trebui sa conţină (nu se rezumă neapărat la):

* schema generală aplicaţiei,
* descriere a fiecărei componente implementate, la nivel de modul,
* diagrame de clase, clase importante şi metode ale claselor importante.

# Testare şi Validare

Aproximativ 5% din total.

# Manual de Instalare și Utilizare

În secţiunea de Instalare trebuie să detaliaţi resursele software și hardware necesare pentru instalarea și rularea aplicaţiei, precum și o descriere pas cu pas a procesului de instalare. Instalarea aplicaţiei trebuie să fie posibilă pe baza a ceea ce se scrie aici.

În acest capitol, trebuie să descrieţi cum se utilizează aplicaţia din punct de vedere al utilizatorului, fără a menţiona aspecte tehnice interne. Folosiţi capturi ale ecranului şi explicaţii pas cu pas ale interacţiunii. Folosind acest manual, o persoană ar trebui să poată utiliza produsul vostru.

# Concluzii

Cca. 5% din total.

Capitolul ar trebui sa conţină (nu se rezumă neapărat la):

* un rezumat al contribuţiilor voastre
* analiză critică a rezultatelor obţinute
* descriere a posibilelor dezvoltări și îmbunătăţiri ulterioare

# Bibliografie

[1] A. Bak, S. Bouchafa, and D. Aubert, "Detection of independently moving objects through stereo vision and ego-motion extraction," in *IEEE Intelligent Vehicles Symposium (IV)*, San Diego, USA, 2010, pp. 863-870.

[2] A. Chambolle and T. Pock, "A First-Order Primal-Dual Algorithm for Convex Problems with Applications to Imaging," *Journal of Mathematical Imaging and Vision,* vol. 40, pp. 120-145, 2011.

[3] R. C. Gonzalez and R. E. Woods, *Digital Image Processing. Second Edition.*: Addison-Wesley Longman Publishing Co., Inc., 2001.

[4] Ajax Tutorial, <http://www.tutorialspoint.com/ajax/>.

2["Google release of WebRTC source code from Harald Alvestrand on 2011-05-31"](http://lists.w3.org/Archives/Public/public-webrtc/2011May/0022.html). public-webrtc@w3.org. Retrieved 2012-09-12.

3 [Charter of the Real-Time Communication in WEB-browsers (rtcweb) working group](http://datatracker.ietf.org/wg/rtcweb/charter/)

**4**[**Jump up^**](https://en.wikipedia.org/wiki/WebRTC#cite_ref-4) [*"WebRTC 1.0: Real-time Communication Between Browsers"*](http://www.w3.org/TR/webrtc/).

 5[*"Introduction — HTML Standard"*](http://www.whatwg.org/specs/web-apps/current-work/multipage/introduction.html#history-1). Whatwg.org*. Retrieved2012-09-12*.

**6**[**Jump up^**](https://en.wikipedia.org/wiki/WebRTC#cite_ref-7) [*"Beyond HTML5: Peer-to-Peer Conversational Video | Ericsson Labs"*](http://www.ericsson.com/research-blog/context-aware-communication/beyond-html5-peer-peer-conversational-video/).

http://searchunifiedcommunications.techtarget.com/definition/real-time-communications

# Anexa 1 (dacă este necesar)

…

Secţiuni relevante din cod

…

Alte informaţii relevante (demonstraţii etc.)

…

Lucrări publicate (dacă există)

etc.