UNIVERSITATEA TEHNICĂ CLUJ-NAPOCA
FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
SECŢIA CALCULATOARE

PROIECT DE DIPLOMĂ

SISTEM DISTRIBUIT DE CONTROL AL INTELIGENȚEI AMBIENTALE

Coordonator științific
Dipl. Ing. Cosmina IVAN

Absolvent
Tudor Armand CIULEANU

Cluj - Napoca
2009

UNIVERSITATEA TEHNICĂ CLUJ-NAPOCA
FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
SECŢIA CALCULATOARE

VIZAT DECAN **VIZAT ŞEF CATEDRĂ**

Prof. Dr. Ing. Sergiu NEDEVSCHI **Prof. Dr. Ing. Rodica POTOLEA**

SISTEM DISTRIBUIT DE CONTROL AL INTELIGENȚEI AMBIENTALE

Absolvent: **Tudor Armand CIULEANU**

1. CONŢINUTUL PROIECTULUI:
   - **A. Piese scrise**
     - Introducere
     - Studiu bibliografic
     - Fundamentare teoretică
     - Specificaţiile şi arhitectura sistemului
     - Proiectarea de detaliu
     - Utilizarea sistemului
     - Punerea în funcţiune şi rezultate experimentale
     - Concluzii
     - Bibliografie
   - **B. Anexe**
     - Listingul codului sursă
     - DVD conţinând programul, baza de date şi documentaţia
2. LOCUL DOCUMENTAȚIEI: Universitatea Tehnică Cluj-Napoca
3. CONSULTANȚI: Șef lucrări Dipl. Ing. Cosmina IVAN
4. DATA EMITERII TEMEI: 30.10.2008
5. TERMEN DE PREDARE: 19.06.2009

CONDUCĂTOR DE PROIECT **SEMNĂTURĂ ABSOLVENT**

Șef lucrări Dipl. Ing. Cosmina IVAN

Cuprins

Lista Figurilor

Capitolul 1 Introducere

1.1 Context

În zilele noastre totul se întâmplă mai rapid, oamenii sunt foarte ocupați, stresați și au timp liber foarte puțin. Astfel timpul efectuării unei operații în propria locuință poate fi scăzut drastic printr-un sistem distribuit de control al inteligenței ambientale. Omul modern este presat de timp și are nevoie de un nivel de confort ridicat. Un sistem de control al locuinței la distanță ar putea înlocui nenumărate drumuri între locul de muncă și locuință a căror scop este verificarea stării unor sisteme în locuință sau schimbarea stării acestora.

Automatizarea locuințelor este o tendință destul de nou apărută în case, apartamente și sedii de firme. În apartamente și case sunt folosite multe din metodele de control ale sediilor de firme(controlul instalațiilor electrice și al instalațiilor de climatizare, controlul ușilor și al ferestrelor, controlul sistemului de supraveghere și alarmă, etc.) dar pe lângă acestea există și funcții specifice locuințelor(controlul sistemelor multimedia, udarea plantelor și a gazonului, hrănirea animalelor de casă, etc.).

Sistemele de control existente sunt orientate pe câte o categorie de dispozitive, nu le înglobează pe toate sub o interfață prietenoasă și ușor de utilizat. Aplicația client a sistemelor actuale trebuie instalată pe computerul de pe care se face controlul, acesta fiind un dezavantaj major.

1.2 Motivație

Am ales tema acestui proiect „Sistem distribuit de control pentru inteligența ambientală” deoarece domeniul automatizarii locuințelor este interesant și cred ca se pot aduce multe îmbunătățiri sistemelor existente. Întrucât nu am nici o legătură directă cu nici una din categoriile dispozitivelor de control(dispozitive electrice, sisteme de securitate, sisteme de supraveghere, sisteme de control a instalațiilor de alimentare cu apă, etc.) încerc să am o viziune de ansamblu asupra acestui domeniu și să înglobez diferitele tipuri de dispozitive în aceeași interfață astfel încât utilizatorul să poată avea acces ușor și rapid la orice dispozitiv din propria casă.

Pe de alta parte implementarea unui astfel de sistem presupune integrarea de cunoștințe din multe domenii diferite și pune în aplicare multe din cunoștințele acumulate de-a lungul celor cinci ani de facultate.

Nu în ultimul rând, un alt factor în alegerea temei a fost dorința de a realiza practic, deține și utiliza un astfel de sistem.

1.3 Obiective

Pe măsură ce oamenii au devenit tot mai ocupați și timpul petrecut acasă tot mai scurt nevoia pentru un sistem de control al locuinței a crescut și ea. Lucrarea de față este o soluție web care încearcă să vină în întâmpinarea acestei nevoi.

1.3.1 Obiective principale:

Obiectivul principal al lucrării este acela de a oferi spre utilizare o variantă de sistem de control eficientă, ușor de utilizat, sigură și care să realizeze controlul tuturor tipurilor de dispozitive dintr-o locuință utilizând aceeași interfață.

În momentul de față exista numeroși producători care oferă diverse sisteme de control a locuințelor. Toate aceste sisteme au un dezavantaj major, faptul că sunt orientate spre o

anumită categorie de elemente controlabile (dispozitive electrice, sisteme de supraveghere video, sisteme de alarmă, electrovalve, etc.). Toate programele de control trebuie instalate pe computerul de pe care se face controlul. Această operație nu este întotdeauna posibilă și întotdeauna va fi o operație consumatoare de timp.

Interfața sistemului distribuit de control al inteligenței ambientale trebuie să fie ușor de folosit pentru orice utilizator și trebuie să fie ușor accesibilă. Deoarece tehnologia web s-a dezvoltat foarte mult în ultimii ani, aplicația va avea o interfață web. Aceasta va putea fi accesată de pe orice computer conectat la Internet utilizând doar un browser web.

Sistemul trebuie să fie scalabil pe orizontală la costuri reduse și fără a se aduce modificări în aplicație sau configurații. Pentru a îndeplini acest obiectiv sistemul va avea o baza de date centralizată în care se vor ține datele tuturor locațiilor conectate la sistem împreună cu configurația acestora. Pentru adăugarea unei noi locații se va face adăugarea ei fizică și cea virtuală adică adăugarea în baza de date. Pentru adăugarea virtuală a unei noi locații se va folosi tot o interfață web destinată administratorilor.

Sistemul va oferi posibilitatea de a vizualiza imagini de la camerele de supraveghere video din locație, de a controla dispozitive electrice, de a controla electrovalve și de a verifica temperatura din locație.

1.3.2 Obiective secundare:

Din interfață utilizatorul va avea posibilitatea adăugării programelor automate. Programele automate pot acționa asupra oricărui dispozitiv din locație, oprindu-l sau pornindu-l la ora prestabilită. Programele automate vor avea posibilitatea de repetare automată.

Utilizatorul va putea verifica prin intermediul interfeței temperatura citită de senzorii de temperatură din locație.

1.4 Prezentarea lucrării

Această lucrare conține specificațiile și detaliile sistemului de control distribuit al inteligenței ambientale. Lucrarea este impărțită în 8 capitole.

Capitolul Introducere este capitolul introductiv în care sunt prezentate motivația și obiectivele acestui proiect.

Studiul bibliografic este capitolul în care sunt descrise sisteme existente de control al locuințelor. În acest capitol sunt prezentate sisteme existente asemănătoare și este făcut un rezumat al referințelor bibliografice.

Partea de fundamentare teoretică prezintă concepte care ajută la înțelegerea părții de arhitectură și implementare. În acest capitol sunt prezentate concepte precum sistemele distribuite, modelele client server pe 2 și 3 nivele, aplicațiile web, serviciile web, sisteme de automatizare a clădirilor și standardul X10. Toate acestea sunt prezentate din perspectiva temei sistemelor de control distribuit a locuințelor.

Specificațiile și arhitectura sistemului este capitolul în care sunt prezentate în prima parte cerințele funcționale și non funcționale împreună cu scenarii de utilizare pentru cerințele funcționale, iar în a doua parte este prezentată arhitectura generală. În partea de arhitectura generală este prezentată arhitectura aleasă în contrast cu alte arhitecturi posibile și este motivată alegerea făcută.

Proiectarea de detaliu prezintă detaliile de implementare ale aplicației. În prima parte sunt prezentate modulele sistemului împreună cu diagramele modulelor și cu modelele arhitecturale folosite. Al doilea subcapitol prezintă modalitățile de realizare a interfeței utilizator. În următoarele subcapitole sunt prezentate modalitățile de comunicare între modulele sistemului, structura bazei de date și alte detalii de implementare.

În capitolul Utilizarea sistemului sunt prezentate echipamentele hardware/software necesare pentru rularea aplicației în fiecare locație după care este prezentat un scurt manual de utilizare al aplicației atât pentru clienți cât și pentru administratori. În final sunt prezentate condițiile de funcționare ale aplicației.

Punerea în funcțiune și rezultatele experimentale prezintă tehnologiile folosite, modul de distribuire a aplicației, rezultatele experimentale, dificultățile întâmpinate și modalitățile de rezolvare a acestora.

Ultimul capitol „Concluzii” prezintă realizările, avantajele aduse de soluție și direcțiile de dezvoltare.

Capitolul 2 Studiu bibliografic

2.1 Continutul bibliografic

Cartea [1] prezintă provocările dezvoltării, implementării si întreținerii sistemelor distribuite de mari dimensiuni. Cartea se axează pe principiile și prioritățile proiectării unui sistem distribuit. Cartea conține multiple exemple de complexități diferite si câteva studii de caz.

Lucrarea [2] prezintă principiile și practicile sistemelor distribuite. Această carte m-a ajutat la întelegerea avantajelor şi metodelor de comunicare într-un sistem distribuit.

Lucrarea [3] prezintă o investigație a potențialului controlului la distanță a sistemelor de automatizare a locuințelor. Această lucrare discută probleme de implementare a acestor sisteme și prezintă soluții posibile utilizând ca mediu de transfer diverse tipuri de tehnologii de rețele.

Documentul [4] conține manualul de referință al plăcii Cerebot care este produsă de Digilent. Acest document conține o descriere funcțională a plăcii Cerebot, o prezentare a capacităților ei și modul de programare a acesteia. Am folosit acest document la implementarea aplicației pentru placa Cerebot.

Documentul [5] conține manualul de referință a modulului PmodTMP produs de firma Digilent utilizat pentru citirea temepraturii. Documentul conține descrierea funcțională a modulului și un exemplu de folosire a acestuia. Am folosit acest document la implementarea aplicației pentru placa Cerebot, la partea de citire a temperaturii.

Documentul [6] conține manualul de referință a modulului PmodHB3 utlizat pentru controlul electrovalvelor. Documentul conține descrierea funcțională a modulului și o descriere a controlului vitezei motoarelor prin modularea lățimii pulsului. Am folosit acest document la implementarea aplicației pentru placa Cerebot, la partea de control a electrovalvelor.

Cartea [7] este un ghid la nivel professional de ASP.net 2.0. Această carte mi-a folosit în special pentru dezvoltarea nivelului de prezentare a aplicației. Deși în aplicatie am folosit Framework-ul .Net 3.5 această carte mi-a fost foarte utilă întrucât princpiile au rămas aceleași și Framework-ul actual pastrează funcționalitatea celui anterior.

Articolul [8] prezintă supozițiile lui Deutsch la zece ani după ce acestea au fost create. Citirea acestui articol m-a ajutat să proiectez și să implementez o aplicație distribuită mai rapidă și mai sigură.

În lucrarea [9] sunt prezentate strategii și tehnici de construire a aplicațiilor sigure într-o lume conectată la rețea. Deși cartea a apărut în 2002 conceptele din aceasta sunt valabile și în ziua de azi. Lucrarea m-a ajutat la securizarea aplicației.

Documentul [10] prezintă modul de utilizare a utilitarului CM11.dll. Acest document m-a ajutat în înțelegerea modului de funcționare și utilizare a modulului CM11.dll care este utilizat în comunicația cu modulul hardware CM11.

Articolul [12] prezintă posibilitatile ca un aparat mobil portabil (PDA sau telefon mobil) sa aibă rolul de aparat de comandă de la distanță pentru sistemele de automatizare a clădirilor.Articolul prezintă studiul celor de la universitatea Carnegie Mellon in care studiaza posibilitatea controlului diferitelor locații(birouri, săli de conferințe, săli de clasă, locuințe, fabrici si unitați militare) prin intermediul dispozitivelor portabile.

Cartea [13] este un ghid avansat de C#. Acesta prezintă modalitați de scriere a paginilor web, a serviciilor web, a componentelor aplicațiilor distribuite, a aplicațiilor Windows și multe altele. Această carte mi-a folosit în special pentru dezvoltarea nivelului aplicației și nivelului de date.

Articolul [14] prezintă tendințele recente de a integra diferite tipuri de dispozitive din locuințe in sisteme software. Articolul oferă o solutie de intergrare a rețelei EIB in sisteme software utilizând UPnP.

În cartea [15] sunt prezentate atacurile web și metode de apărare impotriva acestora. Partea a 3-a a cărții sunt prezentate tipuri de atacuri web și tehnici de implementare pentru a le preveni. Această carte mi-a folosit pentru securizarea aplicației web.

Documentul [16] prezintă protocolul de comunicație cu modulul CM11(controller-ul X10). Acesta prezintă comenzile care pot fi trimise către CM11, modul de transmisie și răspunsurile posibile. Acest document mi-a folosit pentru a înțelege protocolul de comunicație dintre dll-ul CM11.dll și modulul CM11.

Pagina [17] din enciclopedia Wikipedia prezintă istoria automatizării locuințelor, standarde de cablare, arhitecturi de sisteme de automatizare a locuințelor și tipuri de dispozitive controlabile. Această pagină mi-a folosit pentru informarea inițială și ca punct de pornire spre pagini mult mai detaliate specializate pe anumite ramuri ale acestui domeniu.

Lucrarea [18] este un ghid avansat de JavaScript. Aceasta m-a ajutat la scrierea codului care se execută pe sistemul clientului, în special la wrapper-ul pentru controlul video.

2.2 Evoluția sistemelor distribuite de control a locuințelor

Conceptul de automatizare există de mult timp. Totul a început cu un student care a conectat două fire electrice la acele de ceasornic ale unui ceas mecanic pentru a închide circuitul la ora fixă și a aprinde un bec. Pe măsură ce a trecut timpul companiile au dezvoltat sisteme de control a sistemelor de alarmă, actuatorilor, camerelor video, etc. Astfel au fost create primele case automatizate și în puțin timp a apărut expresia „casă inteligentă”. În anul 1988 a apărut cuvântul „domotics” care este inspirat din cuvântul latinesc „domus” care înseamna casă și cuvântul robotics.

În literatura de specialitate articolul „Sisteme client-server pentru supravegherea proceselor prin intermediul Web”, Moraes Fernando [11] definește temenul „Domotics” astfel: „Interacțiunea tehnologiilor și serviciilor aplicată diferitelor clădiri cu scopul de a crește nivelul de securitate, comfort, comunicații și consum de energie”.

La inceput dispozitivele de control erau total independente, acestea nu puteau comunica între ele și pentru fiecare era nevoie de un alt dispozitiv de comandă. Între timp au apărut diverse standarde care grupează tot mai multe dispozitive diferite impreună. Totuși aceste noi tehnologii sunt în fazele incipiente ale dezvoltării; deoarece incă nu există standarde robuste, de multe ori apar probleme de compatibilitate.

Sistemele de control a locuințelor nu sunt întotdeauna acceptate de utilizatorii finali. Scopul cercetătorilor este acela de a introduce automatizarea în viața populației în așa fel încât ea să simtă doar beneficiile ei. De exemplu, în efortul de a face aceste sisteme accesibile ca preț pentru utilizatorii finali s-a susținut folosirea tehnologiei X10 care este ieftină dar destul de veche. Avantajul major al acestei tehnologii este reprezentat de faptul că nu necesită cablaje suplimentare. Tehnologii tot mai noi apar în acest domeniu, astfel tehnologiile fără fir încep să le înlocuiască pe cele cu fir. În articolul „Taking handheld devices to the next level" aparut în publicatia IEEE Computer Society Brad A. Myers [12] spunea că în viitor rețelele din locuințe vor conține nenumărate mașini de calcul, multe dintre acestea cu conexiune wireless. În prezent există multe tendințe de a integra diverse dispozitive din locuințe în sisteme software [14], tendințe care s-au născut din ideea de sisteme de calcul omniprezente. Această evoluție oferă multe posibilități domeniului.

„Din păcate multe dintre facilitățile computerizate ale momentului sunt mai mult o piedică decât un ajutor deoarece interfețele sunt de multe ori prea complexe pentru a putea fi înțelese din intuiție" [12].

În ultima vreme s-a dovedit ca domeniul „Domotics" are multe ramuri interesante dintre care cel al „sistemelor de control al locuințelor de la distanță" s-a dovedit a fi cel unul dintre cele mai provocatoare. Posibilitatea de a avea acces constant la majoritatea dispozitivelor dintr-o clădire, de oriunde, rezolvă multe din problemele pe care utilizatorii le au la întoarcerea în locuință și în același timp ajută la economisirea energiei.

Accesul poate fi făcut de pe diferite dispozitive digitale care pe măsură ce trece timpul sunt tot mai mici și mai performanțe.

2.3 Exemple existente

Întrucât automatizarea locuințelor este o tendință modernă, multe firme au răspuns cererii si au dezvoltat diverse aplicații de control de la distanță pentru sistemele de automatizare a locuințelor.

2.3.1 Control 4

Control 4 este o companie care a fost înființată în 2003 în Statele Unite ale Americii și care produce sisteme de automatizare a locuințelor. Sistemul oferit de Control 4 este foarte complex, firma oferind o gamă largă de produse compatibile cu acest sistem. O configurație cu toate tipurile de dispozitive compatibile cu sistemul oferit de Control 4 este prezentată în Figura 2.1.

. Acest sistem are următoarele funcționalități:

- comandă sistemele audio/video din locuință
- comandă pornirea/oprirea sau schimbarea intensității dispozitivelor luminoase
- controlează temperatura din locație prin intermediul termostatului centralei termice
- permite integrarea sistemului de supraveghere video

Sistemul poate fi utilizat atât din locația în care este instalat prin intermediul telecomenzilor cât și de la distanță prin intermediul unei interfețe web.

Pentru a utiliza un astfel de sistem, se va cumpăra un „house controller", un dispozitiv de control hardware, la care se conectează dispozitivele controlabile. Acest dispozitiv se conectează la Internet pentru a oferi acces de la distanță. După configurarea fizică a locuinței trebuie înregistrat dispozitivul pe site-ul de control al firmei(my.control4.com) pentru a avea access online.

Dispozitivele controlate folosesc tehnologie wireless WiFi sau ZigBee.

Avantajele acestui sistem sunt reprezentate de interfața web, ușor de utilizat și numărul mare de dispoztive de comandă care se pot folosi.

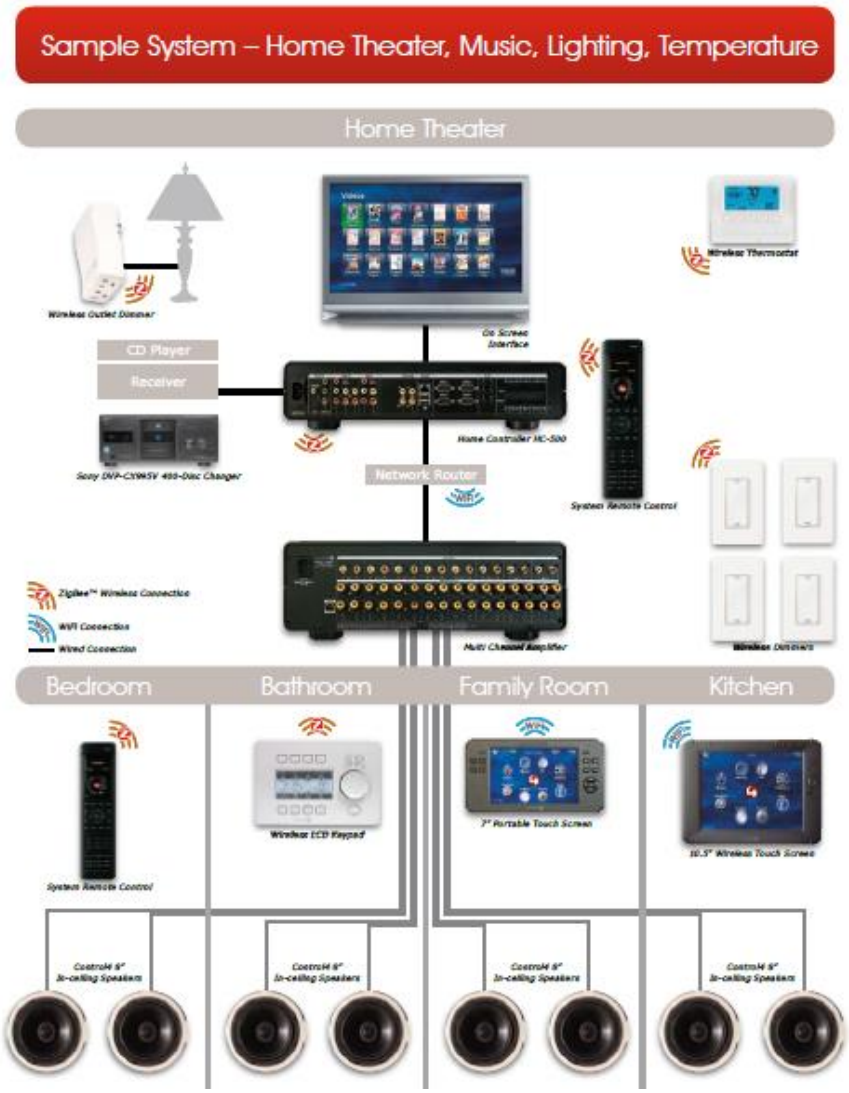


Figura 2.1 Exemplu de sistem oferit de Control 4

Dezavantajul major al sistemului este acela că nu poate controla decât dispozitive din rețeaua electrică neputând da comenzi unor dispozitive din rețeaua de alimentare cu apă sau gaz. Un alt dezavantaj este reprezentat de faptul că administrarea locației trebuie făcută de către utilizatorul final.

2.3.2 Visonic PowerMax

PowerMax este o soluție de automatizare a locuinței oferită de firma Visonic. Soluția PowerMax este destinată utilizării locale. Prin adăugarea pachetului PowerLink la soluția PowerMax se garantează accesul printr-o interfață web la sistemul local.

Pachetul de extensie PowerLink conține un modul care se conectează la sistemul existent și la internet. Prin intermediul interfeței web a pachetului PowerLink(Figura 2.2) se pot controla dispozitivele din locație.

Acest sistem are următoarele funcționalitați:

- comandă pornirea/oprirea dispozitivelor luminoase
- setează temperatura ambientală prin controlul instalației de aer condiționat
- controlul sistemului de alarmă a locuinței

- vizionarea camerelor de supraveghere video
- comandă pornirea sau oprirea diverselor dispozitive electrice
- oferă posibilitatea adăugării de noi utilizatori din interfața client

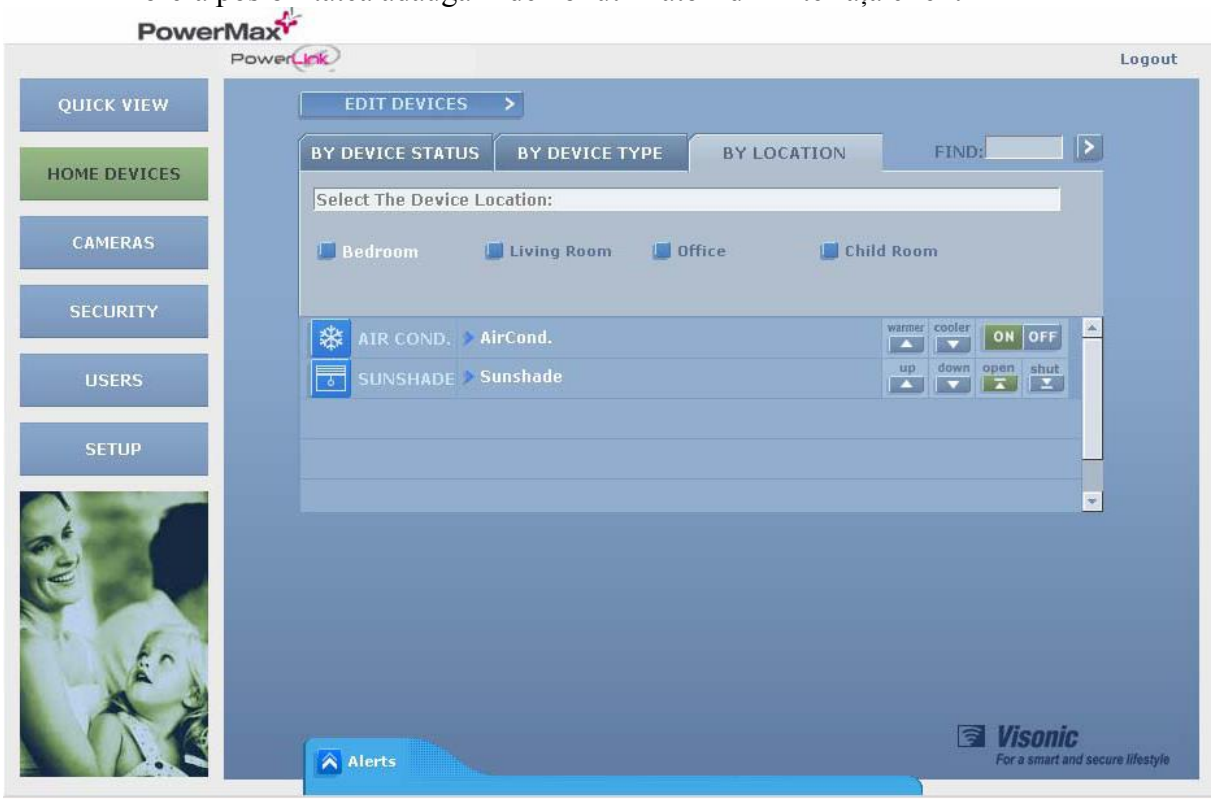


Figura 2.2 Interfața sistemului PowerMax oferit de Visonic

Dezavantajul soluției este reprezentat de faptul că nu poate controla orice tip de dispozitv, deși oferă o gamă mai largă de dispozitive controlabile decât sistemul Control 4. Acest sistem nu oferă posibilitatea controlării dispozitivelor din rețeaua de apă sau gaz.

Capitolul 3 Fundamentare teoretică

3.1 Sisteme distribuite în controlul locuințelor de la distanță

Un sistem distribuit este definit de George Coulouris [2] ca un sistem ale cărui componente sunt localizate în calculatoare care comunică print-o reţea şi îşi coordonează acţiunile numai prin schimburi de mesaje. În concordanţă cu această definiţie sistemele distribuite sunt caracterizate de concurenţa componentelor, lipsa unui control global şi căderea independentă a componentelor. De obicei sistemele distribuite sunt folosite pentru partajarea într-un mod eficient a resurselor. Aceste resurse sunt de multe ori importante sau scumpe şi sunt accesate de clienţi prin intermediul unui server. Concurenţa este normală în astfel de sisteme. Lipsa unui ceas global este o caracteristică fundamentală a sistemelor distribuite aceasta ducând la imposibilitatea sincronizării continue a componentelor. Un avantaj major al sistemelor distribuite este independenţa componentelor: căderea unei componente de obicei nu duce la căderea întregului sistem.

3.1.1 Cele 8 supoziții eronate în programarea distribuită

În industria software termenul de sisteme distribuite este cunoscut de mai multe decenii. Două exemple sunt ARPANET şi SWIFT. Ambele au apărut în aproximativ aceeaşi perioadă, 1969. ARPANET a fost un sistem al Departamentului de Apărare American şi a evoluat în Internet şi a fost un protocol folosit pentru transferurile de bani [1]. Cu toate acestea, în 1994, Peter Deutsch de la SUN, a elaborat 7 supoziții pe care arhitecții și designerii de sisteme distribuite le presupun uneori adevărate, ceea ce se dovedește a fi greșit pe termen.

În 1997, James Gosling adaugă încă o astfel de supoziție [8]. Ipotezele sunt acum cunoscute sub numele de "Cele 8 erori ale programării distribuite":

1. Reţeaua este de încredere.
2. Latenţă este zero.
3. Lăţimea de bandă este infinită.
4. Rețeaua este sigură.
5. Topologia nu se schimbă.
6. Există un singur administrator.
7. Costul transportului este zero.
8. Reţeaua este omogenă.

Prima supoziție **“Rețeaua este de încredere”** este eronată deoarece deși intervalul mediu de timp între două defecțiuni ale unui switch este de 50.000 de ore pot interveni alte probleme. Aceste probleme pot varia de la pierderea alimentării până la ruperea firelor de transmisie. Există unele aplicații pentru care o întrerupere poate fi vitală, de exemplu sistemele de plată online. Pentru a evita pe cât posibil astfel de probleme trebuie puse în balanță riscurile cu investițiile necesare.

A doua supoziție “**Latența este zero**” este și ea falsă. Latența reprezintă timpul necesar pentru parcurgerea distanței dintre sursț și destinație de către informație. Latența este relativ mică pe LAN dar crește foarte tare pe WAN. Latența pune probleme mai mari decât lățimea de bandă. “Cred că este interesant de observat că lățimea de bandă a crescut în ultimii 11 ani de 1468 de ori în timp ce latența(durata unui ping) s-a îmbunătățit doar de 10 ori. Mai mult decât atât, latența este limitată de natură. Timpul minim pentru parcurgerea distanței dintre două puncte pe pământ este determinat de viteza maximă a transmisiei informațiilor și anume viteza luminii. La aproximativ 300.000 km/s întotdeauna va dura cel puțin 30 de milisecunde trimiterea unui ping din Europa în Statele Unite și înapoi, chiar dacă procesarea

s-ar executa în timp real". Din cauza latenței aplicații care se comportă bine pe computerele de dezvoltare merg foarte încet pe serverele de producție.

Următoarea supoziție "**Lățimea de bandă este infinită**" este falsă dar nu este la fel de gravă ca și celelalte deoarece lățimea de bandă crește de la an la an. Pe de altă parte deși lățimea de bandă crește și cantitatea de informație pe care încercam să o transmitem este din ce în ce mai mare.

Cea de-a patra supoziție "**Rețeaua este sigură**" presupune că toți membrii rețelei sunt pașnici. Deși această supoziție a fost creată în 1991 și au trecut aproape două decenii de atunci, rețelele nu sunt mai sigure, chiar dimpotrivă, în fiecare an numărul de atacuri crește cu 64%. În medie o companie este atacată de 32 de ori pe săptămână. Din acest motiv securitatea trebuie implementată în aplicații din primele zile de dezvoltare.

Supoziția "**Topologia nu se schimbă**" reprezintă exact opusul realității întrucât topologia rețelei este într-o continuă schimbare. Din acest motiv în procesul de dezvoltare nu trebuie să ne bazăm pe condițiile de "laborator" și trebuie să ținem cont de schimbările posibile. Rutele specifice nu sunt de dorit deoarece oricând poate ceda un router și ruta va fi schimbată.

Următoarea supoziție "**Există un singur administrator**" poate fi adevărată în cazul în care aplicația este destinată unor companii mici dar în general nu este așa. De obicei departamentele IT ale companiilor au diferiți administratori care sunt împărțiți pe domenii: baze de date, servere web, rețele, Linux, Windows, etc. Problemele pot apărea în momentul în care aplicația este împărțită pe mai multe servere administrate de administratori diferiți, posibil din companii și locații diferite. În cazul în care se face un „update" la aplicație trebuie ținut totdeauna cont de faptul ca „update-urile" nu vor fi făcute simultan și că trebuie să se păstreze sistemul funcțional în continuare.

Cea de-a șaptea supoziție "**Costul transportului este zero**" poate fi interpretată în două moduri:

- Costul transmisiei între nivelele aplicație și transport este zero. Acest lucru este fals deoarece trebuie făcută serializarea informației în biți pentru a transmite datele ceea ce consuma resurse și se adaugă la latență.
- Costul instalării și întreținerii rețelei este zero. Există costuri la cumpărarea echipamentelor, securizarea rețelei și costuri de operare și întreținere.

Ultima supoziție "**Rețeaua este omogenă**"a fost adăugată celor 7 în 1997 de către James Gosling. Astăzi această supoziție este cât se poate de falsă deoarece aproape orice dispozitiv modern se poate conecta la o rețea cu sau fără fir. Câteva exemple de dispozitive ar fi computere pe diverse sisteme(Windows, Linux, MaxOS), PDA-uri, telefoane mobile, diverse dispozitive de jocuri(PlayStation 3, Xbox), DVD playere, etc. În zilele noastre o rețea omogenă reprezintă excepția nu regula. Pentru a trece peste aceasta barieră se folosesc tot mai mult standardele XML și Serviciile Web.

3.1.2 Aspecte de care trebuie ținut cont în proiectarea sistemelor distribuite

La construirea sistemelor distribuite trebuie să luăm în considerare următoarele aspecte: eterogenitatea componentelor, problema sistemelor deschise, securitatea, scalabilitatea, tratarea eşecurilor, concurenţa componentelor şi transparenţa pentru a spori şi a eficientiza utilizarea sistemelor distribuite.

Eterogenitatea se referă la diferenţele privind infrastructura reţelei, hardware-ul componentelor, sistemele de operare folosite, limbajele de programare şi implementările diferiţilor proiectanţi. Aceste diferenţe sunt rezolvate folosind acelaşi protocol de comunicare, utilizând standarde pentru interschimbarea mesajelor, folosind limbaje de programare care pot fi corelate şi definind interfeţe compatibile între sisteme.

Sistemele deschise sunt caracterizate de extensibilitate. Interfaţa cheie a acestor aplicaţii este publicată. Sunt extensibile la nivelul hardware şi la nivelul serviciilor, existând posibilitatea de a adăuga servicii noi sau de a modifica serviciile existente. Sistemele deschise pot fi construite din componente eterogene, produse de diverşi furnizori, dar presupun respectarea standardelor publicate.

Securitatea este un aspect foarte important în sistemele distribuite. De foarte multe ori informaţiile transmise între punctele A şi B au un caracter confidenţial acestea putând varia de la date personale până la secrete militare.

Securitatea resurselor are trei componente:

- confidenţialitatea - evitarea accesului persoanelor neautorizate la informaţia confidenţială
- integritatea – păstrarea informaţiei în stare nealterata
- disponibilitatea – protecţia contra blocării resurselor.

Un sistem este descris **scalabil** când rămâne efectiv şi eficient după creşterea numărului resurselor şi a utilizatorilor. În cazul ideal nici sistemul nici software-ul nu ar trebui să se schimbe când mărimea sistemului creşte, dar acest fapt este greu de realizat.

Eşecurile apar inevitabil fie din cauza utilizatorilor ori din cauza dezvoltatorilor(erorilor nedetectate în faza de dezvoltare a aplicaţiei). Eşecurile în sistemele distribuite sunt parţiale astfel tratarea lor este mai greoaie. Există diferite tehnici pentru tratarea eşecurilor: detectarea, camuflarea, tolerarea, refacera după eşecuri şi prin impunerea redundanţei.

Concurenţa este una din caracteristicile principale ale sistemelor distribuite, unde resursele sunt partajate, iar mai mulți utilizatori accesează simultan aceleași resurse. Sistemul trebuie astfel conceput încât să fie capabil să gestioneze resursele în așa fel încât ele să își păstreze integritatea dar în același timp să fie rapid accesibile (blocarea resurselor nu este întotdeauna o soluție). Sistemul este responsabil pentru asigurarea funcţionării corecte ale acestora într-un mediu concurent.

Transparenţa se referă la imaginea sistemului alcătuit de către utilizator, sistemul pare un întreg şi nu o colecţie de componente independente [1].

3.1.3 Arhitecturi de sisteme distribuite

Arhitectura unui sistem se defineşte ca structura conceptuală alcătuită din componente specifice. Multitudinea sistemelor informatice distribuite implementate până în prezent relevă o varietate mare a arhitecturilor, dar care pot fi totuși încadrate în câteva modele arhitecturale.

Un model arhitectural defineşte modul în care interacționează între ele componentele unui sistem, precum și localizarea (maparea) lor într-o rețea de calculatoare. Modelul arhitectural al unui sistem distribuit are rolul de a simplifica și abstractiza (în sensul de a evidenția caracteristicile esențiale ale sistemului) funcțiile componentelor sistemului. Modelul arhitectural este influențat de:

- plasarea componentelor în cadrul rețelei - căutând să definească modelele corespunzătoare de distribuire a datelor și a prelucrărilor;
- interacțiunea dintre componente - adică, rolurile lor funcționale și modelele de comunicare dintre ele.

Modelele de alocare a sarcinilor de lucru într-un sistem distribuit se reflectă direct asupra performanţelor şi eficacității sistemului rezultat. Localizarea componentelor unui sistem distribuit este determinată de aspectele de performanţă, siguranţă în funcţionare, securitate şi costurile implicate

Putem vorbi despre patru tipuri de arhitecturi principale: modelul client-server, servicii asigurate de mai multe servere, servere proxy, cache-uri şi procese pereche (peer processes). Dintre acestea o să detaliez primele două care sunt relevante din punct de vedere al proiectului.

3.1.3.1 *Modelul Client-Server pe 2 nivele*

Cel mai des întâlnit model de sistem distribuit este modelul client-server. Din punct de vedere istoric este modelul cel mai important. Un sistem distribuit este alcătuit din mai multe servere care oferă diferite servicii pentru procesele numite clienţi. Un server la rândul lui poate să fie client pentru un alt server. Procesele interacţionează pentru a accesa resursele comune.

Modelul Client-server descrie relaţia dintre două programe de calculator, în care un program, programul client, face o cerere la un altul, programul server. Funcțiile standard cum ar fi transmiterea de e-mail-uri, acces web şi accesul la baze de date, se bazează pe modelul client-server. De exemplu, un browser de web este un program client prin care utilizatorul poate accesa informaţii de la orice web server din lume.

Modelul client server a devenit una dintre soluțiile de bază pentru rețele de calculatoare. Majoritatea aplicațiilor scrise în ziua de azi folosesc modelul client-server. Protocoalele care stau la baza internetului cum ar fi HTTP, SMTP, Telnet și DNS au la bază acest model.

Fiecare instanță a aplicației client poate trimite “request-uri” către unul sau mai multe servere de date. Serverele pot accepta aceste cereri după care le procesează și trimit informația cerută către client. Deși acest concept poate fi aplicat din diverse motive la o mare varietate de tipuri de aplicații, fundamentele arhitecturii rămân la fel.

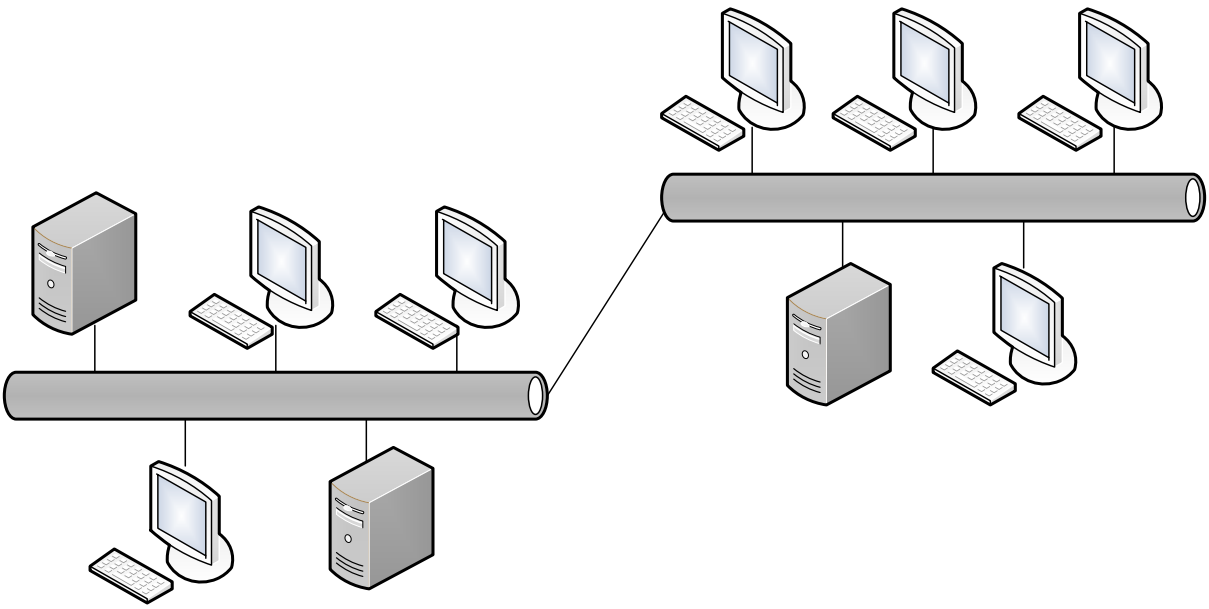


Figura 3.1 Exemplu de arhitectură client server pe 2 nivele

După cum se poate vedea și în Figura 3.1 cel mai simplu model client-server este format doar din două tipuri de hosturi: client și server. Acest tip de arhitectură se mai numește și arhitectura pe 2 nivele (two-tier). Permite dispozitivelor să împartă fișiere și resurse. Cele

două nivele sunt formate din client care formează un nivel și aplicația în combinație cu serverul celălalt nivel.

Cele mai frecvente aplicații client sunt browserele web, clienții de e-mail și online chat. Serverele pot fi și ele de multiple tipuri de exemplu: servere web, servere de fișiere, servere de listare, etc.

3.1.3.2 *Modelul Client-Server pe 3 nivele*

Arhitectura pe 3 nivele („three-tier") este o arhitectură client-server în care interfața, nivelul logic (de „business") și nivelul de stocare de date sunt dezvoltate și menținute ca module independente.

Modelul pe 3 nivele este considerat un design pattern. În afară de avantajele deja cunoscute ale programării modulare cu interfețe bine definite, această arhitectură permite înnoirea sau schimbarea independentă a oricărui nivel. De exemplu schimbarea sistemului de operare în nivelul de prezentare ar afecta doar interfața utilizator.

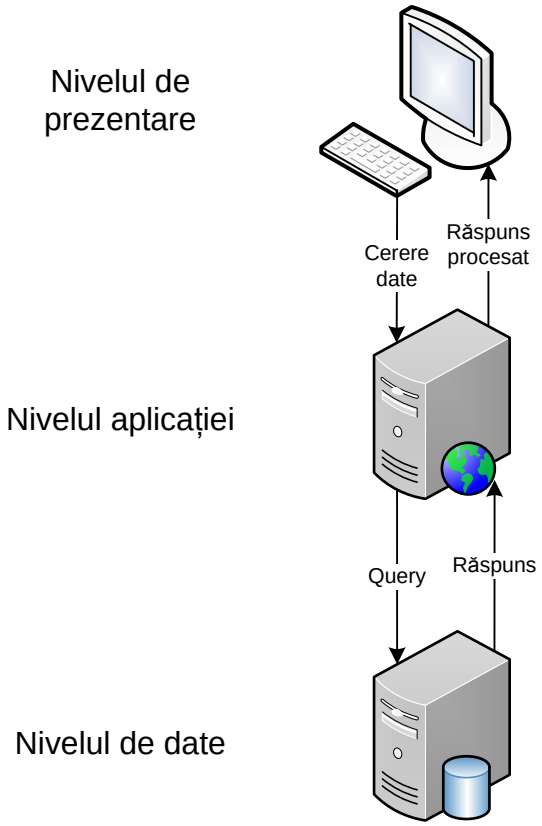


Figura 3.2 Modelul arhitecturii pe 3 nivele

Cele 3 nivele componente (Figura 3.2) sunt:

- Nivelul de prezentare – este nivelul cel mai de sus al aplicației. Nivelul de prezentare este cel văzut de utilizatorul final. Acesta afișează informația obținută de la celelalte nivele în browser (aplicație client).
- Nivelul aplicației – este nivelul cu logica aplicației, în acest nivel este implementată funcționalitatea aplicației (se face procesarea datelor).
- Nivelul de date – este nivelul la care se află serverele de baze de date. Informația este stocată la acest nivel care face ca datele să fie independente atât de logica de

calcul cât și de interfață. Faptul că datele au propriul nivel crește scalabilitatea și performanța aplicației.

La o primă privire arhitectura pe 3 nivele pare foarte similară cu conceptul MVC. Deși par similare cele 2 modele au topologii diferite. În arhitectura pe 3 nivele clientul nu ajunge niciodată direct la nivelul de date, modelul fiind liniar. În arhitectura MVC vederea trimite cererea controller-ului care la rândul sau trimite cererea modelului după care modelul reîmprospătează direct vederea, modelul fiind astfel triangular.

În dezvoltarea aplicațiilor web arhitectura cu trei nivele este des folosită pentru site-uri. Transferul de date între nivele face parte din arhitectură. Protocoalele folosite în acest sens sunt: SNMP, CORBA, Java RMI, .NET Remoting, Windows Communication Foundation, Sockets, UDP și Serviciile Web.

Nivelele separate rulează de obicei pe servere diferite dar acest lucru nu este obligatoriu.

3.1.4 Aplicații web

În ingineria software o aplicație web este o aplicație care este accesată utilizând un browser de web printr-o rețea cum ar fi internetul sau intranetul. Aceste aplicații sunt scrise în limbaje compatibile cu browserele web (HTML, JavaScript, Jscript, VBScript, Java, etc.).

Aplicațiile web sunt foarte populare datorită prezenței pretutindeni a browserelor web și a ușurinței cu care acestea pot fi folosite de către client. Posibilitatea de a împrospăta și întreține aplicații web fără a fi nevoie de distribuirea și instalarea softului pe mii de computere este motivul cheie pentru popularitatea lor.

Datorită ultimelor dezvoltări tehnologice în domeniul browserelor web și a limbajelor de scriere a codului client interfețele au devenit foarte prietenoase. Tehnologii precum (Flex, Flash, JavaScript, Java, etc.) permit executarea în cadrul browserelor web a unor acțiuni specifice aplicațiilor desktop (desenarea pe ecran, drag and drop, redarea audio, accesul la mouse și tastatura). Dezvoltatorii web preferă să folosească codul client („client-side code”) pentru a adaugă interactivitate paginii deoarece nu este necesară reîncărcarea întregii pagini. Recent a apărut tehnologia AJAX care combină codul client cu codul server și permite reîmprospătarea unor controale din pagină fără a reîmrospăta întreaga pagină web.

Un alt avantaj al aplicațiilor web este portabilitatea. Site-urile web se pot scrie în așa fel încât să fie compatibile cu o varietate de browsere web și sisteme de operare. Printre altele pentru dezvoltarea interfețelor aplicațiilor se poate folosi Adobe Flash sau Java applets care sunt compatibile cu majoritatea browserelor web întrucât acestea se pot adaugă sub forma de plug-in-uri.

Tendința actuală în industria software-ului este de a oferi acces web la aplicații care până nu demult erau aplicații locale. Aceste programe permit utilizatorului să plătească o taxă lunară sau anuală pentru folosirea aplicației care nu trebuie instalată local. O astfel de companie este cunoscută sub numele de ASP (provider de servicii aplicație). La astfel de aplicații este mult mai ușor de controlat pirateria întrucât ele nu ajung niciodată să fie deținute de un utilizator final.

După cum am arătat mai sus aplicațiile web au multe avantaje. Pe lângă avantaje ele au și unele dezavantaje: transferurile se fac prin Internet deci pot apărea probleme de securitate, datele nu sunt stocate local deci orice pierdere de conexiune duce la inaccesibilitatea temporară a datelor și pierderea datelor care erau în curs de transfer, browserele web au diferite setări care pot fi schimbate de utilizatorul final și pot face anumite aplicații să nu mai funcționeze.

3.1.5 *Securitatea aplicațiilor web*

Internetul a devenit aleea preferată a criminalilor pentru a împrăștia „malware” – software murdar. Aceste programe sunt concepute special pentru a infiltra ori deteriora un sistem fără ca proprietarul să fie informat. O amenințare pe web este considerată orice amenințare care utilizează internetul pentru a facilita crima cybernetică.

Amenințările web utilizează multiple tipuri de software murdar care utilizează protocoalele HTTP, HTTPS. Amenințările „Threat” web pot produce pagube diverse: pagube financiare, furt de identitate, pierderea de informații/date confidențiale, furtul codurilor de acces, stricarea reputației personale sau a firmei și diminuarea încrederii consumatorilor în comerț online și online banking.

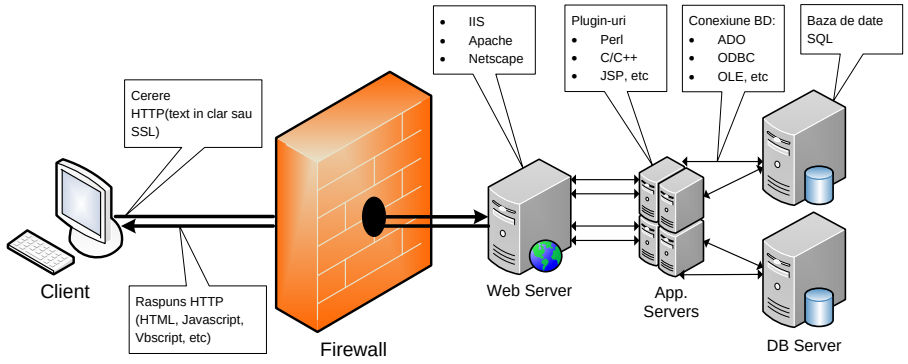


Figura 3.3 Configurație tipică web

Arhitecturile moderne sunt din ce în ce mai robuste și mai greu de pătruns de către persoane nedorite. În Figura 3.3 se poate observa o configurație web tipică în care există o ierarhie client -> firewall -> server web -> servere de aplicație -> server baza de date. Protecția zonei serverelor este dată de firewall. Majoritatea serverelor web sunt despărțite de Internet printr-un firewall care permite doar transferurile http care sunt considerate sigure. Hackerii nu luptă cu zidul (firewall-ul) deoarece au ușa deschisă (traficul Internet). Un hacker are nevoie doar de un browser web și o conexiune la Internet pentru a executa un atac.

Aceste atacuri nu pot fi prevenite de către firewall-uri deoarece porturile *80(HTTP) și 443(HTTPS) sunt de obicei deschise. Din acest motiv precum se poate vedea și în Figura 3.4 url-ul poate acționa ca o “rachetă de croaziera” [15]. După cum se poate vedea în exemplul din Figura 3.4 fiecare nivel din URL ne duce tot mai aproape de nivelul de date.

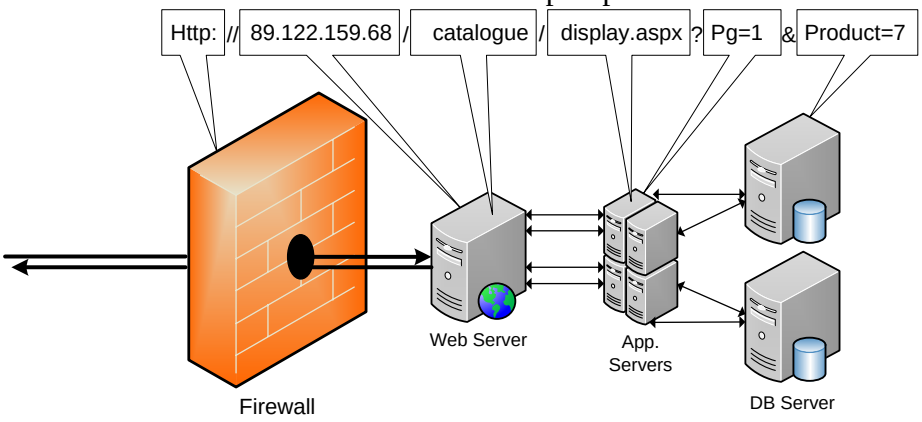


Figura 3.4 URL-ul “racheta de croazieră”

Efectele atacurilor folosind url-ul pot fi: dezvăluirea de căi din interiorul aplicației (paths), dezvăluirea codului sursă și a conținutului fișierelor arbitrare, dezvăluirea datelor din baza de date sau execuția arbitrară a unor comenzi.

Cele mai cunoscute și folosite zece atacuri web sunt:

- *URL misinterpretation* – serverul web nu reușește să parseze URL-ul corect astfel se poate ajunge la executarea unor comenzi nedorite.
- *Directory Browsing* „Browsing-ul directoarelor" – posibilitatea de a afla structura de directoare de pe serverul web. Pentru a evita acest atac configurația serverului trebuie să fie strictă.
- Exemplul dezvăluie structura de directoare a partiției C: *http://TARGET/scripts/..%255c..%%35cwinnt/system32/cmd.exe?/c+dir+c:*
- *Retrieving „non-web" files* – preluarea fișierelor non-web precum arhive, backup-uri, fișiere header, fișiere text, etc. Pentru a preveni acest atac trebuie să se dezactiveze trimiterea anumitor tipuri de fișiere de pe server și trebuie eliminată prezența fișierelor confidențiale din structura de directoare a aplicației.
- *Reverse Proxiying* – proxy-urile web sunt concepute pentru a permite accesul din interiorul unei rețele spre exterior dar ele pot funcționa și în direcția opusă. Pentru a nu fi posibil acest atac mapările url-urilor către serverele interne trebuie făcută cu grijă.
- *Java Decompilation* – codul Java poate fi decompilat destul de ușor. Acesta ar putea dezvălui informații sensibile precum parole, căi din aplicație, sau logica aplicației(generarea ID-urilor de sesiune, metode de criptare). Pentru reducerea riscurilor trebuie eliminate informațiile sensibile și fișierele care nu sunt necesare din *.jar-uri
- *Input Validation* – este cauza de bază a majorității atacurilor. Toate intrările ar trebui validate(tip, lungime, caractere speciale). Singura protecție față de acest atac sunt practicile bune de scriere a codului.
- *Source Code Disclosure* – posibilitatea de a primi fișiere de cod ale aplicației neparsate. Atacatorii pot dobândi codul sursa a aplicației web. Codul poate fi folosit pentru a descoperi „porți" de acces. Se poate evita prin folosirea practicilor de scriere de cod sigur.
- *SQL Qurey Poisoning(SQL injection)* – parametri din URL sau câmpurile de intrare ajung să fie folosiți în query-urile SQL. În funcție de scopul atacului se poate ajunge chiar și pâna la ștergerea completa a bazei de date. Singura metoda de a preveni acest atac este atenția la scrierea codului. Niciodată nu trebuie create query-uri din concatenare de string-uri nevalidate.
  Exemplu:
  http://target/login.asp?userid=bob%27%3b%20update%20logintable%20set%20passwd%3d%270wn3d%27%3b--%00
  Execută query-ul următor care schimbă parola utilizatorului ‚bob'
  ```
  SELECT preferences
  FROM logintable
  WHERE userid='bob'
  UPDATE logintable
  SET password='0wn3d'
  ```
- *Session Hijacking* – Protocolul HTTP nu își păstrează starea. Din acest motiv se folosesc diferite mecanisme pentru păstrarea sesiunii. Dacă se descoperă cheia sesiunii aceasta este compromisă. Pentru protecție trebuie urmărite ID-urile de sesiune de pe server, acestea trebuie marcate cu mărci de timp și adrese IP, ID-urile de sesiune trebuie generate criptografic, etc.
- *Buffer Overflows* – acest atac poate fi executat dacă nu se verifică lungimile intrărilor. Un overflow poate duce la blocarea aplicației (denial of service) și la

execuția comenzilor nedorite. Pentru a preveni acest atac trebuie validate intrările și trebuie făcute teste de buffer overflow.

3.1.6 Modele de realizare a comunicării - Serviciile web

Un serviciu web este descris de W3C ca un software proiectat pentru realizarea interacțiunii dintre două sisteme prin intermediul unei rețele. Serviciile web sunt foarte des API-uri care pot fi accesate printr-o rețea precum internetul și executate pe mașina remote care deține serviciul. Alte abordări cu funcționalitate asemănătoare cu a serviciilor web sunt:

- Common Object Request Broker Architecture(CORBA) – este un standard definit de OMG(Object Management Group) care permite aplicațiilor software scrise în diverse limbaje de programare și aflate pe mașini diferite să coopereze.
- Distributed Component Object Model(DCOM) – metoda propusa de Microsoft pentru intercomunicarea obiectelor distribuite pe mai multe computere dintr-o rețea.
- Java Remote Method Invocation(RMI) – metoda propusa de SUN pentru invocarea metodelor la distanță.

Serviciile web definite de W3C cuprind mai multe sisteme diferite dar în termeni comuni se referă la client și servere care comunică prin protocolul HTTP prin WEB. Aceste servicii se pot împarți în 2 categorii: serviciile web mari și serviciile web RESTful.

Prima categorie, „BIG webservices”, au la bază XML-uri care urmează standardul SOAP. În acest caz de obicei există o descriere a serviciilor care poate fi citită de restul computerelor, scrisă în WSDL. WSDL-ul nu este obligatoriu dar ajută la generarea automată de cod client în multe framework-uri SOAP Java și .Net.

A doua categorie “RESTful” începe să recâștige teren. Acestea se integrează mai bine cu protocolul HTTP și nu necesită SOAP sau WSDL.

Pentru a face un serviciu web accesibil consumatorilor acesta trebuie publicat. Pentru a putea consuma un serviciu consumatorii accesează interfața de serviciului. Această interfață descrie serviciul utilizând un limbaj standard WSDL (limbaj de descriere a serviciilor web). Opțional un provider de servicii își poate înregistra serviciul web într-un registru de servicii web,UDDI (Descriere, Descoperire și Integrare Universală). În Figura 3.5 este o reprezentare a funcționării serviciilor web, Brokerul de Servicii (UDDI) este opțional.

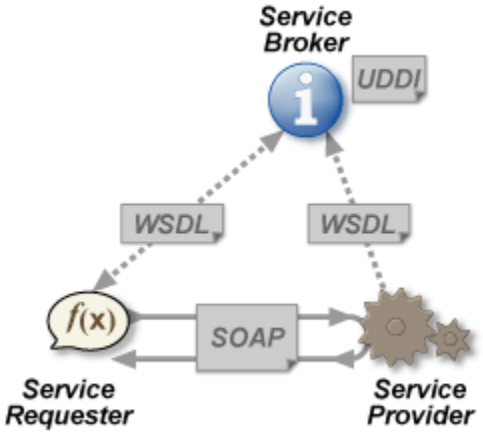


Figura 3.5 Modelul de funcționare a unui serviciu web

SOAP(protocol de acces simplu la obiecte) este un protocol pentru schimbul de date structurate prin intermediul serviciilor web în rețele de calculatoare. Acest protocol are

mesajele în format XML-ul și se bazează pe protocoalele nivelului aplicație (RPC și HTTP) pentru negocierea și transmiterea mesajelor.

Exemplu de cerere SOAP:

```
POST /InStock HTTP/1.1
Host: www.example.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soap:Envelope xmlns:soap=http://www.w3.org/2001/12/soap-envelope
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
      <soap:Body xmlns:m="http://www.example.org/stock">
            <m:GetStockPrice>
                  <m:StockName>IBM</m:StockName>
              </m:GetStockPrice>
      </soap:Body>
</soap:Envelope>
```

Exemplu de răspuns SOAP:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
      <soap:Body xmlns:m="http://www.example.org/stock">
             <m:GetStockPriceResponse>
                  <m:Price>34.5</m:Price>
            </m:GetStockPriceResponse>
      </soap:Body>
</soap:Envelope>
```

WSDL (serviciu de descriere a serviciilor web) este un limbaj care are la bază formatul XML și este folosit pentru descrierea serviciilor web. într-un WSDL este descrisă metoda de acces către serviciu și operațiile executate de acesta. Limbajul WSDL este folosit de către UDDI.

Exemplu:

```
<definitions name="HelloService"
   targetNamespace="http://www.examples.com/wsdl/HelloService.wsdl"
   xmlns="http://schemas.xmlsoap.org/wsdl/"
   xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
   xmlns:tns="http://www.examples.com/wsdl/HelloService.wsdl"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <message name="SayHelloRequest">
      <part name="firstName" type="xsd:string"/>
   </message>
   <message name="SayHelloResponse">
      <part name="greeting" type="xsd:string"/>
   </message>
...
   <soap:binding style="rpc"
      transport="http://schemas.xmlsoap.org/soap/http"/>
...
   <service name="Hello_Service">
      <documentation>WSDL File for HelloService</documentation>
      <port binding="tns:Hello_Binding" name="Hello_Port">
         <soap:address
```

```
                location="http://www.examples.com/SayHello/">
            </port>
        </service>
    </definitions>
```

Exemplul anterior reprezinta codul WSDL pentru un serviciu WEB de tip „Hello, World”. Acesta primește ca și parametru de intrare un string, căruia îi adaugă cuvântul „Hello” după care îl trimite înapoi ca și răspuns.

3.2 Sisteme de automatizare a clădirilor

Un sistem de automatizare a clădirilor (BAS – building automation system) este un exemplu de sistem distribuit. Nivelul de automatizare al unei clădiri este descris de sistemul de control. Acesta este un sistem computerizat, o rețea inteligență de dispozitive electronice, concepute să monitorizeze și să controleze partea mecanică și electrică a clădirii.

Sistemul de automatizare ține temperatura din clădire între limite specificate, furnizează lumina în funcție de programul prestabilit, monitorizează performanțele echipamentelor și anunță echipa de reparații care se ocupă de clădire de problemele existente. O clădire controlată de un sistem de automatizare este des referită ca și o clădire inteligentă.

3.2.1 Topologie

Majoritatea rețelelor de automatizare a clădirilor constau din două magistrale una primară și una secundară care interconectează controllere de nivel înalt (specializate pentru automatizarea clădirilor sau controlere programabile) cu controllere de nivel jos adică interfețe utilizator de intrare ieșire. O astfel de topologie se poate observa în Figura 3.6.

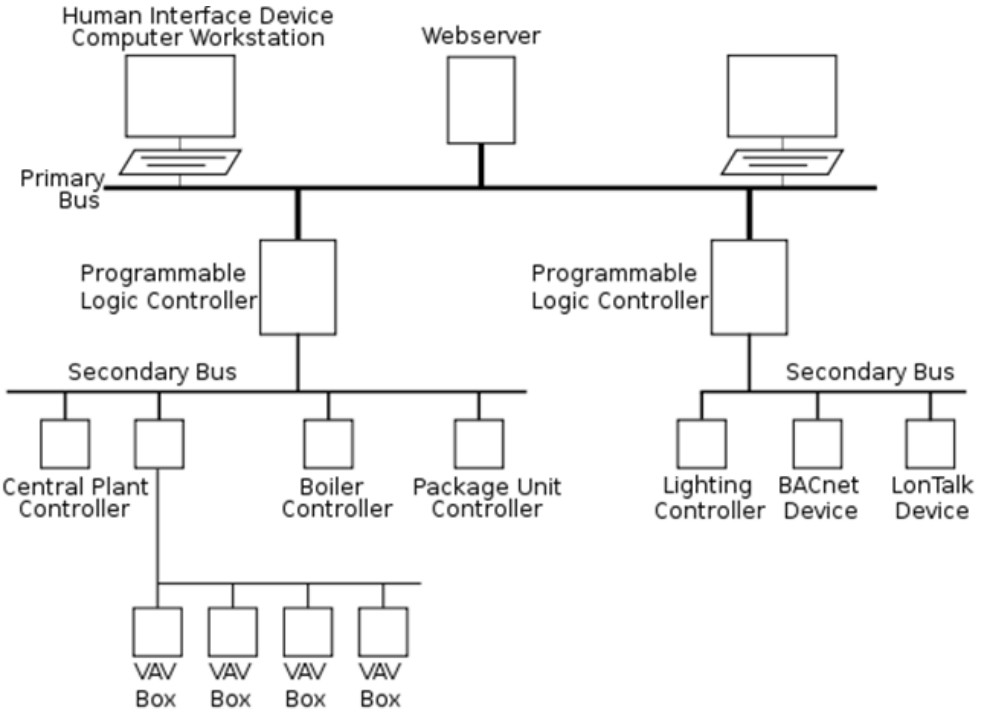


Figura 3.6 Tolopogie a componentelor de control într-o clădire automatizată[17]

În mod tradițional casele au cablajul în interiorul pereților. De obicei există cablaj pentru liniile de curent, telefonie, televiziune și sonerie. Există multiple standarde pentru diferite tipuri de transmisii. Pentru anumite sisteme de control este nevoie de cablaj separat, altele funcționează pe sistemele deja existente. În tabelul alăturat puteți observa standardele

de comunicații existente, mediul de transmisie, vitezele și distanțele pe care se pot folosi. Cele mai des întâlnite medii de transmisie pentru cele două bus-uri se găsesc în Tabel 3.1 [17]:

Tabel 3.1

Tehnologia	Mediu de transmisie	Viteza transmisie	Distanța maximă
Ethernet	Cablu torsadat(UTP)	10 Mbit/s – 1 Gbit/s	100m
	Fibra optică	1 Gbit/s – 10 Gbit/s	2km – 15km
HomePlug	Cablaj electric	14 Mbit/s - 200 Mbit/s	200m
Universal Powerline Association	Cablaj electric	200 Mbit/s	200m
ITU-T G.hn	Cablaj electric / linie telefonică / cablu coaxial	până la 1 Gbit/s	N/A
HomePNA	Linie telefonică	10 Mbit/s	300m
Wi-Fi/IEEE 802.11	Frecvența radio	11 Mbit/s – 248 Mbit/s	30m – 100m
FireWire/IEEE 1394	Cablu torsadat(UTP) / Optical fiber	400 Mbit/s – 3.2 Gbit/s	4.5m – 70m
Bluetooth	Frecventa radio	1 Mbit/s – 10 Mbit/s	10m – 100m
IRDA	Infraroșu	9600 bit/s – 4 Mbit/s	2m
LonWorks	Cablu torsadat(UTP)/Cablaj electic / Frecvența radio / Coaxial	1.70 kbit/s – 1.28 Mbit/s	1500m – 2700m
INSTEON	Cablaj electric + Wireless	1.2 kbit/s	1,000m+(Electric) 50m+ (Wireless)
X10	Electrical wiring	50 bit/s – 60 bit/s	
European Installation Bus / KNX	Cablu torsadat(UTP) / Cablaj electric / Frecvența radio / Infraroșu / Ethernet	1200 bit/s – 9600 bit/s	300 m – 1000 m
EHS	Cablu torsadat(UTP)/Cablaj electic	2.4 kbit/s – 48 kbit/s	
Batibus	Cablu torsadat(UTP)	4800 bit/s	200 m – 1500 m
Zigbee	Frecvența radio	20 kbit/s – 250 kbit/s	10 m – 75 m
Zwave	Frecvența radio	9.6 kbit/s – 40 kbit/s	1 m – 75 m
USB	Cablu torsadat(UTP)	12 Mbit/s – 480 Mbit/s	5 m

Majoritatea dispozitivelor controlabile existente funcționează doar în combinație cu dispozitivele de comandă ale companiei producătoare. Fiecare companie are controllerele pentru aplicații specifice. Unele sunt concepute pentru o singură funcție și pot funcționa cu un număr limitat de alte dispozitive, altele sunt mult mai flexibile. Intrările și ieșirile dispozitivelor sunt fie analogice fie digitale (binare).

Intrările analogice sunt folosite pentru citirea variabilelor de exemplu: temperatura, umiditate, presiune etc. O intrare digitală indică dacă un dispozitiv este pornit sau oprit.

Ieșirile analogice controlează viteza sau poziția unor dispozitive de exemplu actuatorul unei valve sau motoarele de control a poziției camerelor de supraveghere. Ieșirile digitale sunt folosite pentru deschiderea și închiderea releelor și a întrerupătoarelor. Un exemplu ar fi aprinderea luminii din parcare în momentul în care un senzor cu fotocelula arată că este întuneric afară.

3.2.2 Infrastructura

Controllerele sunt în general mici, circuite făcute special pentru scopul pe care îl îndeplinesc cu intrări și ieșiri. Aceste controllere variază în dimensiuni și funcționalitate în funcție de scopul pe care trebuie să îl îndeplinească în locuință.

Intrările permit controllerelor să citească temperatura, umiditatea, presiunea, curentul de aer și alți factori esențiali. Ieșirile le permit să trimită comenzi și semnale de control dispozitivelor „slave” și altor părți ale sistemului. Controllerele utilizate pentru automatizarea clădirilor pot fi grupate în 3 categorii:

- ***Controllere logice programabile (PLC)*** – furnizează răspunsul cel mai rapid dar costă de 2 până la 3 ori mai mult decât un System/Network controller. Acestea sunt folosite pentru zonele în care viteza răspunsului este critică.
- ***System/Network Controllers*** –îndeplinesc aceleași funcții ca și PLC-urile dar la viteze și costuri mai mici.
- ***Terminal unit Controllers*** –sunt folosite pentru controlul dispozitivelor simple cum ar fi o lampă, o priză, un ventilator, etc. În cazul acestor dispozitive se alege cel potrivit și se folosește, nu este nevoie de crearea de logică de control nouă.

Câțiva dintre producătorii de sisteme de control a clădirilor pe diferite domenii sunt: Automated Logic Corporation, Beckhoff Automation, Cisco Systems, Computrols Inc., Honeywell Home and Building Control, Invensys Building Systems, Johnson Controls Inc., Trend Control Systems Ltd., Schneider Electric, Siemens Building Technologies.

3.2.3 Protocoale și Standarde

Întrucât există o vastă varietate de obiecte și sisteme controlabile în interiorul unei clădiri în decursul anilor s-au stabilit o multitudine de protocoale și standarde pentru comunicația între acestea.

Câteva dintre cele mai importante standarde sunt:

- ***BACnet*** – este un protocol de comunicație pentru sistemele de automatizare și control al clădirilor. A fost creat special pentru a întâmpina nevoile aplicațiilor de control a încălzirii, ventilației, răcirii aerului, luminii, accesului în clădire, etc.
- ***ZigBee*** – este un protocol pentru rețele WPAN(wireless personal area networks). Protocolul definit de ZigBee se dorește a fi mai ieftin și mai ușor de implementat decât Bluetooth. Acesta poate fi folosit pentru controlul luminilor, a temperaturii, dispozitivelor multimedia, senzorilor de umiditate, senzorilor de fum și foc, etc.
- ***X10*** – este un standard internațional de comunicație prin intermediul rețelei electrice și este folosit pentru automatizarea locuințelor. X10 a fost prima dată introdus în 1975 de Pico Electronics fiind primul protocol de acest gen introdus și în același timp cel mai răspândit.

3.2.3.1 *Standardul X10*

Cablajul electric al clădirii este folosit pentru a trimite semnale digitale între dispozitive X10. Datele digitale sunt codate pe o transportoare de 120kHz care este transmisă în rafală în timpul nivelului 0 (perioada relativ silențioasă) al undei de curent alternativ de 50/60 Hz. Câte un bit este transmis la fiecare trecere pe la nivelul 0 al semnalului.

Datele digitale sunt formate dintr-o adresă și o comandă trimisă de la controller către un dispozitiv controlat. Controllerele mai avansate pot interoga dispozitivele pentru a afla în ce stare se afla. Răspunsul poate fi un simplu „pornit” / „oprit” sau date mai complexe cum ar fi nivelul de luminozitate sau temperatura senzorului.

Frecvența relativ înaltă a semnalului nu poate trece printr-un transformator sau de pe o fază pe alta în sistemele multifazate. Pentru sisteme multifazate un repetor X10 poate fi folosit. Pentru a nu interfera semnalele X10 dintr-o clădire cu semnalele din clădirea alăturată trebuie folosite filtre inductive la intrarea/ieșirea din locuință.

3.2.3.2 *Protocolul X10*

Indiferent dacă se folosesc liniile de alimentare electrica sau comunicarea radio, pachetele transmise folosind protocolul X10 sunt formate din:

Codul casei(4 biți)	Codul Unității(4 biți)	Codul Comenzii(4 biți)

Codurile caselor sunt litere de la A la P, reprezentări binare 0000 -> 1111. Codurile dispozitivelor sunt numere de la 1 la 16, reprezentările binare fiind 0000 -> 1111. Într-o singură locuință se pot folosi mai multe coduri de case astfel se pot controla pâna la 256 de dispozitive X10 (16 coduri de case × 16 coduri de unități).

Protocolul poate transmite comanda "selectează codul A3", urmată de comanda "pornește" ceea ce dă comanda de pornire dispozitivului A3. Multiple unități pot fi adresate în același timp. Astfel se pot trimite comenzi "selectează codul A3", "selectează codul A7", "selectează codul A13" după care se trimite comanda "oprește" ceea ce va opri toate dispozitivele selectate.

În Tabel 3.2 sunt înșirate comenzile protocolului X10 împreună cu codurile lor.

Tabel 3.2

Cod	Funcție	Descriere
0 0 0 0	Oprește tot	Oprește toate dispozitivele cu codul de casă indicat
0 0 0 1	Pornește luminile	Pornește toate luminile
0 0 1 0	Pornește	Pornește un dispozitiv
0 0 1 1	Oprește	Oprește un dispozitiv
0 1 0 0	Intunecă	Reduce intensitatea luminii
0 1 0 1	Luminează	Crește intensitatea luminii
0 1 1 1	Extended Code	Cod de extensie
1 0 0 0	Trimite cerere	Cere răspuns de la aparatele cu codul de casă indicat
1 0 0 1	Trimite raspuns	Răspuns la comanda precedentă
1 0 1 x	Presetare luminozitate	Permite alegerea a 2 nivele de luminozitate predefinite
1 1 0 1	Status Pornit	Răspuns la cererea de status indicând dispozitiv pornit
1 1 1 0	Status Oprit	Răspuns la cererea de status indicând dispozitiv oprit
1 1 1 1	Cerere Status	Cerere de status

Comenzile de oprire a tuturor unităților sau de pornire a tuturor luminilor primesc ca parametru codul casei. Astfel dacă într-o clădire se vor folosi mai multe coduri de casă („house code"), acestea o vor impărți în zone pe care aceste comenzi acționează [16].

3.2.3.3 *Dezavantaje X10*

O problemă majoră a sistemului X10 o reprezintă atenuarea excesivă care apare între cele două linii active ale sistemului tipic de alimentare cu energie electrică. Consumatori mari de curent pot duce la imposibilitatea temporară a transmisiilor X10. Aceasta problemă poate fi îndepărtată prin instalarea unui condensator între firele de fază ca și o cale pentru semnalele X10.

O siguranță poate atenua semnalele X10. Semnalele X10 ce trec prin siguranțe s-ar putea să nu fie destul de puternice pentru a controla dispozitive aflate de partea cealaltă.

De asemenea anumite surse de alimentare electrică folosite în echipamentele electronice moderne (televizoare, computere, receivere) „consumă" semnalele X10 oferind o cale cu impedanța scăzută semnalelor de frecvență înaltă. Există filtre care se pot folosi înaintea acestor echipamente.

Semnalele X10 pot fi transmise doar pe rând. Dacă două semnale X10 sunt trimise deodată acestea se anulează sau dau comenzi neașteptate.

Capitolul 4 Specificaţiile şi arhitectura sistemului

4.1 Sistem de control pentru inteligența ambientală

Un sistem de control pentru inteligența ambientală este un sistem distribuit format din multiple module care au ca misiune furnizarea accesului la locuință a utilizatorului de la distanță. Sistemul trebuie să poată îndeplini sarcini și în mod autonom în urma preprogramării acestora de către proprietar (utilizator al sistemului). Utilizatorul aflat la distanță va putea supraveghea locuința prin intermediul camerelor de supraveghere video, a microfoanelor și a senzorilor de temperatură și va putea controla locuința putând opri sau porni lumini, electrocasnice sau robinete de apă.

Un astfel de sistem oferă controlul complet asupra locuinței pentru utilizatorul aflat la distanță. Controlul trebuie să poată fi făcut ușor prin intermediul unei interfețe prietenoase și un sistem cu reacție rapidă. Deoarece un astfel de sistem conectează sistemele din locuință la o rețea globală (Internetul), sistemul trebuie să fie foarte bine securizat deoarece mai devreme sau mai târziu va fi supus atacurilor.

4.2 Schema bloc a sistemului

Sistemul de control de la distanță a locuinței este distribuit în 3 locații(Figura 4.1):

- locația principală
- locația remote
- locația clientului

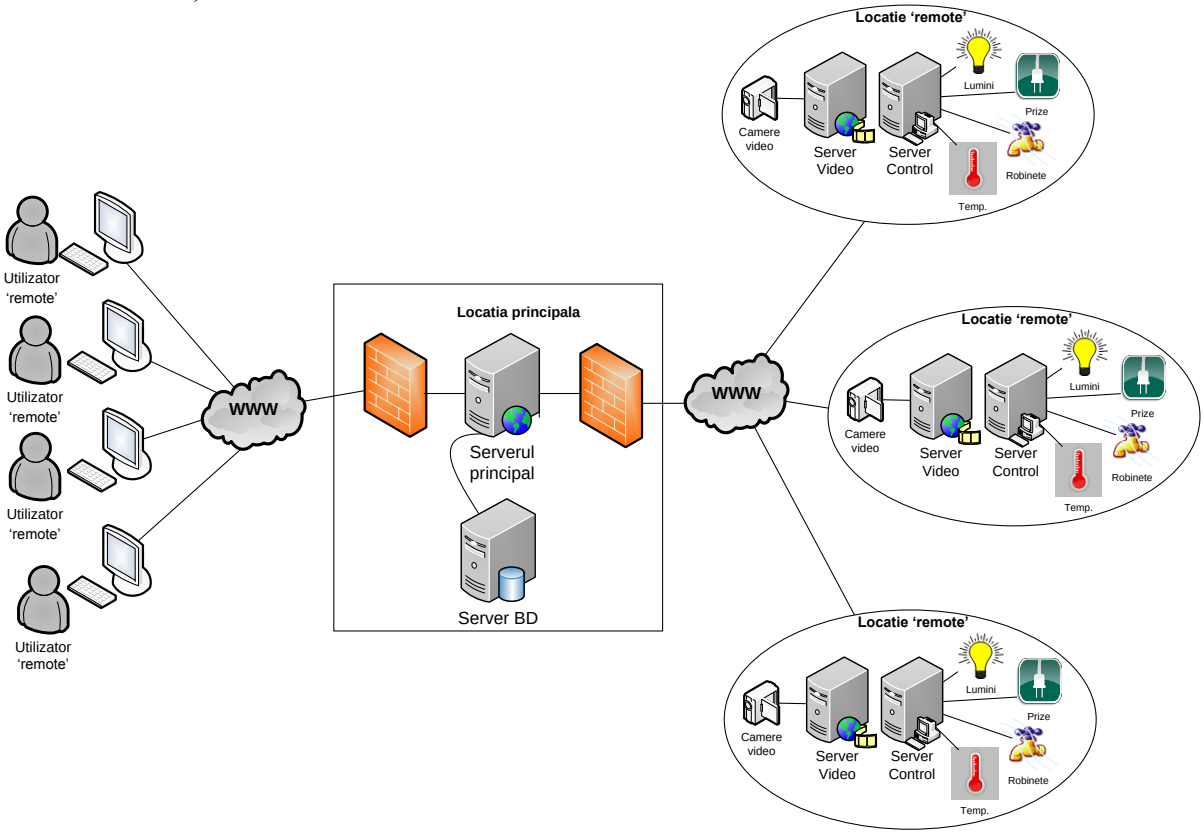


Figura 4.1 Schema bloc a sistemului

Locația remote reprezintă locuința utilizatorului. În sistem vor fi atâtea locații remote cate locuințe vor fi contactate. În locația remote se vor afla serverul de supraveghere video și serverul de control care se pot situa în două sisteme separate sau într-unul singur.

Serverul de supraveghere este conectat la camerele de supraveghere video. Acesta înregistrează local imaginile relevante și va da acces la imagini în direct clientului aflat la distanță. Serverul de control este conectat la sistemele care controlează luminile, prizele, temperatura și robinetele din locuință.

Locația principală este locația în care sunt stocate datele despre toate locațiile remote. Sistemul va avea o singură bază de date care va fi situată aici. Aplicația web la care se conectează clienții va fi situată tot în această locație.

Locația clientului este o locație în continuă schimbare. Clientul se poate conecta la sistem de oriunde de pe glob cu ajutorul unui browser de web.

Toate locațiile sunt conectate la Internet, acesta fiind mediul lor de comunicație.

4.3 Cerințele sistemului

4.3.1 Funcţionale

Aplicaţia implementată trebuie să răspundă următoarelor cerinţe:

a. Configurarea locațiilor conectate la sistem

În sistem trebuie să existe mai multe tipuri de utilizatori: utilizatori finali cu diferite drepturi și administratori. Managementul locațiilor este sarcina administratorilor. Prin managementul locațiilor se înțelege adăugarea / editarea / ștergerea clădirilor, adăugarea / editarea / ștergerea zonelor din interiorul clădirilor, adăugarea/editarea/ștergerea diferitelor controale existente în clădiri.

b. Managementul conturilor de acces

Administratorul va adăuga utilizatori noi în sistem. La adăugare, parola de acces va fi generată automat și trimisă proprietarului contului. Astfel clientul va fi singurul care își va cunoaște parola și care va putea accesa locuința. Administratorul va mapa un utilizator la locuința sa și va adăuga cheile de acces pentru sistemul de supraveghere și control.

c. Posibilitatea de conectare la multiple locații din aceeași interfață

Un utilizator va putea fi proprietarul a multiple clădiri în care se instalează sistemul. Acesta va trebui să se poată conecta la oricare din aceste clădiri cu ajutorul aceleiași interfețe și folosind același cont de acces.

d. Vizualizarea imaginilor camerelor de supraveghere

În sistem vor exista 4 roluri dintre care 3 vor fi de utilizatori cu diferite drepturi. Camerele de supraveghere video trebuie să poată fi vizualizate de către utilizatorii cu drepturi depline și de către utilizatorii pasivi. Aceștia trebuie să poată urmări imaginea de la 4 camere video simultan. Utilizatorii vor putea controla calitatea și dimensiunile imaginilor vizualizate. Vizualizarea camerelor video se va realiza prin intermediul interfeței web a aplicației.

e. Controlul dispozitivelor conectate la instalația electrică

Dispozitivele de control din sistem pentru instalația electrică vor putea fi controlate de utilizatorii cu drepturi depline și utilizatorii activi. Aceștia vor putea porni sau opri dispozitive din instalația electrică cum sunt luminile, alimentarea prizelor și siguranțele prin intermediul interfeței web a aplicației.

f. Controlul electrovalvelor

Dispozitivele de control din sistem pentru rețeaua de apă vor putea fi controlate de utilizatorii cu drepturi depline și utilizatorii activi. Aceștia vor putea porni sau opri robinetele din instalația de apă a locuinței prin intermediul interfeței web a aplicației.

g. Verificarea temperaturii

Verificarea temperaturii de la senzorii de temperatură din clădire se va face prin intermediul interfeței web. Doar utilizatorii cu drepturi depline și cei activi vor avea dreptul de a vizualiza temperaturile.

h. Programe automate

Dispozitivele controlabile din clădire(lumini, prize, siguranțe, robinete) vor putea fi programate pentru a se porni sau opri la anumite ore. Programelor automate li se va putea seta un interval de repetare care variază între o zi și o săptămână cu increment de o zi. Utilizatorii cu drept de control vor putea adauga sau șterge astfel de programe. Tot ei vor putea activa sau dezactiva programele existente.

i. Managementul parolelor

Parolele utilizatorilor nu trebuie să fie știute de nimeni altcineva, nici măcar de administratorul care creează contul. Astfel, pentru un cont nou creat administratorul nu va seta o parolă ci aceasta va fi autogenerată și trimisă clientului. Atât utilizatorii cât și administratorii vor putea să își schimbe parolele prin intermediul interfeței.

4.3.2 Non funcționale

a. Accesibilitate

Aplicația va trebui să fie accesibilă din orice locație cu efort minim și cu cunoștințe minime legate de tehnologie. Pentru ca acest lucru să fie posibil se va dezvolta o interfață web prietenoasă pentru aplicație.

b. Securitate

Sistemul va trebui să aibă un nivel de securitate ridicat deoarece controlul locuințelor poate fi făcut de la distanță de oricine deține codurile de acces sau reușește să pătrundă fraudulos în aplicație. Pentru securitatea conturilor parolele de acces vor fi auto generate și știute doar de proprietarul contului. Întrucât clădirile sunt conectate la Internet prin intermediul sistemului acestea sunt expuse atacurilor. Pentru a evita atacurile online sau furtul de conturi de acces se va avea în vedere în timpul dezvoltării aplicației acest risc. Toate transferurile de date se vor face printr-un canal securizat utilizând protocolul HTTPS și nu HTTP.

c. Scalabilitatea

Sistemul va trebui să fie ușor scalabil pe orizontală deoarece noi locuințe se vor adăuga zilnic. Pentru a îndeplini aceasta cerință, pe lângă locațiile „remote” va exista și o locație centrală care va găzdui baza de date a sistemului. În această bază de date vor fi ținute toate conturile de acces și detaliile locațiilor. Pentru adăugarea unei noi locații nu trebuie decât introdusă o noua înregistrare în baza de date.

În locațiile remote se vor putea adăuga pană la 256 de elemente electrice controlabile, 7 robinete controlabile și 5 senzori de temperatură.

d. Disponibilitate 24/7

Sistemul va trebui să fie funcțional 24 ore/zi, 7 zile pe săptămână. Pentru acest lucru toate conexiunile la Internet trebuie să fie stabile. În locația centrală se vor utiliza conexiuni redundante. Toate serverele și dispozitivele de control vor fi conectate la UPS-uri de mari dimensiuni.

e. Timp de raspuns

Timpii de raspuns a aplicației vor trebuie să fie scăzuți. Conexiunea la Internet a sistemului trebuie să aibă lățime destul de mare pentru a putea face față schimbului de date, în special când se vizualizează imagini de la 4 camere de supraveghere simultan.

4.4 Scenarii de utilizare

Pe baza funcţionalităţilor enumerate anterior şi a descrierii aplicaţiei, s-au evidenţiat două tipuri de utilizatori. Utilizatorul trebuie să fie autentificat pentru a accesa funcţionalităţile asociate cu rolul lui. Autentificarea se face pe baza numelui de utilizator (username) şi a parolei.

După autentificare fiecare rol are acces la anumite zone ale aplicației și are anumite drepturi. Funcționalitățile comune tuturor rolurilor sunt: autentificare, log out și schimbare a parolei.

Funcţionalităţile specifice rolurilor sunt următoarele:

- **Administrator**:
  - adăugarea/editarea/ștergerea de utilizatori și administratori (parola nu poate fi schimbată de către administrator)
  - adăugarea/editarea/ștergerea locuințelor
  - adăugarea/editarea/ștergerea mapărilor dintre locuințe și utilizatori
  - adăugarea/editarea/ștergerea locațiilor din locuințe
  - adăugarea/editarea/ștergerea controalelor X10 din locuințe (lumini, prize)
  - adăugarea/editarea/ștergerea celorlalte controale din locuințe (robinete, temperatura)
  -

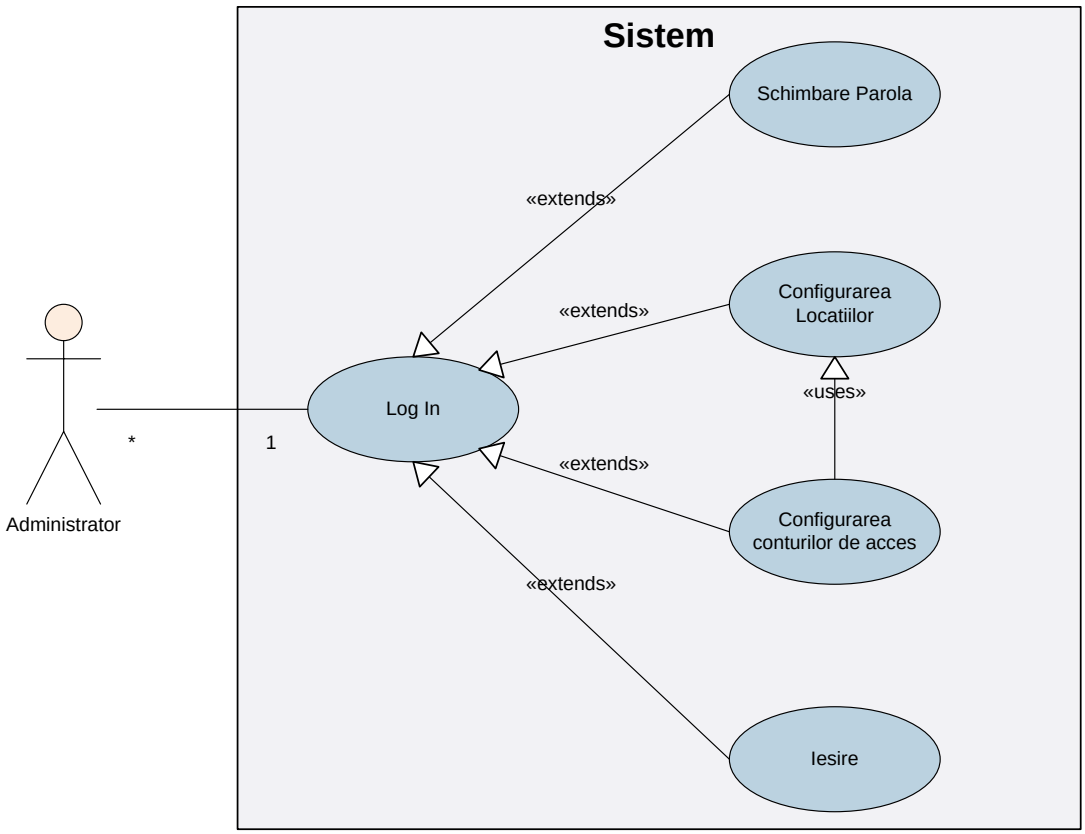


Figura 4.2 Diagrama Use Case pentru rolul de administrator

- **Utilizator cu drepturi depline**:
  - urmărirea camerelor video din locație și funcții specifice (start, stop, zoom, schimbarea calității imagini, etc.)
  - pornirea/oprirea luminilor din clădire
  - pornirea/oprirea aparatelor conectate la prize
  - pornirea/oprirea siguranțelor
  - pornirea/oprirea robinetelor
  - pornirea simultană a tuturor luminilor
  - oprirea simultană a tuturor aparatelor electrice și a luminilor
  - verificarea temperaturilor citite de la senzorii de temperatură
  - adăugarea programelor automate de oprire și pornire a controalelor la anumite ore și cu un anumit interval de repetare
  - oprirea și ștergerea programelor automate

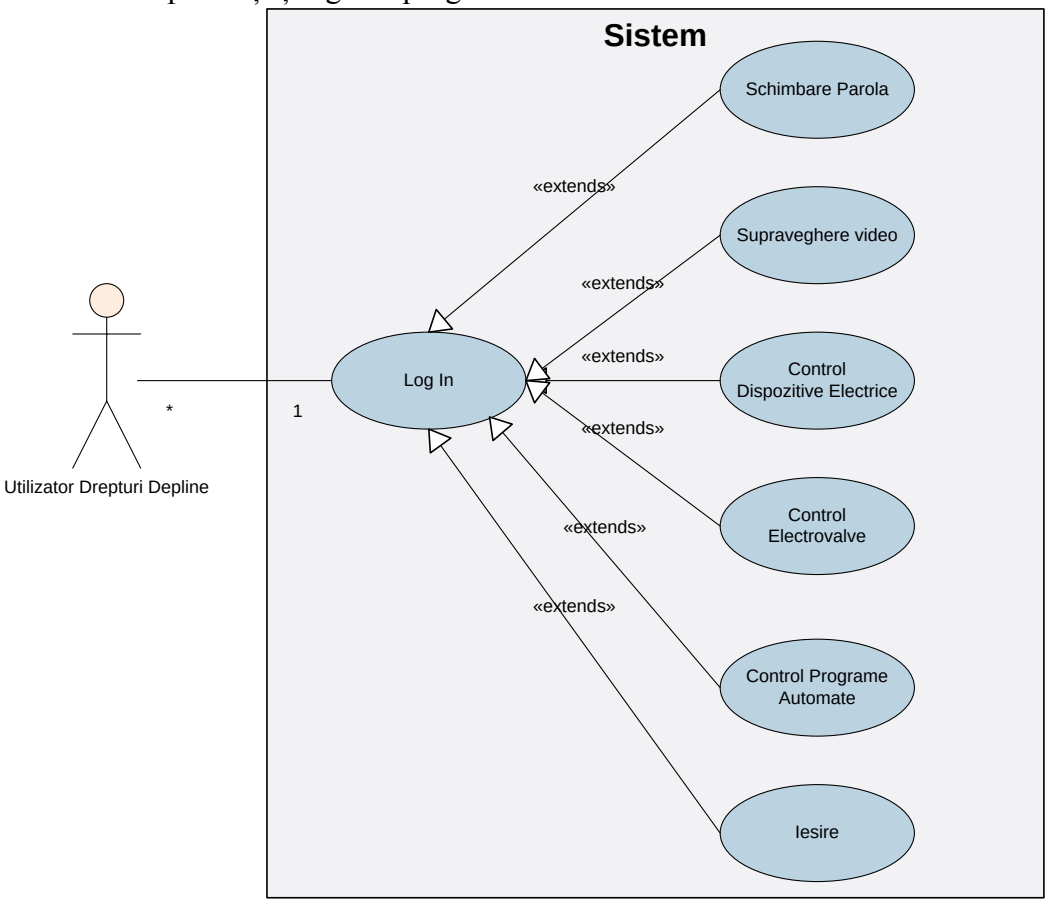


Figura 4.3 Diagrama Use Case pentru rolul de utilizator cu drepturi depline

- **Utilizator pasiv** (poate vizualiza camerele video dar nu poate controla locuință):
  - urmărirea camerelor video din locație și funcții specifice (start, stop, zoom, schimbarea calității imagini, etc.)
- **Utilizator activ** (poate prelua controlul locuinței dar nu poate vizualiza camerele video)**:**
  - pornirea/oprirea luminilor din clădire
  - pornirea/oprirea aparatelor conectate la prize
  - pornirea/oprirea siguranțelor
  - pornirea/oprirea robinetelor
  - pornirea simultană a tuturor luminilor
  - oprirea simultană a tuturor aparatelor electrice și a luminilor

- verificarea temperaturilor citite de la senzorii de temperatură
- adăugarea programelor automate de oprire și pornire a controalelor la anumite ore și cu un anumit interval de repetare
- oprirea și ștergerea programelor automate

Din specificaţiile anterioare se deduc cazurile de utilizare care definesc scenariile de bază ale sistemului. Scopul principal al creării cazurilor de utilizare este de a identifica cursul principal al aplicației și nu de a defini toate combinațiile posibile de evenimente din sistem. În Figura 4.2 este prezentată diagrama use case principală pentru un administrator iar în Figura 4.3 este prezetată diagrama use case pentru un utilizator cu drepturi depline. Pentru sistemul propus s-au identificat următoarele cazuri de utilizare:

Caz de utilizare nr.1: Log În

Descriere: Autentificarea utilizatorului, după care în funcţie de rolul său este redirecționat la o anumită zonă a aplicației și poate îndeplinii funcții specifice.

Actori primari: Administrator, Utilizator cu funcții depline, Utilizator activ, Utilizator pasiv

Precondiţie: sistemul trebuie să fie pornit și un browser de web IE8 deschis la pagina https://www.smart-homes.ro (pagina aplicației)

Postcondiţie: Utilizatorul este autentificat şi în funcţie de rolul îndeplinit în sistem are acces la diferite funcţionalităţi.

Scenariul de succes:

1. Pagina de autentificare este afișată: se cer numele de utilizator și parola.
2. Sistemul verifică datele furnizate și redirecționează utilizatorul către zona aplicației care se potrivește rolului lui.

Extensie:

*a. Din motive tehnice browser-ul nu reușește să deschidă aplicația .

1 Apare pagina de avertizare cu textul „The Page Cannot Be Displayed”.

2a. Datele de autentificare sunt incorecte.

1 Un mesaj de eroare care avertizează utilizatorul este afișat.

Caz de utilizare nr.2: Configurarea locațiilor conectate la sistem

Descriere: Adăugarea unei noi locații și zone și controale pentru aceasta de către un administrator

Actori primari: Administrator

Precondiţie: Utilizatorul este autentificat ca administrator

Postcondiţie: O nouă casă, locații și controale pentru aceasta au fost adăugate.

Scenariul de succes:

1. Administratorul navighează la pagina de adăugare a locațiilor și deschide tab-ul de adăugare.
2. Administratorul completează datele noii locații și apasă butonul de adăugare „Add”.
3. Noua locație este adăugată și apare în tabelul cu locații.
4. Administratorul apasă butonul de editare a detaliilor locației.
5. Pagina cu detaliile locuinței este deschisă.
6. Administratorul introduce numele zonei și apasă butonul add.
7. Noua zonă apare în tabela de zone.
8. Administratorul repetă pașii 6-7 până când termină de introdus toate zonele.
9. Administratorul deschide tab-ul de adăugare de controale X10.

10. Completează datele controlului și apasă butonul de adăugare „Add”.
11. Noul control apare în tabela de controale X10.
12. Administratorul repetă pașii 10-11 până când termină de introdus toate controalele X10.
13. Administratorul deschide tab-ul de adăugare a celorlalte controale X10.
14. Completează datele controlului și apasă butonul de adăugare „Add”.
15. Noul control apare în tabela de alte controale.
16. Administratorul repetă pașii 14-15 până când termină de introdus toate controalele X10.
17. Administratorul apasă butonul de revenire la pagina de management a locațiilor.
18. Pagina de management a locațiilor se deschide.

Extensie:

3a. Utilizatorul nu a completat toate câmpurile obligatorii.
1. Un mesaj de avertizare este afișat și se revine la pasul 2.

5a. Din motive tehnice browser-ul nu reușește să deschidă pagina.
1. Apare pagina de avertizare cu textul „The Page Cannot Be Displayed”.

7a. Utilizatorul nu a completat toate câmpurile obligatorii.
1. Un mesaj de avertizare este afișat și se revine la pasul 6.

11a. Utilizatorul nu a completat toate câmpurile obligatorii.
1. Un mesaj de avertizare este afișat și se revine la pasul 10.

15a. Utilizatorul nu a completat toate câmpurile obligatorii.
1. Un mesaj de avertizare este afișat și se revine la pasul 14.

18a. Din motive tehnice browser-ul nu reușește să deschidă pagina.
1. Apare pagina de avertizare cu textul „The Page Cannot Be Displayed”.

Caz de utilizare nr.3: Configurarea conturilor de acces

Descriere: Adăugarea unui nou utilizator și maparea acestuia la o locație
Actori primari: Administrator
Precondiţie: Utilizatorul este autentificat ca administrator
Postcondiţie: Un nou utilizator a fost adăugat și are acces la cel puțin o locație
Scenariul de succes:

1. Administratorul deschide tab-ul de adăugare de noi utilizatori.
2. Completează datele utilizatorului și apasă butonul de adăugare.
3. Noul utilizator apare în tabelul de utilizatori și utilizatorul primește un e-mail cu parola.
4. Se deschide tab-ul de mapări.
5. Se selectează utilizatorul pentru care se face maparea, după care se selectează locația și se introduc cheile de acces la serverele video și de control după care administratorul apasă butonul add.
6. Noua mapare apare în tabela de mapări.
7. Se repeta pașii 5-6 pentru a mapa un utilizator la mai multe locații.

Extensie:

3a. Utilizatorul nu a completat toate câmpurile obligatorii, a introdus același nume de utilizator de două ori sau a introdus un e-mail invalid.
1. Un mesaj de avertizare este afișat și se revine la pasul 2.

3b. Trimiterea e-mail-ului a eșuat.

1. Administratorul trebuie să șteargă contul apăsând butonul de ștergere.
2. Administratorul trebuie să reia pasul 2.

5a. Locația la care se va face maparea utilizatorului nu a fost încă adăugată.
1. Administratorul executa pașii 1-18 al UC2.
2. Administratorul selectează pagina de management a utilizatorilor.
3. Se continuă cu pasul 4.

6a. Datele introduse sunt incomplete sau invalide.
1. Un mesaj de avertizare este afișat și se revine la pasul 5.

Caz de utilizare nr.4: Vizualizarea imaginilor camerelor de supraveghere

Descriere: Vizualizarea imaginilor de supraveghere de către un utilizator care are acest drept și schimbarea calității imaginii

Actori primari: Utilizator

Precondiţie: Utilizatorul este autentificat ca utilizator nu ca administrator

Postcondiţie: Utilizatorul vizualizează imagini de la camerele de supraveghere la rezoluție înaltă

Scenariul de succes:
1. Utilizatorul deschide pagina de supraveghere(„Surveillance”).
2. Sistemul deschide până la maxim 4 ferestre de supraveghere.
3. Utilizatorul apasă butonul play pentru fereastra pe care dorește să o deschidă.
4. Se pornește streaming-ul în fereastra respectivă.
5. Pentru a porni și alte camere video utilizatorul repeta pașii 3-4.
6. Utilizatorul apasă butonul de schimbare a rezoluției și selectează rezoluția 640*480.
7. Imaginea devine mai clară.

Extensie:

2a. Din motive tehnice browser-ul nu reușește să deschidă pagina.
1. Apare pagina de avertizare cu textul „The Page Cannot Be Displayed”.

2b. Utilizatorul nu are nici o camera video în locație.
1. Se va afișa fereastra goală.

4a. Din motive tehnice conexiunea cu serverul video nu reușește.
1. Va apărea un mesaj de eroare de conexiune „Connection error”.

4b. Utilizatorul nu are dreptul de a vizualiza camerele video sau codurile de acces au fost schimbate în locația remote.
1. Va apărea un mesaj de eroare care anunța greșeala codurilor de acces.

Caz de utilizare nr.5: Controlul dispozitivelor conectate la instalația electrică

Descriere: Pornirea/Oprirea unui dispozitiv conectat la instalația electrică

Actori primari: Utilizator

Precondiţie: Utilizatorul este autentificat ca utilizator nu ca administrator

Postcondiţie: Dispozitivul și-a schimbat starea de funcționare își schimbă starea de funcționare.

Scenariul de succes:
1. Utilizatorul deschide pagina de dispozitive („Appliances”).
2. În funcție de elementele care doresc a fi controlate (lumini sau prize) va deschide tab-ul corespunzător „lights” sau „plugs”.

3. Se afișează un tabel cu toate dispozitivele de tipul selectat.
4. Utilizatorul apasă butonul de pornire sau de oprire în funcție de acțiunea pe care vrea să o execute.
5. Becul/aparatul electrocasnic se pornește sau se oprește în funcție de operația care a fost selectată.

Extensie:

2a. Din motive tehnice browser-ul nu reușește să deschidă pagina
1. Apare pagina de avertizare cu textul „The Page Cannot Be Displayed”

2b. Utilizatorul nu are nici un dispozitiv de control instalat pe instalația electrică video în locație
1. Se va afișa fereastra cu tab-urile goale

4a. Din motive tehnice conexiunea cu serverul video nu reușește
1. Comanda nu va ajunge la clădirea remote și va apărea un mesaj de eroare

4b. Utilizatorul nu are dreptul de a controla dispozitivele electrice
1. Va apărea un mesaj de eroare care anunța greșirea codurilor de acces.

Caz de utilizare nr.6: Controlul electrovalvelor

Descriere: Pornirea/oprirea unei electrovalve
Actori primari: Utilizator
Precondiţie: Utilizatorul este autentificat ca utilizator nu ca administrator
Postcondiţie: Electrovalva și-a schimbat starea de funcționare.
Scenariul de succes:

1. Utilizatorul deschide pagina de dispozitive („Appliances”).
2. Deschide tab-ul corespunzător electrovalvele („Taps”)
3. Se afișează un tabel cu toate electrovalvele conectate la sistem.
4. Utilizatorul apasă butonul de pornire sau de oprire în funcție de acțiunea pe care vrea să o execute
5. Becul/aparatul electrocasnic se pornește sau se oprește în funcție de operația care a fost selectată.

Extensie:

2a. Din motive tehnice browser-ul nu reușește să deschidă pagina
1. Apare pagina de avertizare cu textul „The Page Cannot Be Displayed”

2b. Utilizatorul nu are nici un dispozitiv de control instalat pe instalația electrică video în locație
1. Se va afișa fereastra cu tab-urile goale

4a. Din motive tehnice conexiunea cu serverul video nu reușește
1. Comanda nu va ajunge la clădirea remote și va apărea un mesaj de eroare

4b. Utilizatorul nu are dreptul de a controla dispozitivele electrice
1. Va apărea un mesaj de eroare care anunță greșirea codurilor de acces.

Caz de utilizare nr.7: Adăugarea/execuția program automat pentru aprinderea unei lumini

Descriere: Adăugarea
Actori primari: Utilizator, Aplicația WorkerStarter
Precondiţie: Utilizatorul este autentificat ca utilizator nu ca administrator

Postcondiţie: Lumina este aprinsă
Scenariul de succes:

1. Utilizatorul deschide pagina de dispozitive („Scheduling”).
2. Deschide tab-ul de adăugare a unui program.
3. Completează datele necesare pentru programul dorit și apasă butonul add.
4. Noul program va apărea în tabela de programe.
5. Aplicația WorkerStarter inițiază câte o citire din baza de date în fiecare minut.
6. Programul adăugat este găsit de către procesul inițiat de WorkerStarter.
7. Sistemul din locația centrală reîmprospătează baza de date setând programul ca executat sau în cazul în care este un program repetitiv setând timpul următoarei execuții.
8. Aplicația centrală se autentifică la serverul de control din locația „remote”.
9. Modulul central dă comanda de aprindere a luminii.
10. Lumina se aprinde în locația „remote”.

Extensie:

1a. Din motive tehnice browser-ul nu reușește să deschidă pagina.
    1. Apare pagina de avertizare cu textul „The Page Cannot Be Displayed”.

4a. Utilizatorul nu a completat toate câmpurile obligatorii sau a introdus date invalide.
    1. Un mesaj de avertizare este afișat și se revine la pasul 3.

6a. Din motive tehnice programul nu este găsit la momentul potrivit.
    1. Acesta nu va mai fi executat.

8a. Din motive tehnice serverul central nu poate stabili conexiunea cu serverul remote.
    1. Programul nu va mai fi executat.

Caz de utilizare nr.8: Modificarea parolei de acces pentru un utilizator
Descriere: Adăugarea
Actori primari: Utilizator
Precondiţie: Utilizatorul este autentificat ca utilizator nu ca administrator
Postcondiţie: Parola a fost schimată
Scenariul de succes:

1. Utilizatorul deschide pagina status.
2. Apasă butonul change passowrd.
3. Se deschide pagina de modificare a parolei.
4. Utilizatorul introduce vechea parolă, noua parolă și confirmarea noii parole după care apasă butonul „Schimbă”.
5. Sistemul schimbă parola în baza de date (recripteaza ID-ul utilizatorului cu noua parolă).

Extensie:

1a. Din motive tehnice browser-ul nu reușește să deschidă pagina.
    1. Apare pagina de avertizare cu textul „The Page Cannot Be Displayed”.

5a. Utilizatorul nu a completat toate câmpurile obligatorii, a introdus parola de .confirmare greșită sau parola veche este invalidă
    1. Un mesaj de avertizare este afișat și se revine la pasul 4.

4.5 Arhitectura generală

Arhitectura generala a sistemului este prezentată în Figura 4.4. Sistemul este compus din mai multe module distribuite în locații diferite. Modulele de control sunt situate în locațiile „remote” iar modulul de aplicație (web), comandă și date se află în locația centrală.

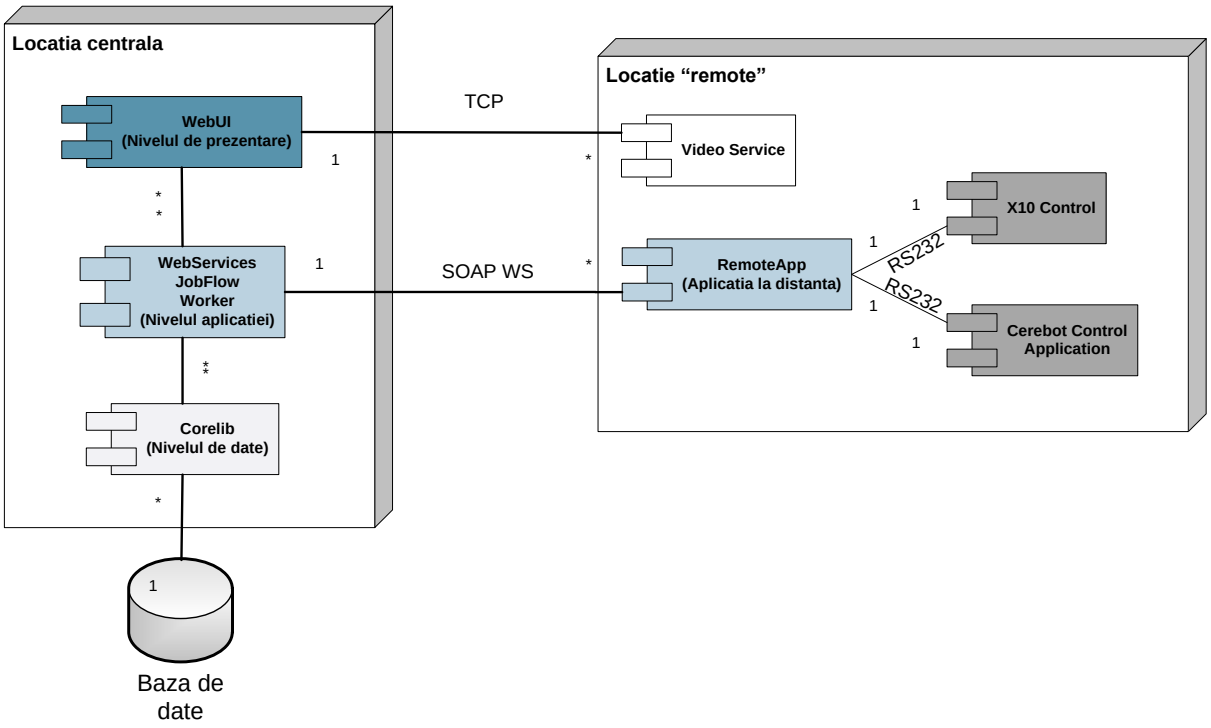


Figura 4.4 Arhitectura sistemului

În fazele incipiente ale proiectului am gândit mai multe arhitecturi posibile. După compararea avantajelor și dezavantajelor lor am ajuns la concluzia că arhitectura aleasă este cea mai sigură, ușor de întreținut și eficientă ca și costuri.

În următoarele rânduri voi prezenta trei variante de arhitecturi posibile pentru sistem.

Varianta I – O aplicație client instalată pe computerul de pe care se face accesul + câte o aplicație în fiecare locație „remote” și câte o bază de date în fiecare locație „remote”

Pentru a accesa și controla locația „remote” trebuie instalată o aplicație client pe computerul de pe care se face accesul. Acest lucru reprezintă un dezavantaj major deoarece fiecare instalare ia timp, necesită cunoștințe în domeniu și utilizatorul trebuie să dețină drepturi de administrator pe acel computer.

Varianta II – Câte o aplicație și câte o baza de date în fiecare locație „remote”

În această variantă în fiecare locație care se dorește a fi controlată de la distanță trebuie instalată o aplicație întreagă ceea ce implică existența bazei de date în locația respectivă. Utilizatorul se va conecta de la distanță la locația respectivă utilizând un browser și adresa IP a locației. În Figura 4.5 se poate observa un nod din configurația unui astfel de sistem.

Singura situație în care un asemenea sistem ar putea fi avantajos ar fi cazul în care se dorește utilizarea lui într-o singură locație (nu 1 locație/utilizator ci 1 locație/sistem). În acest caz costurile unei locații suplimentare (locația centrala) nu ar fi justificate întrucât va exista o singură bază de date și o singură adresă care trebuie reținută pentru efectuarea conectării.

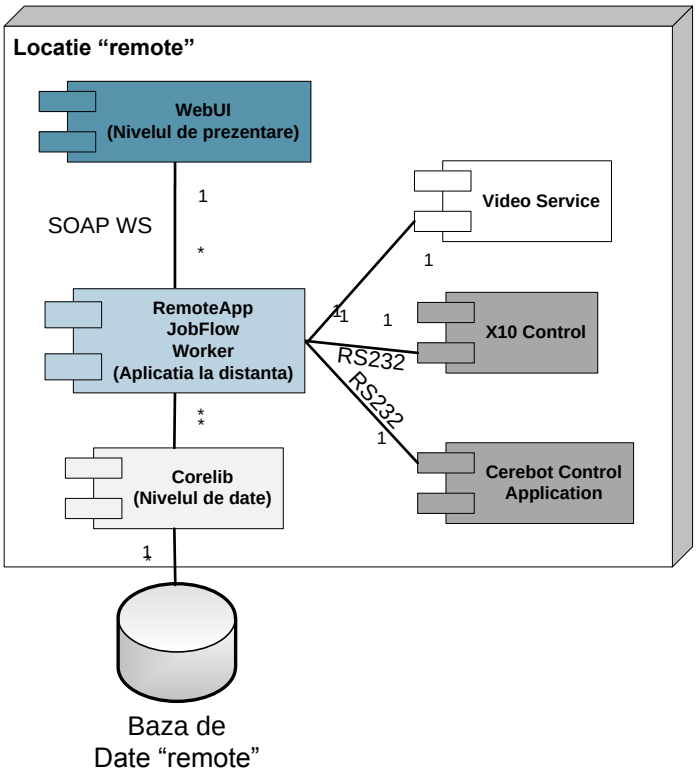


Figura 4.5 Arhitectura posibilă Varianta II

Dacă ţinta este un sistem comercial care să fie folosit în multiple locaţii, o astfel de configuraţie ar prezenta multiple dezavantaje:

- greu de întreţinut – pentru orice modificare trebuie reinstalat sistemul în toate locaţiile unde a fost instalat
- costuri ridicate – în fiecare locaţie trebuie setat un server de baze de date
- greu accesibil – utilizatorul trebuie să reţină IP-ul pentru fiecare locaţie
- nesigur – codurile de acces se află răspândite în multiple locaţii care sunt în grija unor persoane care nu au cunoştinţe în domeniu

Varianta III – O aplicaţie centrală, o baza de date centrală + câte o aplicaţie şi câte o bază de date în fiecare locaţie „remote”.

Această variantă este o variantă de mijloc între cea propusă anterior şi cea aleasă de mine. În acest caz există câte o bază de date în fiecare locaţie în care sunt ţinute datele despre locaţie şi programe automate şi o bază de date centrală în care se ţin doar chei de autentificare şi URL-uri ale serviciilor din locaţiile „remote”. Faţă de varianta anterioară, Figura 4.6, aceasta arhitectură ar fi prezentat avantajul accesului uşor (o singură interfaţă din care se pot accesa multiple locaţii).

Costurile foarte ridicate datorită serverelor SQL din fiecare locaţie reprezintă dezavantajul major al acestei variante în comparaţie cu cea aleasă de mine. Această variantă are avantajul de a putea executa programele automate local nefiind nevoie de comenzi la distanţă care în circumstanţele potrivite (pană de curent prelungită la locaţia centrală, întreruperea conexiunii la Internet în locaţia centrală) ar putea eşua.

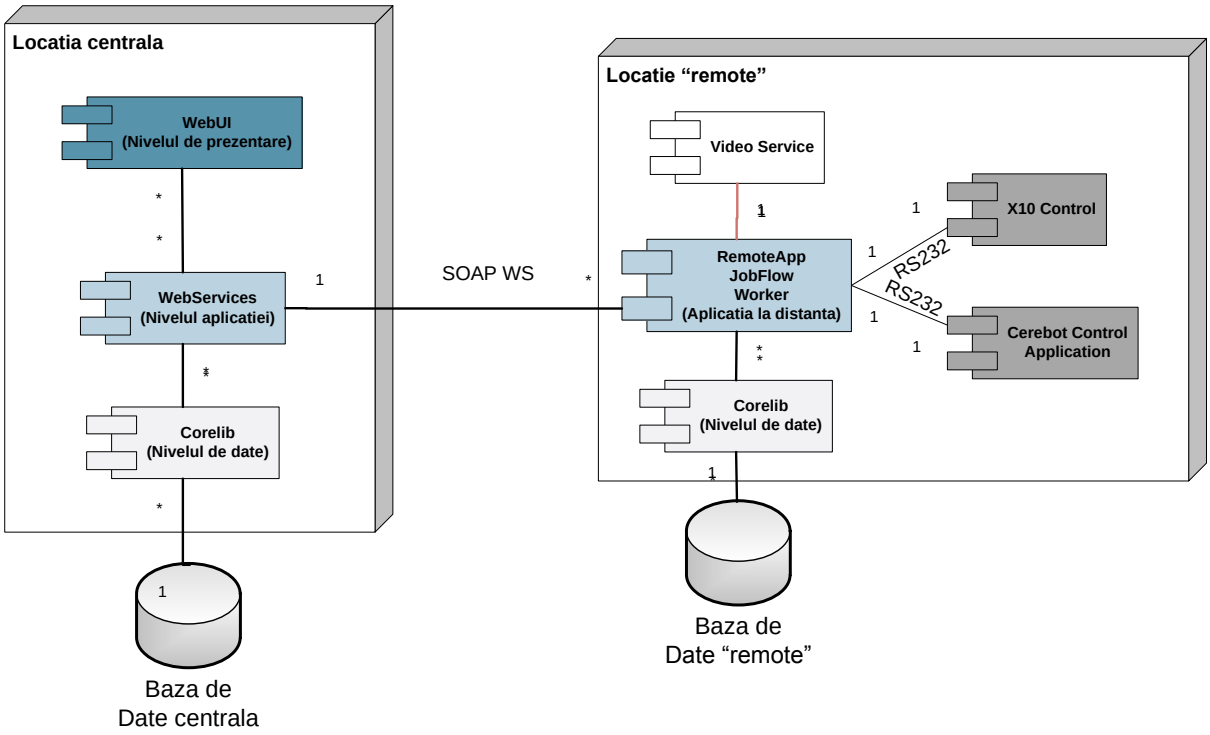


Figura 4.6 Arhitectura posibilă Varianta III

Varianta IV – O aplicație centrală, o bază de date centrală + câte un modul de servicii web în fiecare locație „remote”, toate transferurile între locația remote și client trec prin locația centrală.

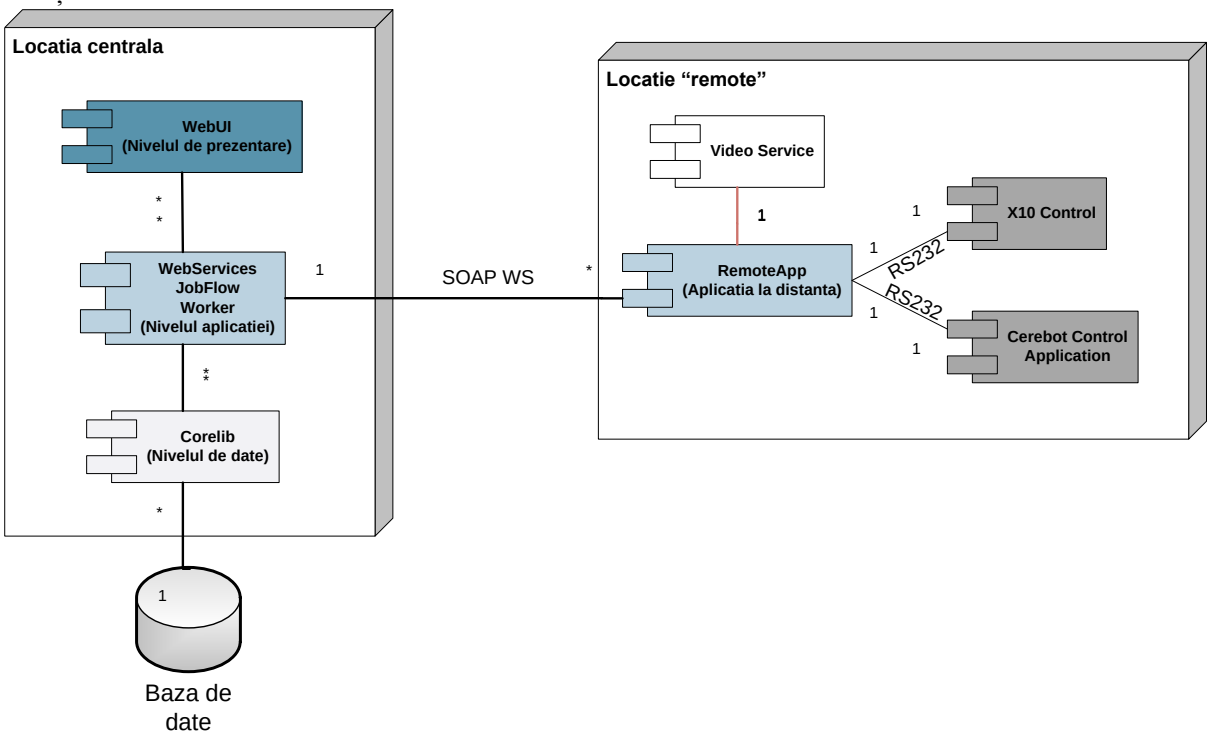


Figura 4.7 Arhitectura posibilă Varianta IV

Această variantă este cea mai apropiată de varianta finală. Spre deosebire de varianta anterioară în această variantă există o singură bază de date în care se țin detaliile despre toate locațiile „remote”. Avantajul centralizării datelor este reprezentat de faptul că în cazul în care

locația centrală devine indisponibilă, toate locațiile vor fi indisponibile și programele automate nu se vor executa pe acea perioadă.

În varianta IV, prezentată în Figura 4.7 toate transferurile se execută prin intermediul locației centrale. Acest aspect poate fi problematic în momentul în care există mulți utilizatori conectați la sistem care urmăresc camerele de supraveghere din locațiile proprii. Deoarece lățimea de bandă este limitată la locația centrală s-ar produce o gâtuire („bottleneck") și calitatea transmisiilor ar fi redusă.

Varianta aleasă și implementată este o derivație a variantei anterioare, singura modificare o constituie modalitatea de transfer a imaginilor de la camerele de supraveghere între locația remote și client. Clientul cere detaliile de conectare (url, nume de utilizator, parola) la serverul video pentru locația dorită de la locația centrală după care face conexiune directă la acesta. Astfel transferul de imagini se face direct de la serverul video din locația remote la browser-ul web al clientului.

În sistemul de față, modulele, prezentate în arhitectura din Imaginea XX sunt următoarele:

- WebAppMS (în locația centrală)
- WebApp.Corelib (în locația centrală)
- WorkerStarter (în locația centrală)
- RemoteApp (locația remote)
- CerebotControlApp (locația remote)

WebAppMS (în locația centrala) reprezintă modulul principal și este situată în locația centrală. Acest modul conține nivelul de prezentare (interfața web) și nivelul logic (serviciile web) al aplicației centrale. Acesta este modulul care trimite comenzile date de utilizator sau de aplicația worker modulelor „remote".

WebApp.Corelib (în locația centrala) este componenta care reprezintă nivelul de date din arhitectura pe trei nivele. Acest modul furnizează datele de care are nevoie aplicația WebAppMS. Baza de date și modulul WebApp.Corebil sunt situate la locația centrală.

WorkerStarter (în locația centrala) este modulul (serviciu Windows) care inițiează în fiecare minut pornirea serviciilor automate. Acest serviciu rulează pe serverul central.

RemoteApp (locația remote) este aplicația care rulează pe serverul de control care se afla în locația „remote". Acest modul este compus din servicii web care sunt apelate de serviciile web din aplicația principală WebAppMS. Aceste servicii dau comenzi aplicației CerebotControlApp situată pe placa cu microcontroler Cerebot sau sistemului de control X10 prin intermediul a două porturi seriale.

CerebotControlApp (locația remote) reprezintă programul scris pentru placa cu microcontroler Cerebot. Acesta citește temperatura de la senzorii de temperatură și deschide sau închide electrovalvele.

Pe lângă modulele descrise sistemul comunică cu alte două module auxiliare, care au fost dezvoltate de firma producătoare de sisteme de supraveghere Geovision respectiv firma producătoare de sisteme de control a dispozitivelor din rețeaua electrica „Marmitek". Aceste module sunt:

- Sistemul de control X10 – Aplicația RemoteApp trimite prin portul serial comenzi sistemului de control X10 care pornește sau oprește dispozitive din instalația electrică.
- Serverul Video Geovision – Acesta comunică direct cu aplicația încărcată în browser-ul de web al clientului din modulul WebAppMS. După ce clientul se conectează la acest modul acesta începe transmisia video(„streaming").

Capitolul 5 Proiectarea de detaliu

Pentru realizarea aplicației am folosit diferite limbaje de programare în funcție de necesitățile fiecărui modul. Limbajele folosite în dezvoltarea aplicației sunt :

- Asp.net, JavaScript și C# pentru aplicațiile web
- C# pentru aplicațiile auxiliare
- C pentru codul de microcontroler

Pentru implementarea modulelor PC am utilizat mediul de dezvoltare Microsoft Visual Studio 2008, împreuna cu .Net Framework 3.5 SP1.

Pentru implementarea programului de microcontroler am utilizat mediul de dezvoltare AVR Studio 4 împreuna cu compilatorul AVR-GCC.

Pentru stocarea datelor am creat o baza de date MS SQL utilizând Microsoft SQL Server 2008 Management Studio.

5.1 Structura sistemului

5.1.1 Diagrama sistemului

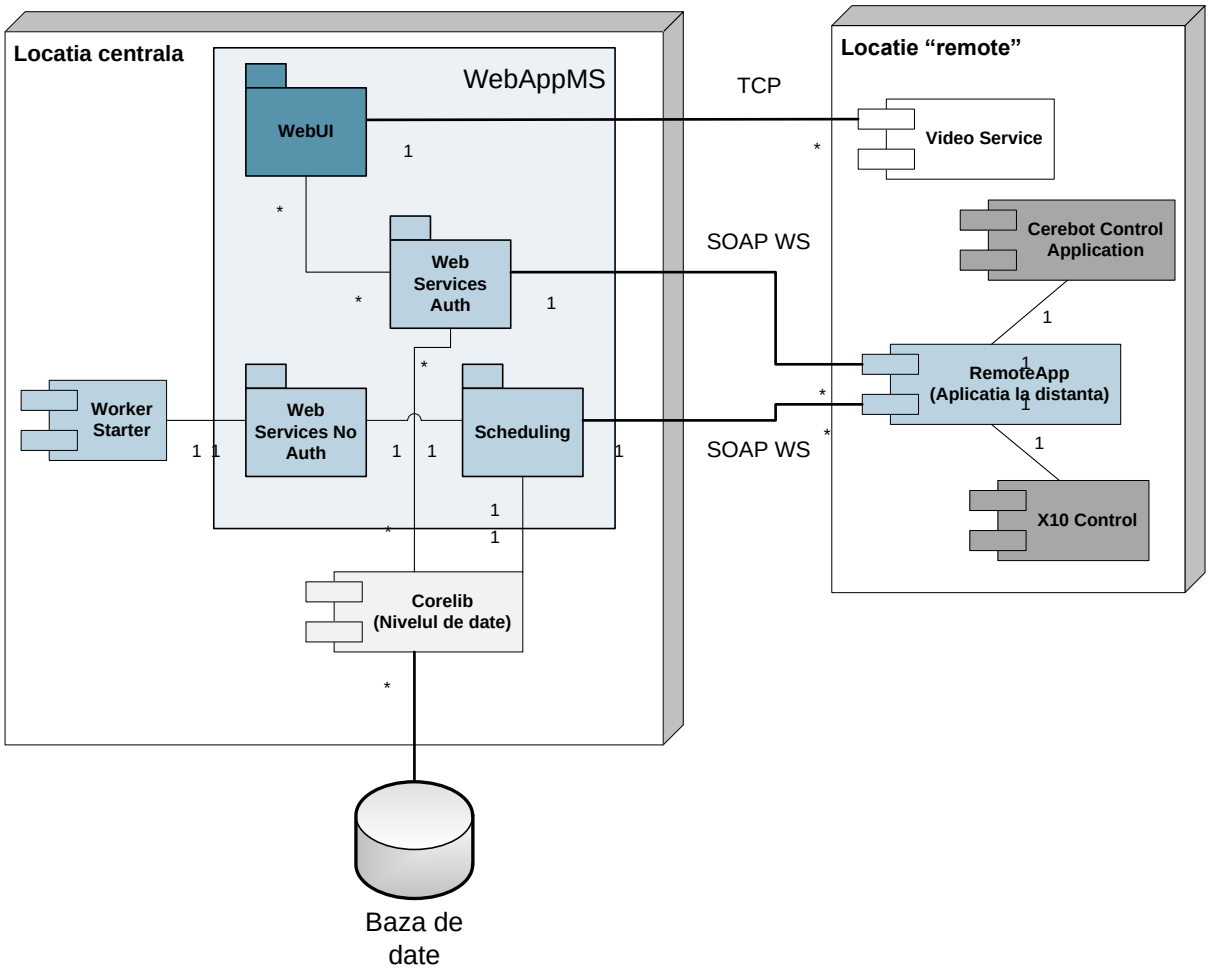


Figura 5.1 Diagrama sistemului

În Figura 5.1 este prezentată diagrama sistemului care este format din cele 7 module: WebAppMS, WorkerStarter, Corelib, RemoteApp, Cerebot Control Application, X10 Control și Video Service. Dintre acestea voi prezenta în detaliu primele cinci deoarece X10 Control și Video Service nu au fost dezvoltate de mine, acestea fiind utilizate în aplicație pentru controlul dispozitivelor electrice și pentru supravegherea video.

5.1.2 *Locația centrală*

Locația centrală are o arhitectură cu 3 nivele:

- nivelul de prezentare este reprezentat de interfața web aflată în pachetul WebUI parte a modulului principal WebApp
- nivelul aplicației este format din serviciile web, modulul de programare automată și diverse utilitare ale modulului principal WebApp și modulul WorkerStarter
- nivelul de date este reprezentat de modulul Corelib.

5.1.2.1 *WebAppMS*

Aceasta este modulul principal al sistemului și este compus din mai multe pachete care îndeplinesc diverse roluri. Modulul conține interfața web a aplicației, serviciile web care leagă nivelul de prezentare de cel de date și de modulul aflat pe computerul la distanță., pachetul de execuție al comenzilor automate și alte utilitare.

5.1.2.1.1 Structura modulului

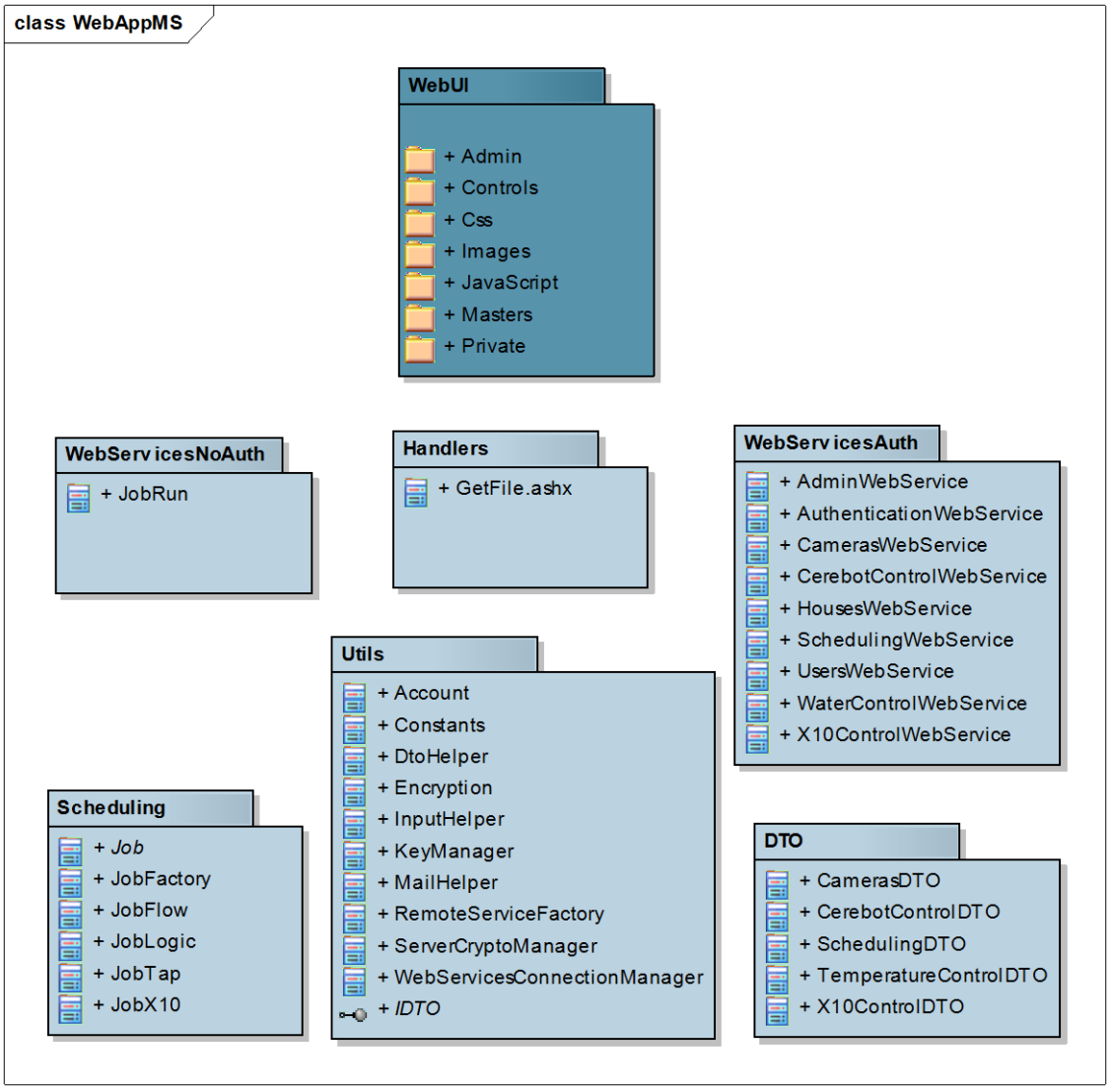


Figura 5.2 Diagrama de pachete a modulului WebAppMS

În Figura 5.2 se poate observa diagrama de pachete a modulului WebAppMS. Pe aceasta diagramă se pot distinge cele două nivele diferite (de prezentare, al aplicației), acestea fiind marcate cu nuanțe diferite.

Pachetul **WebUI** reprezintă nivelul de prezentare al aplicației. Acesta conține interfața web a aplicației și poate fi accesat doar de utilizatori autentificați. Pachetul este structurat pe sub-pachete astfel:

- *Admin* – conține paginile de administrare și un fișier de configurare care permite accesul în această zonă doar utilizatorilor autentificați care au rolul de administrator.
- *Controls* – conține fișiere .ascx(control-uri) care sunt folosite în mai multe locuri în interfață. Control-ul folosit pentru vizualizarea imaginilor video CameraControl.ascx se află în acest subpachet.
- *Css* – conține fișierul StyleSheet.css în care sunt definite stilurile folosite în interfața aplicației.
- *Images* – conține imaginile utilizate în interfață.
- *JavaScript* – acest sub-pachet conține fișiere de tip .js care conțin cod JavaScript și pot fi incluse în diferite pagini ale interfeței. În acest pachet se află fișierul Cookies.js care se ocupă de crearea și citirea cookies-urilor.
- *Masters* – acest sub pachet conține paginile master folosite în interfață: Title.Master, Admin.Master și Private.Master.
- *Private* – conține toate paginile pe care un utilizator (nu administrator) le poate accesa. Aceste pagini pot fi accesate doar de utilizatori autentificați care au rolul de „user”.

Pachetul **WebServicesAuth** conține serviciile web care necesită autentificare pentru a fi accesate. Interfața comunică cu nivelul de date prin intermediul serviciilor web aflate în acest pachet. Comunicația interfeței cu serviciile expuse de aplicațiile la distanță nu se face direct ci tot prin intermediul serviciilor web din acest pachet. Acest pachet conține următoarele servicii web:

- *AdminWebServices.asmx* – conține toate metodele care sunt folosite în partea de administrare: adăugare/ștergere/editare utilizatori, locuințe, controale, mapări etc.
- *AuthenticationWebService.asmx* – conține metodele de autentificare la aplicația principală și la aplicațiile de pe mașinile „remote”.
- *CerebotControlWebService.asmx* – da comenzi plăcii Cerebot și citește valori de la aceasta, interoghează Corelib pentru date referitoare la aceste dispozitive.
- *HousesWebService.asmx* – interoghează Corelib pentru date despre locații.
- *SchedduliungWebService.asmx* – metode de management a task-urilor automate.
- *UsersWebService.asmx* – metode de management a parolelor utilizatorilor.
- *X10ControlWebService.asmx* – interoghează Corelib pentru detalii referitoare la controalele X10 și trimite comenzi aplicației la distanta.

Pachetul **WebServicesNoAuth** conține fișierul JobRun.asmx. Acest fișier conține metoda RunJob care este apelată de modulul WorkerStarter. Metoda JobRun apelează RunAllActiveJobs din Clasa JobFlow. O singură instanță a acestei metode poate rula la un moment dat. Pentru a evita situațiile în care o rulare începe înaintea terminării celei dinainte am folosit un Mutex.

Pachetul **DTO** conține obiecte a căror scop este transferul de date (Data Transfer Object). Acestea sunt obiecte intermediare folosite pentru modelarea datelor primite de la nivelul de date și trimiterea lor către interfață.

Pachetul **Handlers** conține fișierul GetFile.ashx, acesta este folosit pentru descărcarea control-ului ActiveX la prima rulare.

Pachetul **Utils** conține diverse clase utilitare folosite în mai multe locuri în aplicație. Fișierele componente ale pachetului sunt:

- *Account.cs* – la autentificare un obiect de acest tip este creat și adăugat pe sesiune. În acest obiect sunt ținute detaliile utilizatorului și cheile publice și private.
- *Constants.cs* – în acest fișier sunt definite enumerările folosite în aplicație.
- *DtoHelper.cs* – metodele din această clasă statică sunt folosite pentru a copia câmpuri din și în DTO-uri.
- *Encryption.cs* – clasa Encryption conține metodele de criptare/decriptare RSA, AES, SecureString și cele de generare a parolelor automate.
- *InputHelper.cs* – clasa conține diverse metode statice care ajută la transformarea parametrilor non nullable în nullable.
- *KeyManager.cs* – conține metode care generează chei AES și vectori de inițializare AES, generează chei RSA, generează hash SHA384.
- *MailHelper.cs* – conține metoda de trimitere a unui mail la o adresă dată.
- *Mails.cs* – adaugă cererea de trimitere a unui mail într-un ThreadPool și folosește clasa *MailHelper* pentru a trimite mail.
- *RemoteServiceFactory.cs* – conține metoda factory de generare a un provider de tipul cerut. La generare se setează locația serviciilor (citită din baza de date) și se setează cookie-ul folosit la autentificarea la serverul „remote”.
- *ServerCtyptoManager.cs* – conține metodele de criptare/decriptare utilizate de server pentru programele automate. Acestea utilizează metodele de criptare AES definite în Encryption.cs.
- *WebServicesConnectionManager.cs* – conține metodele de handling a conexiunii la baza de date. În metoda GetConnection conexiunea este citită de pe sesiune, dacă ea nu există este creată.

Pachetul **Scheduling** conține clasele care se ocupă de rularea automată a programelor. Metoda RunAllActiveJobs din clasa JobFlow este apelată de serviciul automat. Clasele din acest pachet implementează pattern-ul arhitectural Factory.

5.1.2.1.2 Șabloane arhitecturale în modulul WebAppMS

Am folosit **Factory Method** pentru execuția task-urilor automate. Folosirea acestui șablon permite ca pe viitor adăugarea a noi task-uri să se facă cu ușurință. Modulul WorkerStarter apelează o dată pe minut serviciul web RunJob din JobRun.asmx. În această metodă se creează o instanță a clasei JobFlow și se apelează metoda RunAllActiveJobs.

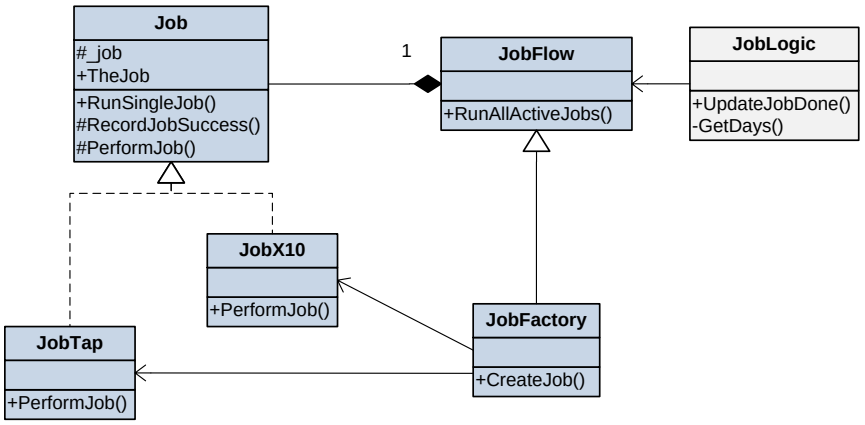


Figura 5.3 Diagrama de clase a pachetului Scheduling

În metoda `RunAllActiveJobs` se aduc din baza de date toate task-urile care trebuie executate în minutul curent. Pentru fiecare dintre acestea se apelează metoda CreateJob() din clasa JobFactory.

În Figura 5.3 este prezentată diagrama de clase a pachetului Scheduling. Metoda `PerformJob` se autentifică la serverul remote după care execută operația specifică (oprire/pornire dispozitiv electric în cazul JobX10, închidere/deschidere electrovalva în cazul JobTap). Pe job-ul nou creat se execută metoda `RunSingleJob`.

Am folosit șablonul arhitectural **Factory Method** și pentru instanțierea serviciilor remote. Aceste servicii se obțin prin metoda `GetProvider` din clasa RemoteServiceFactory. La instanțierea acestor servicii se apelează metoda `MaintainSession` care citește cookie-ul de autentificare la serverul remote de pe sesiune adăugându-l instanței, cu ajutorul acestuia id-ul sesiunii din apelul făcut va coincide cu cel de la apelul de autentificare. Tot în această metodă se setează și locația serviciilor de pe serverul „remote”.

Pentru instanțierea conexiunii la baza de data am folosit șablonul arhitectural **Singleton**. Metoda `GetConnection` din WebServicesConnectionManager citește de pe sesiune conexiunea, dacă aceasta există ea este întoarsă ca răspuns la metodă, în caz contrar este creată o nouă conexiune și adăugata pe sesiune.

5.1.2.2 *WebApp.CoreLib*

Acest modul reprezintă nivelul de date al aplicației. Conține obiectele de date, providerii de date și conexiunea la baza de date. WebAppMS face toate interogările la baza de date prin intermediul acestui modul.

5.1.2.2.1 Structura modulului

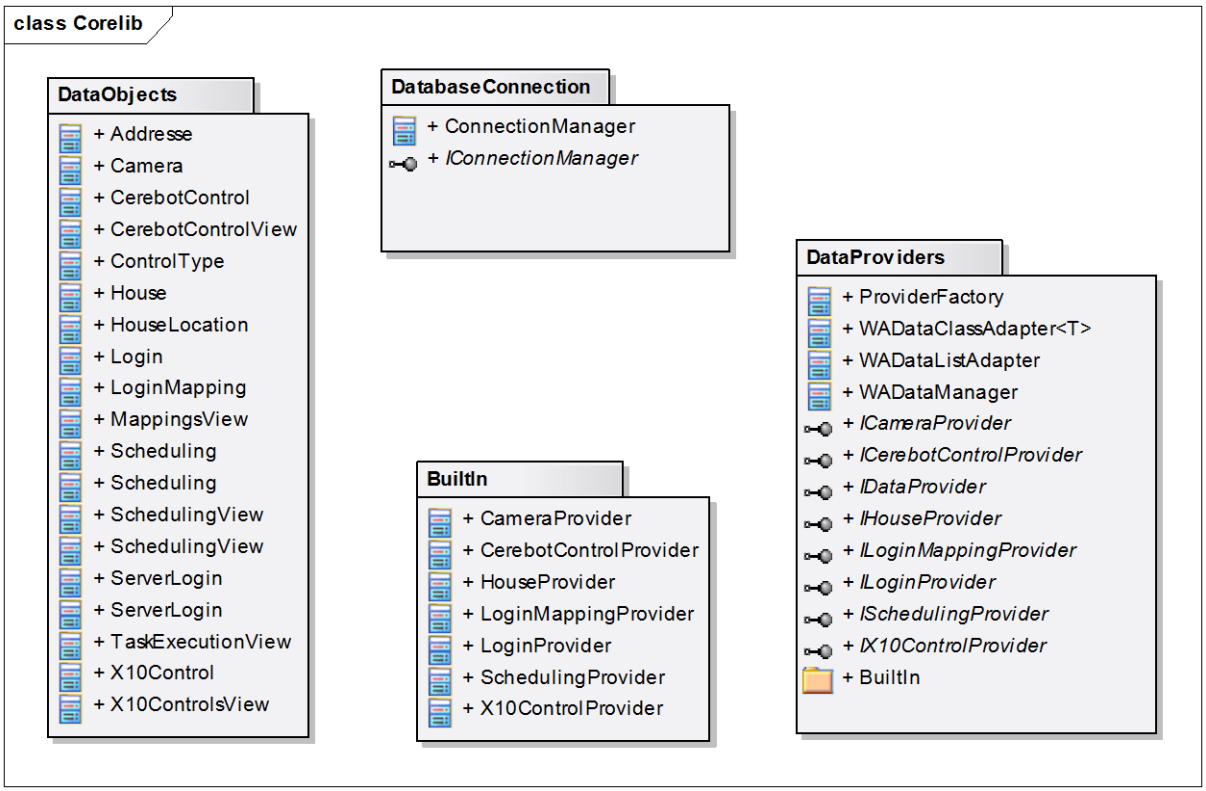


Figura 5.4 Diagrama de pachete a modulului Corelib

În Figura 5.4 este prezentată diagrama de pachete a modulului Corelib. Pachetul **DatabaseConnections** conține interfața ConnectionManagerInterface și clasa ConnectionManager. Metodele interfeței sunt:

- CloseConnection() – închide conexiunea
- GetConnection() – întoarce conexiunea existentă sau creează una nouă
- GetDedicatedConnection() – creează o conexiune nouă chiar dacă deja există una
- SetConnection(DbConnection cx) – setează o conexiune

În clasa ConnectionManager este definită conexiunea curentă „_current" de tip IConnectionManager. Aceasta este folosită în interiorul acestui modul și este setată de serviciile apelante. În cazul modulului WebAppMS în fișierul Global.asax pe evenimentul Application_Start se setează: `ConnectionManager.Current = new WebServicesConnectionManager();` WebServicesConnectionManager a fost implementat în modulul WebAppMS, aceasta este o aplicație web deci vom avea o conexiune pe sesiune.

Pachetul **DataObjects** conține obiectele folosite pentru citirea datelor din baza de date. Acestea sunt mapări ale tabelelor și view-urilor din baza de date. Toate clasele din acest pachet extind clasa abstractă DataClass care face parte din utilitarul Zonkey36 pe care l-am folosit pentru citirea și scrierea datelor în baza de date.

Pachetul **DataProviders** conține providerii de date. Pachetul conține:

- ProviderInterfaces.cs – conține interfețele tuturor providerilor. Toți providerii de date implementează interfața comuna IDataProvider.
- ProviderFactory.cs – este apelat pentru generarea unui provider de un anumit tip, metodele acestei clase sunt folosite de modulul WebAppMS.
- ZonkeyHelpers.cs – conține declarațiile claselor WADataClassAdapter<t>, WADataManager, WADataListAdapter care extind clasele corespunzătoare din utilitarul Zonkey și sunt folosite în providerii de date. În aceste clase este setată conexiunea ca fiind cea curentă din ConnectionManager.
- Subpachetul BuiltIn – conține implementările providerilor din interfețele definite în ProviderInterfaces.cs.

5.1.2.2.2 Șabloane arhitecturale în modulul CoreLib

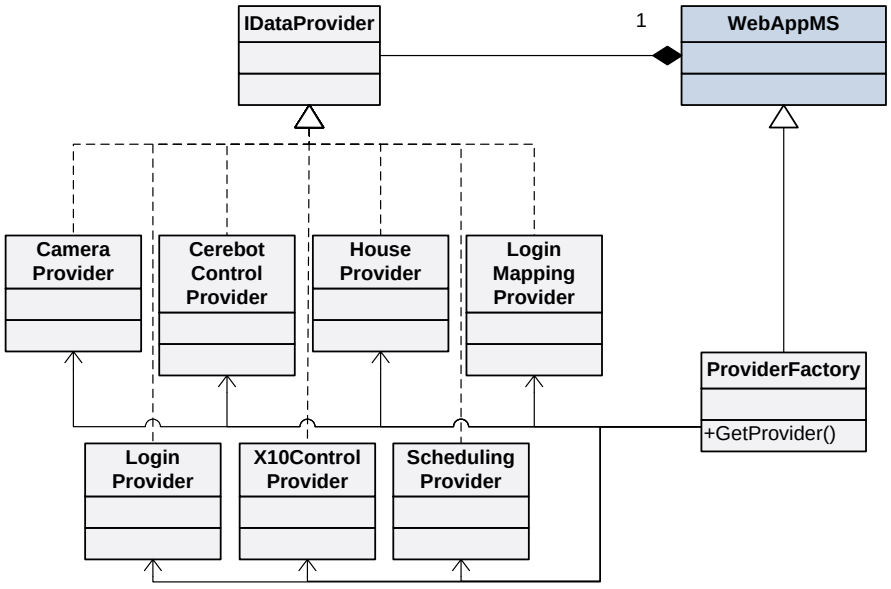


Figura 5.5 Data Provider Factory

Am structurat modulul Corelib pe baza șablonului arhitectural **Factory Method**. În Figura 5.5 este prezentată o diagramă de clase a acestui modul. În această diagramă nu am inclus fișierul ZonkeyHelpers și clasele de conexiune deoarece conexiunile sunt inițiate în clasele definite în ZonkeyHelpers iar aceste clase sunt folosite în implementările tuturor providerilor.

5.1.2.3 *WorkerStarter*

Acest modul este un serviciu Windows care inițiază pornirea programelor automate. Modulul are în componență patru fișiere: App.config, Program.cs, ProjectInstaller.cs, WorkerStarter.cs.

În fișierul de configurație App.config este păstrat intervalul de apel și calea către serviciul web care este apelat în momentul în care timpul stabilit s-a scurs. ProjectInstaller.cs este fișierul de instalare a serviciului. Acesta este autogenerat. În acesta s-a setat tipul de pornire al serviciului pe modul automatic.

Fișierul WorkerStarter.cs conține codul principal al acestui modul. În acesta rulează un timer care la expirarea fiecărui minut rulează serviciul JobRunWebService din aplicația WebAppMS. Acest serviciu este singurul care poate fi apelat fără autentificare prealabilă.

5.1.3 Locațiile Remote

În fiecare locație „remote” va exista câte o instanță a modulului RemoteApp și câte o placă Cerebot care va avea înscrisă în microcontroler aplicația CerebotControl.

5.1.3.1 *RemoteApp*

Modulul RemoteApp este o colecție de servicii web care trimit comenzi prin intermediul porturile seriale modulelor de control Cerebot și X10. Acest modul expune serviciile web AuthenticationServiceRemote.asmx, WaterControlServiceRemote.asmx și X10ControlServiceRemote.asmx care sunt utilizate de locația centrală pentru a efectua comenzile date de utilizator din interfața sau pentru a executa programele automate.

Pe lângă serviciile web modulul mai are în componență următoarele fișiere: Constants.cs, Global.asax, UserManager.cs, Web.config.cs. În Figura 5.6 este prezentată diagrama de clase a modulului RemoteApp.

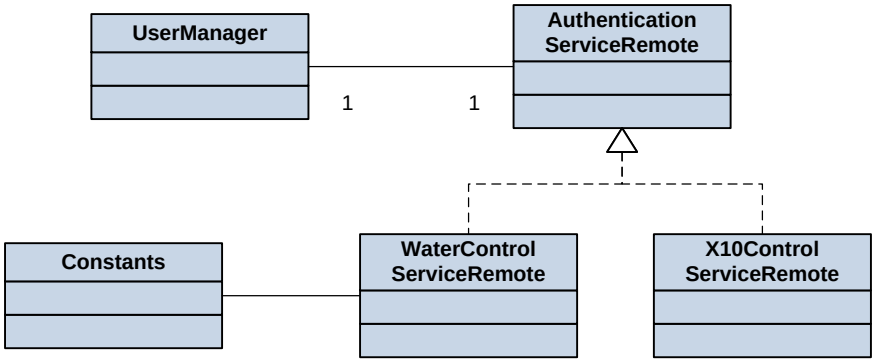


Figura 5.6 Diagrama de clase aplicația „remote”

Fișierul **Constants.cs** conține o enumerațiile CerebotCommands, CerebotControls, CerebotWaterCommands, CerebotStatusCommands în care sunt definite comenzile și subcomenzile care se pot fi date plăcii Cerebot pentru controlul electrovalvelor sau citirea temperaturii.

În fișierul **Global.asax** pe evenimentul de pornire a aplicației „Application_Start” se deschid conexiunile pe porturile seriale pentru comunicația cu placa Cerebot și cu modulul de tip X10, CM11.

Clasa **UserManager.cs** conține metoda Authenticate (string UserName, string Password) care verifică în web.config dacă există definita în AppSetings o pereche cu cheia egală cu numele de utilizator și cu parola echivalentă cu parola furnizată. În caz afirmativ va întoarce adevărat altfel fals în acest caz respingându-se cererea de autentificare.

Fișierul **Web.config** este fișierul de configurare în care în tag-ul AppSettings se definesc perechile cu cheia CerebotPort și X10Port a căror valoare va fi portul serial pe care se execută conexiunea și se va defini un număr de perechi de tip <username;password> cu care se poate autentifica modulul central la acesta.

Serviciul web **AuthenticationServiceRemote.asmx** conține metodele de autentificare și părăsire a aplicației. În constructorul acestui serviciu se verifică dacă utilizatorul este autentificat, prin verificarea existenței numelui de utilizator pe sesiune. Dacă acesta este autentificat atunci se creează un nou GenericPrincipal care este setat ca și CurrentPrincipal pentru threadul curent. Celelalte servicii din acest modul extind acest serviciu. Fiecare metoda are ca și header:

```
[PrincipalPermission(SecurityAction.Demand, Authenticated = true)].
```

Utilizatorului i se va da acces la metoda doar dacă este autentificat, altfel va fi aruncată o excepție.

Serviciul web **WaterControlServiceRemote.asmx** expune metode de control a dispozitivelor conectate la modulul Cerebot. Metodele acestui serviciu sunt:

- TurnOn(string code) – primește ca parametru pinul la care este conectat semnalul de enable al modulului PmodHB3 conectat la electrovalva care trebuie controlată. Trimite comanda de deschidere electrovalvei.
- TurnOff(string code) – primește ca parametru pinul la care este conectat semnalul de enable al modulului PmodHB3 conectat la electrovalva care trebuie controlată. Trimite comanda de închidere electrovalvei.
- GetTemp(string code) – primește ca parametru pinul pe care se va citi temperatura și citește temperatura. Pentru citire trimite comanda de citire, așteaptă pentru ca citirea să se efectueze după care citește răspunsul de pe portul serial.

5.1.3.2 *CerebotControl*

Modulul CerebotControl este format din aplicația scrisă pentru microcontrolerul Atmel ATMEGA64 de pe placa Cerebot [5]. Modulul a fost implementat folosind limbajul C și mediul AVR Studio 4.

Modulul CerebotControl primește comenzi de la aplicația „remote” prin portul serial, le interpretează și trimite comenzile mai departe către modulele adiacente PmodHB3 sau PmodTMP.

Acest modul constă din fișiere header(*.h) și fișiere cod C (*.c). Fișierele header componente ale modului sunt: cerebot.h, dispatch.h, Status.h, stdtypes.h, tmser.h și WaterCtrl.h. Fișierele cod componente ale modulului sunt main.c, PmodTmpDriver.c, Status.c, tmser.c și WaterCtrl.c.

Fișierul **cerebot.h** conține definițiile tuturor porturilor, pinilor și adreselor de pe placa Cerebot. În fișierul **stdtypes.h**(standard types) sunt definite anumite tipuri care sunt folosite în aplicație(BYTE, WORD, fFalse). Restul fișierelor header au asociat câte un fișier *.c, în fișierul header fiind definite semnăturile metodelor și constantele folosite în cadrul fișierului.

Fișierul **main.c** este modulul principal al aplicatiei. După execuția pașilor de inițializare se intră într-o buclă care nu are condiție de terminare. La fiecare execuție a buclei se verifică dacă au sosit date pe portul serial. În caz afirmativ se parsează comanda și se verifică dacă este validă. Comanda poate reprezenta cmdQuery, cmdVersion, cmdReset sau cmdExecute. În cazul în care comanda este cmdExecute se citește comanda byte din comanda

prin metoda *TktParseNext(char *szToken)*. În cazul în care metoda răspunde cu valorea *tktKwd* atunci se apelează metoda *FDispatchCommand(char * szCmd, char * rgszTable[], PFN rgpfnTable[]*.

```
BOOL
FDispatchCommand(char * szCmd, char * rgszTable[], PFN rgpfnTable[])
	{
	int		isz;
	void	(*pfn)(void);
	/* Se sar spatiile albe.
	*/
	while (*szCmd == ' ') {
		szCmd += 1;
	}
	/* Se cauta un nume de comanda care sa coincida cu numele de comanda
	** primit.
	*/
	isz = 0;
	while (rgszTable[isz] != 0) {
		if (strcmp(szCmd,rgszTable[isz]) == 0) {
			/* S-a gasit un nume potrivit.
			*/
			pfn = rgpfnTable[isz];
			/* Se executa comanda din tabela de comenzi.
			*/
			(*pfn)();
			return fTrue;
		}
		isz += 1;
	}
	/* Nu s-a gasit nici un nume potrivit.
	*/
	PutComSz(szMsgBadCommand);
	return fFalse;
	}
```

Metoda FdispatchCommang primește ca parametru comanda (*szCmd*), lista de tipuri de comenzi posibile (*rgszTable*) și lista de metode de care se executa pentru fiecare sursă (*rgpfnTable*).

Fisierele **Status.c** și **WaterCtrl.c** conțin implementarile metodelor (*DoExecuteStatusCommand* și *DoExecuteWaterCommand*) de execuție a comenzilor de status și de control a electrovalverlor.

Metoda *DoExecuteStatusCommand* intră într-o buclă while până la primirea unei noi comenzi pe portul serial. Comanda de citire a temperaturii este 'T'. La primirea acestei comenzi se execută metoda GetTemperature() [4] după care se trimite temperatura pe portul serial spre aplicația „remote".

Metoda *DoExecuteWaterCommand* [6] intră într-o buclă while până la primirea unei noi comenzi pe portul serial. Comanda de inchidere a unei electrovalve este 'C' iar pentru deschidere 'O'. Comanda este urmată de numărul pinului la care este conectat semnalul de enable al modulului PmodHB3 căruia i se trimite comanda. Acest fișier conține și metodele de deschidere și închidere a electrovalvei.

Fisierul **PmodTmpDriver.c** conține metodele de inițializare(*InitPmodTmp*) a modulului PmodTMP și citirea temperaturii(*GetTemperature*) de la acesta. Fișierul **tmser.c** conține metodele necesare pentru realizarea comunicației seriale(inițializare, citire, scriere).

5.2 Interfaţa cu utilizatorul

5.2.1 Prezentarea generală

Locațiile „remote” pot fi accesate prin intermediul interfeței web a aplicației. Interfața utilizator este parte a modulului WebAppMS. Aceasta a fost realizată în asp.net folosind componente html, asp și componente din AjaxControlToolkit 3.0.

Interfața are o pagină publică pentru autentificare și două zone private care pot fi accesate pe baza rolului pe care îl are utilizatorul în aplicație. Zonele private sunt: zona de administrare și zona de control.

Interfața este compusă din următoarele pagini:

- Zona de administrare:
  - Users.aspx
  - Houses.aspx
  - HouseDetails.aspx
  - ChangePassword.aspx
- Zona utilizator
  - Appliances.aspx
  - HomeSelection.aspx
  - Scheduling.aspx
  - Status.aspx
  - Surveillance.aspx
  - ChangePassword.aspx

Paginile din zona de administrare au ca și Master Page, pagina *Admin.Master* iar paginile din zona de utilizator au ca și Master Page, pagina *Private.Master*. Atât *Private.Master* cât, *Admin.Master* cât și *Login.aspx* au Master Page comun *Title.Master*. Pagina *Title.Master* este reprezentată în Figura 5.7.

Figura 5.7 Title.Master

Paginila *Admin.Master* și *Private.Master* sunt similare. În partea de sus a panel-ului aplicației, cel ce este adăugat paginii Title.Master de aceste două pagini master se află butoanele de legătură către paginile accesibile rolului curent.

Pentru ca toate paginile din aplicație să aibă același „look” am folosit fișierul StyleSheet.css în care am definit diferite stiluri. În rândurile următoare este un exemplu de definire a stilului pentru „footer-ul” unui grid și pentru componente din acesta „span” și „td”

```
gridFooter
{
    background: #e5e5e5 url(../Images/greyGrd.gif) repeat-x;
    color: #454545;
}

.gridFooter span
{
    display: block;
    border: solid 1px #cecece;
    background: #dbdbdb;
    color: #3d3d3d;
    padding-left: 5px;
    padding-right: 5px;
}

.gridFooter td
{
    border: 0;
    padding: 3px;
}
```

Pe majoritatea paginilor din interfața pentru afișarea mai multor înregistrări de același tip am folosit tabele, componente „asp:GridView”. Acestea au setată ca și sursă de date un „asp:ObjectDataSource” care are definită în fișierul de cod(code behind) o metodă de selecție pentru elementele de pe pagina curentă, o metoda de numărare a elementelor, o metodă de update și o metodă de ștergere a unui element. În rândurile următoare am pus un fragment de cod reprezentativ pentru un astfel de tabel, fragmentul face parte din pagina de vizualizare și adăugare de controale unei locuințe și este folosit pentru vizualizarea, editarea sau ștergerea unui control Cerebot.

```
<asp:ObjectDataSource ID = "odsCerebotGrid" runat = "server" TypeName =
"WebApp.WebUI.Admin.HouseDetails" SelectMethod = "GetCerebotControls"
EnablePaging = "True" SelectCountMethod = "TotalNumberOfCerebotControls"
UpdateMethod = "UpdateCerebotControls" MaximumRowsParameterName = "maxRows"
StartRowIndexParameterName="startRows" SortParameterName= "SortExpression">
</asp:ObjectDataSource>
<asp:GridView ID="gvCerebotControls" runat="server" DataSourceID =
"odsCerebotGrid" DataKeyNames = "ControlID" AutoGenerateColumns = "False"
AllowPaging = "True" PageSize="15" DeleteMethod="gCerebot_RowDeleting"
OnRowCommand="gvCerebot_RowCommand" AllowSorting="true" CssClass="grid"
HeaderStyle-CssClass="gridHeadInAccordion" RowStyle-CssClass="row0"
AlternatingRowStyle-CssClass="row1">
<Columns>
      <asp:CommandField ShowEditButton="True" HeaderStyle-Width="40px"
      EditText = "E" UpdateText="U" CancelText="C">
      </asp:CommandField>
      …
      …
</Columns>
</asp:ObjectDataSource>
```

În cadrul definirii grid-ului trebuie definite coloanele acestuia. În aceasta lucrare am folosit trei tipuri de coloane: “asp:BoundField” care se leagă direct la o coloană din sursa de date și va fi de tip text, “asp:CommandField” este un câmp de comandă și “asp:TemplateField” în care se pot adăuga elemente custom pentru vizualizare și editare a coloanei(textbox, dropdownlist, checkbox, etc).

Pentru a adăuga cât de multă funcționalitate unei pagini, pe multe pagini am folosit controlul “ajaxToolkit:Accordion” care permite deschiderea succesivă a multiple ferestre în aceeași zonă. Astfel se pot adăuga diverse funcții aceleași pagini folosind un spatiu limitat. Pentru a folosi un astfel de control trebuie definit un element “ajaxToolkit:Accodion” care va avea multiplii fii de “ajaxToolkit:AccordionPane”. În următoarele rânduri voi prezenta un fragment de cod folosit în declararea controlului accordion în pagina “HouseDetails.aspx”

```
<ajaxToolkit:Accordion ID="ControlsAccordion" runat="Server"
SelectedIndex="0" HeaderCssClass = "accordionHeader" HeaderSelectedCssClass
= "accordionHeaderSelected" ContentCssClass = "accordionContent" AutoSize =
"None" FadeTransitions = "true" TransitionDuration = "250"
FramesPerSecond="40" RequireOpenedPane="false" SuppressHeaderPostbacks =
"true">
      <Panes>
      <ajaxToolkit:AccordionPane ID="AccordionPaneLocations" runat="server"
      HeaderCssClass="accordionHeader" HeaderSelectedCssClass =
      "accordionHeaderSelected" ContentCssClass="accordionContent">
            <Header>
                  Locations
            </Header>
            <Content>

            </Content>
      </ajaxToolkit:AccordionPane>
      …
      …
      <ajaxToolkit:AccordionPane ID="AccordionPaneOControls" runat="server"
      HeaderCssClass="accordionHeader" HeaderSelectedCssClass =
      "accordionHeaderSelected" ContentCssClass="accordionContent">
            <Header>
                  Other Controls
            </Header>
            <Content>

            </Content>
      </ajaxToolkit:AccordionPane>
      </Panes>
</ajaxToolkit:Accordion>
```

În interiorul tag-urilor Header se va pune titlul ferestrei, care este vizibil și când fereastra este inchisă. În interiorul tag-urilor content se va pune continutul ferestrei.

Pe langa elementele descrise în rândurile de mai sus, am folosit în foarte multe locuri controlul “asp:UpdatePanel”. Acesta este foarte util în aplicații web deoarece permite reîmprospătarea condiționată a unei porțiuni din fereastră, fără a fi nevoie la fiecare schimbare de o reîncărcare completă. Exemplul următor prezintă folosirea unui astfel de control.

```
<asp:UpdatePanel ID="updLoginsGrid" runat="server"UpdateMode="Conditional">
      <ContentTemplate>
      …
      …
      </ContentTemplate>
</asp:UpdatePanel>
```

În aplicație acest Update Panel conține tabelul cu utilizatorii din sistem. Dacă un utilizator este adăugat se va apela metoda updLoginsGrid.Update(), care va reîncarcă doar ceea ce se află în interiorul tag-urilor <ContentTemplate> și anume tabela cu utilizatorii.

5.2.2 Administrare

Zona de administrare poate fi accesată de utilizatorii care au rolul de „admin”. Această zonă este concepută pentru a fi utilizată de administratorii de sistem, care cunosc configurația sistemului din clădirea care urmează să fie adăugată și cunosc bine aplicația.

Fiecare pagină din această zonă de administrare are ca pagină master pagina Admin.Master. În pagina Admin.Master există butoanele de legătură „Houses” și „Users” care deschid paginile „Houses.aspx” respectiv „Users.aspx”.

Pagina Users.aspx este folosită pentru adăugarea utilizatorilor noi. După cum se poate vedea în Figura 5.8 această pagină conține un grid în care sunt afișate toate conturile utilizatorilor.

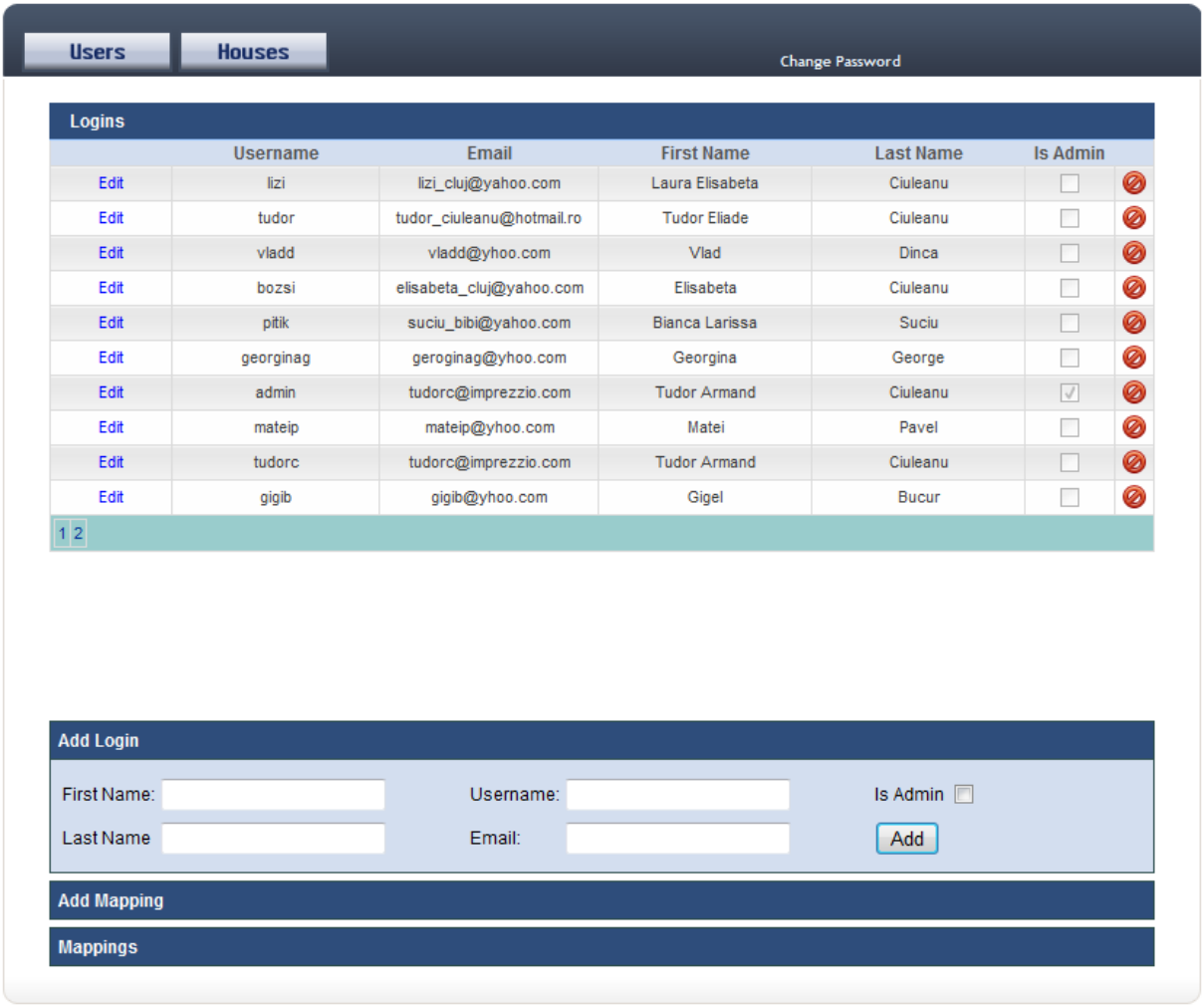


Figura 5.8 Users.aspx

În partea de jos a ferestrei se află un control din AjaxControlToolkit, „accordion” care permite deschiderea alternativă a tab-uri în interiorul lui: „Add Login” – adăugare de utilizator, „Mappings” – vizualizarea mapărilor utilizator-locuința și „Add Mapping” pentru adăugarea unei noi mapări utilizator-locuință.

Pagina Houses.aspx este folosită pentru vizualizarea locuințelor din sistem și adăugarea de locuințe noi. Această pagină conține un Grid cu locuințele în partea superioară și un accordion în partea inferioară în care se completează datele pentru adăugarea unei noi locuințe. Pentru adăugarea de dispozitive pentru o locuință se va apasă „lupa” din dreptul locuinței respective. Această acțiune va face o redirecționare către pagina HouseDetails.aspx. Id-ul casei curente va fi adăugat url-ului. URL-ul va arata astfel *https://www.smart-homes.ro/WebUI/Admin/HouseDetails.aspx?houseid=2d023841-6d1a-48cb-beca-c6583dcde249* .

Pagina HouseDetails.aspx este pagina din care se adaugă, editează sau șterg detaliile unei locuințe, mai exact, dispozitivele controlabile din aceasta. Acest lucru se face deschizând unul din cele 3 tab-uri ale accordion panel-ului de pe pagină, completând datele necesare și apăsând butonul add. Fiecare tab conține o tabelă care afișează dispozitivele existente pană în momentul curent.

5.2.3 Client

Zona pentru clienți poate fi accesată de utilizatorii care au rolul de „user”. Această zonă este concepută pentru a fi utilizată de clienții sistemului, cei care doresc să își acceseze locuința prin intermediul sistemului. Utilizatorii nu trebuie să aibă cunoștințe speciale pentru a face acest lucru.

Controlul Video CameraControl.ascx

Pentru supravegherea video am folosit un control LiveX8220 pus la dispoziție de firma GeoVision, producătoarea plăcii de captură video și a softului pentru aceasta. Acest control este unul Windows, nu web. După transformarea lui în ActiveX control am putut să îl integrez în aplicație.

Pentru a nu reface același cod în multiple locații am creat CameraControl.ascx un wrapper în jurul controlului existent LiveX8220. Wrapperul îi adaugă butoane de „play”, „start”, „stop”, „pause”, „resolution”, „full screen”, „print” și „change camera”.

Pentru a fi cât de eficient am făcut wrapperul în totalitate să ruleze „client-side”. Acest lucru a fost posibil scriind cod JavaScript pentru majoritatea funcțiilor wrapper-ului. Acest lucru înseamnă ca apasărea oricărui buton de control al imaginii nu va genera un postback ci se va executa local. La crearea paginii controlul „LiveX8220” este adăugat dinamic. În Figura 5.9 se poate observa controlul final CameraControl.ascx

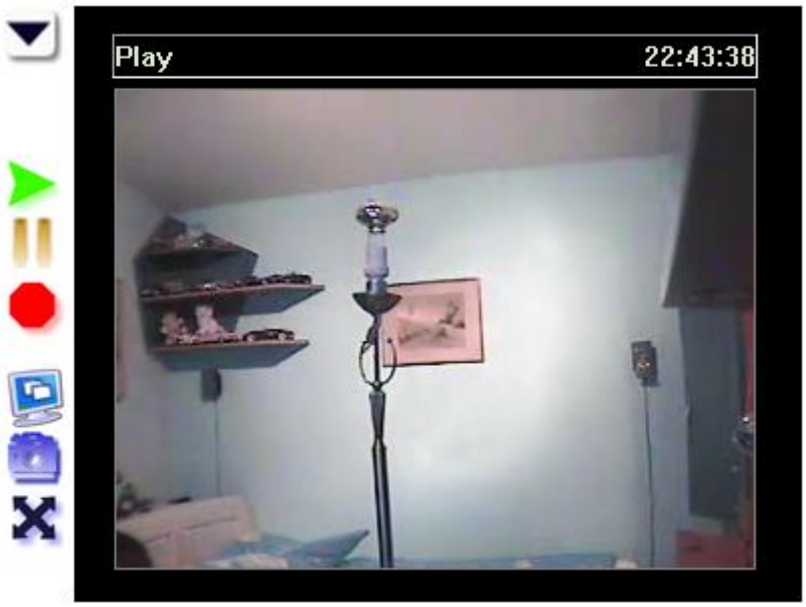


Figura 5.9 CameraControl.ascx

Pe lângă funcționalitatea standard am dorit ca acest control video să își păstreze starea(pornit/pauză/oprit) în timpul navigării de la o pagină la alta. Pentru acest lucru la fiecare acțiune se va adaugă elementul într-un cookie dacă nu există deja sau se va edita elementul existent. Codul de mai jos evidențiază oprirea vizualizării unei camere video.

```
function stop(str) {
    //debugger;
    try {
        //se citeste id-ul dat de server al controlului apelant
        var stringID = str.toString();
        //se inlocuieste tag-ul client al controlului apelant cu
        //id-ul controlului curent
        stringID = stringID.replace("btnStop", "LiveX");
        //se obtine elementul curent
        var obj = document.getElementById(stringID);
        if (obj == null)
            return;
```

```
            //se apeleaza functia de oprire
            obj.StopX();
            //se salveaza starea curenta
            saveCameraState(stringID, "stop", true);
        } catch (e) {
            show_confirm();
        }
    }
```

În fragmentul de cod anterior după schimbarea stării controlului se apelează metoda de salvare a stării, prezentată în rândurile ce urmează. Această metodă citeste coockie-ul cu stări, îl parcurge și modifica elementul găsit sau adaugă un nou element. Cookieul este format din grupări de câte 3 elemente “control”,”stare”,”vizibilitate”. Toate sunt despărtite prin virgulă.

```
function saveCameraState(control, state, visibility) {
    //se citeste cookie-ul "CamerasState3" care contine
    //detaliile despre starea camerelor
    var elem = readCookie("camerasState3");
    if ((elem != "") && (elem != null)) {
        var temp = new Array();
        //se imparte coockieul dupa ','
        temp = elem.split(",");
        var found = false;
        //se cauta controlul apelant și se face modificarea
        for (var i = 0; i < temp.length; i = i + 3) {
            if (temp[i] == control) {
                temp[i + 1] = state;
                temp[i + 2] = visibility;
                found = true;
                break;
            }
        }
        //dacă nu s-a gasit controlul acesta se adauga cookie-ului
        if (found == false) {
            elem = elem+","+control+","+state+","+visibility;
        }
        else {
        //se reformeaza cookie-ul
            elem = "";
            elem = temp.join(",");
        }
    }
    else {
        //se adauga elementul nou cookie-ului
        elem = control + "," + state + "," + visibility;
    }
    //se creeaza cookieul
    createCookie("camerasState3", elem, 1);
}
```

La fiecare încărcare a controlului (evenimentul window.onload) se aplează metoda setCameraState() care setează stările controalelor video afișate.

Pagina Private.Master

Fiecare pagină din această zonă client are ca pagină master pagina Private.Master. Pagina Private.Master este similară cu pagina Admin.Master, conține legăturile „Surveillance”, „Appliances”, „Scheduling” și „Status” care trimit utilizator la paginile „Surveillance.aspx”, „Appliances.aspx”, „Scheduling.aspx” respectiv „Status.aspx”. Pe lângă acestea în partea dreaptă superioară există o icoană care reprezintă un binoclu și deschide o fereastră care conține un control video CameraControl.ascx.

Fereastra care conține controlul video este o fereastră plutitoare „ajaxToolkit:FloatingPannel” care poate fi mișcată în orice zonă a ecranului. Aceasta poate fi de ajutor dacă se dorește urmărirea live a execuției unor comenzi. Fereastra plutitoare setează cookie-ul „floatCPanelProp” care conține coordonatele x și y ale ferestrei pe ecran și starea sa, vizibil sau ascuns. Când se face navigarea de la o fereastră la alta, fereastra video își va păstra poziția și va rămâne în starea precedentă. În Figura 5.10 se poate observa această fereastră video deschisă în pagina „Appliances.aspx”.

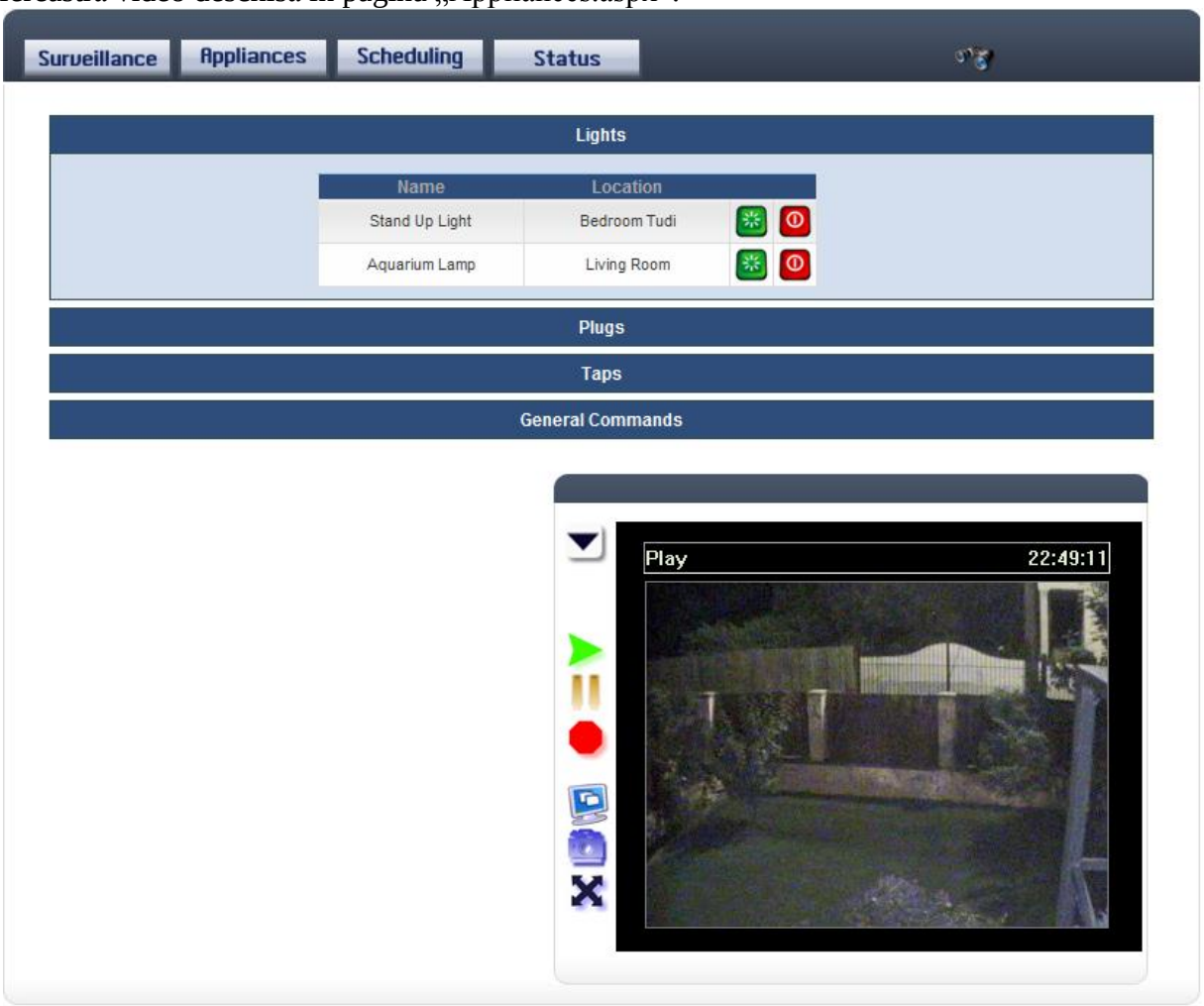


Figura 5.10 Pagina Appliances.aspx împreună cu fereastra video plutitoare

Pagina HomeSelection.aspx este prima pagină de pe care utilizatorii care au mai mult de o casă își selectează casa pe care vor să o controleze. Dacă un utilizator are mai mult de o casă acesta va fi redirecționat automat către pagina de Status.

Pagina Surveillance.aspx este pagina de supraveghere video. Conținutul acestei ferestre se încarcă dinamic în funcție de numărul de camere video definit în baza de date. Fereastra poate conține de la una până la patru instanțe ale controlului video CameraControl.ascx.

Pagina Appliances.aspx este pagina de control, de pe aceasta pagină se pot opri sau porni toate dispozitivele din sistem. Această pagină conține un control de tipul „ajaxToolkit:Accordion” cu patru ferestre componente: „Lights”, „Plugs”, „Taps” și „General Commands”. Primele trei tab-uri conțin cate o tablă cu controalele de tipul respectiv. În dreptul fiecăruia există câte un buton pentru oprire respectiv pornire. Ultimul tab conține butoane care acționează asupra grupurilor de controale: pornește toate luminile, oprește toate dispozitivele electrice.

Pagina Scheduling.aspx este pagina din care se adaugă programele automate. Această pagină conține un tabel cu programele actuale și un „ajaxToolkit:Accordion” cu două tab-uri. Unul pentru filtrarea task-urilor și altul pentru adăugarea lor.

Filtrarea se poate face după tipul, locația și numele controlului. Pentru a putea face filtrarea eficient, și pentru ca tot timpul să existe un rezultat am folosit controlul „ajaxToolkit:CascadingDropdown” ca și extender pentru dropdown-urile asp. Acestea nu permit selecția în al doilea dropdown până când selecția în primul nu este făcută. După ce se face prima selecție în al doilea se vor încarca doar variantele posibile.

În partea de adăugare a unui nou program am folosit și extender-ul „ajaxToolkit:NumericUpDown” care permite incrementarea utilizând mouseul a minutelor și orelor. Pe lângă acesta am folosit „ajaxToolkit:CalendarExtender” pentru a face posibilă selecția datei utilizând interfața vizuală.

Pagina Status.aspx afișează datele locației și temperaturile citite de senzorii de temperatură. De pe această pagină se poate accesa pagina de schimbare a parolei.

Pagina ChangePassword.aspx ajută la schimbarea parolei. Conține un câmp de verificare a parolei vechi și două pentru parola nouă și confirmarea ei.

5.3 Comunicarea între modulele sistemului

Aplicația conține multiple module fiecare aflate în locații diferite. Din acest motiv o comandă dată de client va trece prin mai multe noduri ale aplicației și prin multiple medii. De exemplu pentru închiderea unei electrovalve un semnal dat de utilizator va parcurge următoarea cale:

Browser Client ----(HTTPS)----> Aplicație Centrală ----(HTTPS)----> Aplicație „Remote” ----(RS232)----> Cerebot -> HB3 ----(Impuls ±9V)----> Electrovalvă

5.3.1 Interfața ->aplicație centrală

Interfața web a aplicației este parte componentă a aplicației centrale. Paginile cerute din interfața se încarcă în browser-ul clientului de pe serverul central. Comunicația între browser și server-ul central se face prin HTTPS cu criptare SSL. Acest lucru este important deoarece cheile de acces parcurg calea Client-Server necriptate.

5.3.2 Interfața -> sistem supraveghere

Pentru a nu se crea un „bottle neck” la serverul central, transferul video se face direct între browser-ul clientului și serverul video aflat în locația „remote”. Pentru ca acest lucru să fie posibil, browser-ul clientului citește locația server-ului remote și cheile de acces către acesta din baza de date după care le furnizează ca parametrii controlului ActiveX 8220 integrat în interfață. Streaming-ul video se face prin protocolul TCP pe portul definit de utilizator(default 5050).

5.3.3 Aplicație centrală -> aplicații remote

Comunicațiile între locația centrală și locațiile remote se fac prin protocolul HTTPS cu criptare SSL. Aplicația centrală comunică cu aplicațiile „remote” prin intermediul serviciilor web SOAP. Serviciile web de pe serverul remote au expusă o interfața WSDL. Aplicația centrală citește din baza de date locația server-ului remote și cheile de acces pentru acesta. După aceea se autentifică la serverul dorit. Un ticket de autentificare este creat. Acesta este trimis împreună cu fiecare apel de serviciu web. Un apel către oricare serviciu web(în afară de cel de autentificare) va fi respins dacă nu este însoțit de un ticket de autentificare.

5.3.4 *Aplicație remote -> X10*

Comunicația dintre aplicația „remote” și controllerul de tip X10, pentru dispozitivele electrice, CM11 este realizată prin portul serial, standardul RS232.

Conexiunea este deschisă în fișierul Global.asax pe evenimentul de pornire a aplicației „Application_Start”. Conexiunea astfel deschisă va fi folosită de toate procesele care doresc să comunice cu controllerul CM11 și va fi inchisă doar la închiderea totală a aplicației.

Conexiunea are următoarele caracteristici: utilizează portul definit în fișierul web.config sub numele de „cerebotPort”(ex: „com2”), baud 2400, 8 biți de date, un bit de stop, fără paritate. Serviciile web comunică cu dispozitivul prin metodele din librăria cm11.dll.

Obiectul de tip SerialPort generat este ținut pe sesiune sub numele de „X10Port”. Acesta va fi folosit de toate procesele care încearcă să dea comenzi dispozitivelor X10.

Dispozitivul CM11 accepta 16 comenzi, reprezentate pe 4 biți. Comenzile sunt prezentate în Tabel 3.2. Comenzile trebuie însoțite de codul casei și codul unității. Codurile caselor sunt reprezentate simbolic cu litere de la A la P iar codurile unitaților cu numere de la 1 la 16, în comenzi acestea sunt reprezentate binar pe câte 4 biți.

Între cele două dispozitive după fiecare comandă se transmit semnale de „handshacking”. Pentru a seta luminozitatea locației A1 la 16% se trimit următoarele comenzi:

- 0x04,0x66 PC-ul trimite Adresa A1
- 0x6a CM11 face suma elementelor trimise după care răspunde.
- 0x00 PC-ul trimite OK pentru transmisie
- 0x55 CM11 Interfața pregătită pentru primirea semnalelor
- 0x86,0x64 Funcția de întunecare cu 16%
- 0xea CM11 face suma elementelor trimise după care răspunde.
- 0x00 PC-ul trimite OK pentru transmisie
- 0x55 CM11 Interfața pregătita pentru primirea semnalelor

În cazul apariției unei erori, checksum greșit, se retransmite ultima comandă trimisă.

5.3.5 *Aplicație remote -> Cerebot*

Comunicația între aplicația aflată pe serverul de control din locația remote și placa Cerebot se face prin intermediul interfeței seriale, standardul RS232. Placa cu microcontroler Cerebot are atașat un modul PmodRS232 care este conectat printr-un cablu serial la portul serial al server-ului.

Conexiunea este deschisă în fișierul Global.asax pe evenimentul de pornire a aplicației „Application_Start”. Conexiunea astfel deschisă va fi folosită de toate procesele care doresc să comunice cu placa Cerebot și va fi închisă doar la închiderea totală a aplicației.

Conexiunea are următoarele caracteristici: utilizează portul definit în fișierul web.config sub numele de „cerebotPort”(ex: „com2”), baud 2400, 8 biți de date, un bit de stop, fără paritate.

Obiectul de tip SerialPort generat este ținut pe sesiune sub numele de „WaterPort”.

Pentru comunicația între cele două aplicații am stabilit următorul standard format din doi pași:

6. Serverul trimite: Comanda Tip_Comanda \n
7. Placa Cerebot răspunde cu un mesaj de succes sau de eșec
8. Serverul trimite: Comanda Port \n
9. Placa Cerebot răspunde cu un mesaj de succes sau de eșec

Tipurile de comenzi posibile sunt:

- "Q" - ping

- "V" – citește versiunea aplicației de pe placa cerebot
- "R" – resetează placa cerebot
- "E" – executare de comandă

Tipurile de comenzi posibile sunt:

- "WaterControl" – controlează electrovalve
- "Status" – citește statusul

Tipurile de comenzi pentru controlul robinetelor sunt:

- "O" – deschide electrovalva
- "C" – închide electrovalva

Tipurile de comenzi pentru status sunt:

- "T" – citește temperatura

Pentru a închide electrovalva al cărei pin de selecție are numărul 2 se va trimite următoarea succesiune de comenzi pe portul serial:

```
"E WaterControl \n"
"C 2 \n"
```

5.4 Structura bazei de date

Aplicația are o singură bază de date. Aceasta este situată în locația centrală și conține informații de autentificare și informații despre toate locațiile din sistem.

5.4.1 Descriere generală

În Figura 5.11 este reprezentată diagrama bazei de date. În baza de date se țin informații despre utilizatori (tabela *Logins*), locuințe (tabela *Houses*) și mapările dintre utilizatori și locuințe (tabela *LoginMappings*). Un utilizator poate avea multiple locuințe și aceeași locuință poate fi mapată la mai mulți utilizatori. Cheile de acces ale utilizatorilor nu se țin în baza de date ci doar id-uri criptate cu ajutorul acestor chei.

Codurile de acces la serviciile din fiecare casă sunt ținute pentru fiecare utilizator în tabela de legătura dintre utilizatori și locuințe (tabela *LoginMappings*). Acestea sunt criptate utilizând cheia publica a utilizatorului. Pentru ca serverul să aibă acces la sistemele de control din locuințe acesta are câte un set de chei de acces pentru fiecare locuință care sunt ținute criptate în tabela *ServerLogins*.

Fiecare casă are mai multe încăperi/zone, acestea sunt ținute în tabela *HouseLocations*. În fiecare locație pot exista mai multe dispozitive de control de tip X10 (lumini, prize). Acestea sunt stocate în tabela X10Controls. Dispozitivele de tip Cerebot (robinete, temperatură) sunt ținute în tabela CerebotControls. Tipurile dispozitivelor existente sunt stocate în tabela *ControlTypes*.

În tabela *Scheduling* sunt stocate comenzile care trebuie executate automat de către sistem.

5.4.2 Tabele

În continuare voi prezenta pe scurt tabelele împreună cu legăturile dintre ele. Voi folosi abrevierea PK pentru cheile primare, FK pentru cheile străine și AI pentru autoincrement. Baza de date conține următoarele table:

- **Logins** – *LoginID (uniqueidentifier)* PK, stochează:
  - datele de autentificare (*username*, *loginIDEncrypted*, *isAdmin*),
  - cheile RSA pentru criptarea și decriptarea detaliilor de autentificare la serverul remote (*publicKey*, *privateKeyEncrypted*)
  - detaliile despre utilizatori (*Email*, *FirstName*, *LastName*)

- **Houses** – HouseID (uniqueidentifier) PK, stochează:
  - numele locației (*name*)
  - poziționarea locației (*Country*, *County*, *City*, *Address*)
  - numărul de camere video din locație (*NumerOfCameras*)
  - locațiile serviciilor de supraveghere și control (*CameraServiceLocation*, *ControlServiceLocation*)
- **LoginMappings** – LoginID (uniqueidentifier), HouseID(uniqueidentifier) PK compus, LoginID (bigint AI) FK, HouseID FK, stochează:
  - datele de autentificare la serviciile de supraveghere și control aflate în locația remote (*CameraKeyEncypted*, *CameraUserNameEncrypted*, *ServerKeyEncrypted*, *ServerUserNameEncrypted*) toate cheile și numele de utilizator fiind criptate cu cheia publică a utilizatorului.
- **ServerLogins** – *ServerLoginID (int AI)* PK, *HouseID* FK, stochează:
  - datele de autentificare la serviciile de supraveghere și control aflate în locația remote pentru serviciul automat (*ServerKeyEncypted*, *ServerUserNameEncrypted*), acestea sunt criptate cu cheile serverului.
- **HouseLocations** – *LocationID (bigint AI)* PK, *HouseID* FK, stochează:
  - numele zonelor în care este împărțită locuință
- **ControlTypes** – *TypeID (smallint AI)* PK, stocheaza:
  - tipul controlului (*Type*)
  - interfața cu controllerul (*Target*)
- **X10Controls** – ControlID (bigint AI) PK, TypeID FK, LocationID FK, stochează:
  - numele controlului (*Name*)
  - locația în care se află controlul (*LocationID*)
  - datele de conectare la control (*X10HousID*, *X10UnitID*)
- **CerebotControls**– ControlID (bigint AI) PK, TypeID FK, LocationID FK, stochează:
  - numele controlului (*Name*)
  - locația în care se afla controlul (*LocationID*)
  - portul de conectare la control (*PortNo*)
- **Scheduling** – ScheduleID (bigint AI) PK, TypeID FK, stochează
  - ID-ul controlului țintă (*ControlID*), pe acesta nu s-a putut seta cheie străină deoarece poate face parte din două tabele (*X10Controls*, *CerebotControls*) în funcție de tipul controlului.
  - timpii de pornire și oprire (*turnOnTime*, *turnOffTime*)
  - starea activ/inactiv (*Active*)
  - frecvența de repetare (*Repeat*)

5.4.3 Vederi

Pentru a ușura accesul la date am folosit următoarele vederi:

- **CerebotControlsView** – vederea este compusă din tabelele *CerebotControls* și *ControlTypes* și înlesnește afișarea listei de controale întrucât numele tipului este prezent în vedere. Vederea are următoarele câmpuri: *HouseID*, *PortNo*, *Location*, *Type*, *Name*, *LocationID*, *TypeID*
- **X10ControlsView** – vederea este compusă din tabelele *X10Controls* și *ControlTypes* și înlesnește afișarea listei de controale X10 întrucât numele tipului este prezent în vedere. Vederea are următoarele câmpuri: *HouseID*, X10*HouseID*, *X10UnitID*, *Location*, *Type*, *Name*, *LocationID*, *TypeID*

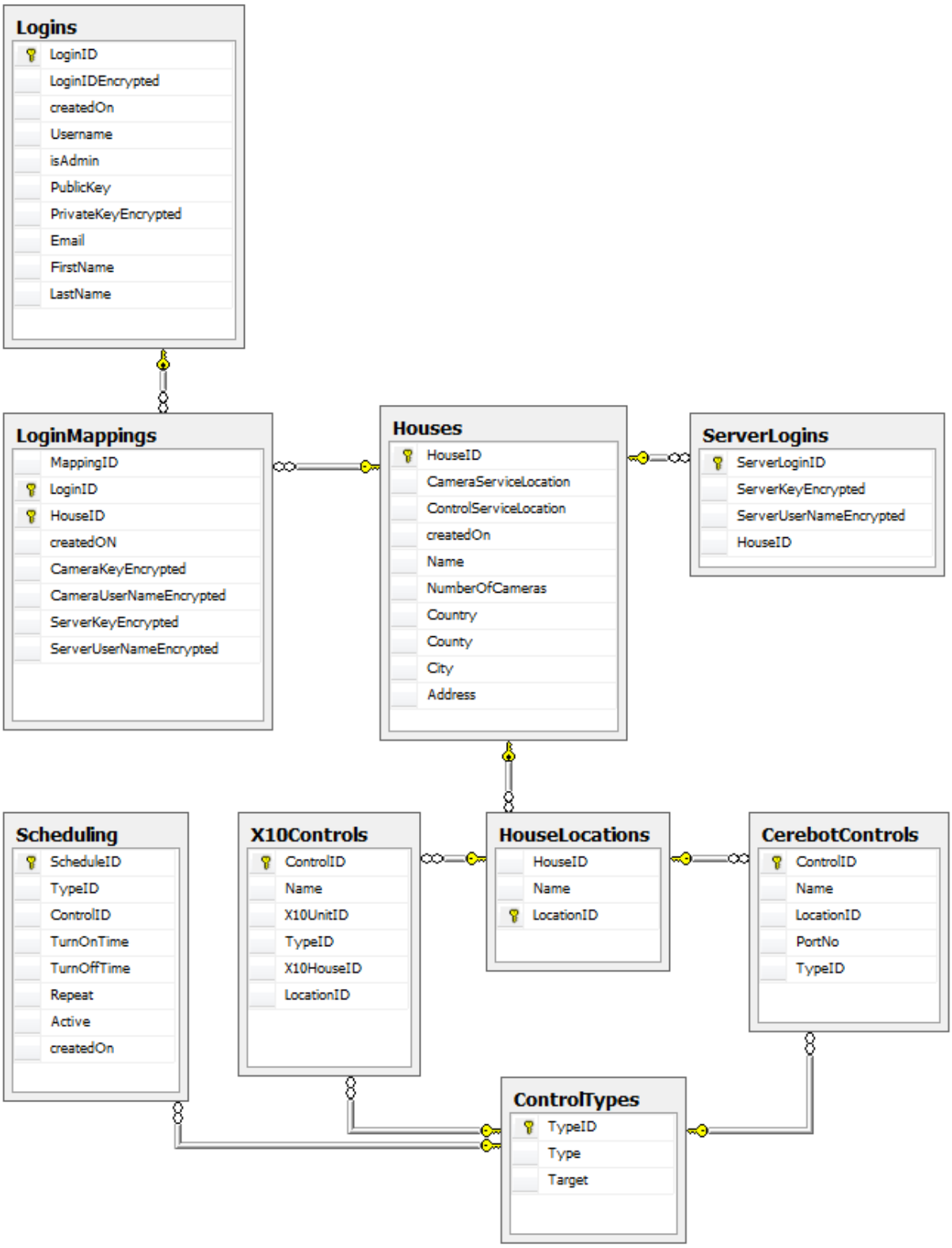


Figura 5.11 Diagrama bazei de date

- **SchedulingView** – vederea este compusă din tabelele *ControlTypes*, *CerebotControls*, *X10Controls*, *HouseLocations*, *Scheduling*. Această vedere conține toate taskurile din tabela scheduling împreună cu datele despre controalele pe care trebuie să le controleze. În funcție de tip caută ia controlul dintr-una din tabelele X10Control sau Cerebot Controls. Vederea are următoarele câmpuri:

ScheduleID, TypeID, ControlID, TurnOnTime, TurnOffTime, Repeat, Active, Type, Name, Location, HouseID

- **TaskExecutionView** – vederea are în componență tabelele *ControlTypes*, *CerebotControls*, *X10Controls*, *HouseLocations*, *Scheduling*, *Houses*, *ServerLogins*. Aceasta vedere conține toate taskurile din tabela scheduling care sunt active și au timpul de început sau de sfârșit egal cu minutul trecut. Vederea conține toate detaliile necesare pentru a putea porni sau opri un control care apare în task. Câmpurile vederii sunt:

5.4.4 Proceduri stocate

Pentru citirea datelor care sunt rezultatul unor reuniuni între tabele s-au folosit următoarele proceduri stocate:

- **spGetAllLocationsForHouseAndType** – procedura întoarce toate locațiile dintr-o locuință în care există cel puțin un control de tipul dat
  - date de intrare: *@typeID smallint, @houseID uniqueidentifier* reprezintă tipul controlului și id-ul casei în care se face căutarea
  - date de ieșire: tabela care conține *LocationID bigint, Name nvarchar(50)*
- **spGetAllTypesForHouse**– procedura primește întoarce toate tipurile de controale dintr-o locuință
  - date de intrare: *@houseID uniqueidentifier* reprezintă id-ul casei în care se face căutarea
  - date de ieșire: tabela care conține *TypeID tinyint, Type nvarchar(50)*
- **spGetNamesByLocationAndTypeID** – procedura întoarce toate controalele de un anumit tip dintr-o locație a unei locuințe
  - date de intrare: *@typeID smallint, @locationID bigint* reprezintă tipul controlului și id-ul zonei în care se face căutarea
  - date de ieșire: tabela care conține *ControlID bigint, Name nvarchar(50)*

Capitolul 6 Utilizarea sistemului

6.1 Echipamente necesare

6.1.1 *Clienți*

Pentru folosirea aplicației clientul are nevoie de:

- un sistem cu specificațiile minime:
  - Procesor: 1Ghz
  - RAM: 1 GB
  - HDD: 20 GB
  - Video: 128 MB
  - Altele: Ieșire Audio, Conexiune Internet
- monitor rezoluție minimă 1280 x 1024 pentru a putea vizualiza întreaga fereastră a aplicației

6.1.2 *Locația centrală*

Pentru instalarea aplicației centrale este nevoie de unul sau două servere. Serverul aplicației se poate desparți de serverul de baze de date, decizia aparține administratorului de sistem și este luată în funcție de numărul de clienți și de configurația deja existentă a serverelor.

Indiferent de configurație, se pot folosi unul sau două servere cu următoarea specificație:

- un sistem cu specificațiile minime:
  - Procesor: QuadCore 2.6Ghz HT
  - RAM: 4 GB
  - HDD: 1 TB
  - Video: 128 MB
  - Altele: Conexiune Internet
- 1 * UPS 2000VA
- un IP real

6.1.3 *Locații remote*

Pentru instalarea aplicației în locația remote (casa utilizatorului) este nevoie de unul sau două servere. Acest lucru este la latitudinea clientului, cele două aplicații (de control și supraveghere video) putându-se instala pe servere diferite sau pe același server.

Configurație cu 1 server:

- un sistem cu specificațiile minime:
  - Procesor: 3Ghz HT, 32-bit
  - RAM: 2 GB
  - HDD: 500 GB
  - Video: 256 MB DDR2
  - Altele: Conexiune Internet, porturi PCI (numărul depinde de numărul de plăci de captură conectate, placă audio), 2 porturi seriale sau 2 porturi USB pentru conectarea adaptoarelor USB->Serial
- 1 * UPS 2000VA
- un IP real

Configurație cu 2 servere:

- un sistem pentru partea de supraveghere video cu specificațiile minime:
  - Procesor: 3Ghz HT, 32-bit
  - RAM: 2 GB
  - HDD: 500 GB
  - Video: 256 MB DDR2
  - Altele: Conexiune Internet, porturi PCI(numărul depinde de numărul de plăci de captură conectate, placă audio)
- un sistem pentru partea de control
  - Procesor: 2Ghz, 32-bit
  - RAM: 2 GB
  - HDD: 20 GB
  - Video: 128 MB
  - Altele: Conexiune Internet, 2 porturi seriale sau 2 porturi USB pentru conectarea adaptoarelor USB->Serial
- 2 * UPS 1500VA
- Conexiune Internet 1 Mbps upload și două IP-uri reale

Pe lângă servere mai este nevoie și de echipamentul pentru supraveghere video și echipamentul pentru control.

Supraveghere video:

- placa de captură video Geovision aleasă în funcție de numărul de camere video care trebuie conectate. Modelele care se pot folosi sunt următoarele: GV-2008, GV-1480, GV-1240, GV-1120, GV-800, GV650, GV-600
- camere de supraveghere video (max. 32 bucăți)
- Control prize/lumini(se va folosi un transmițător și un număr de maxim 256 module auxiliare)
- transmițător X10 model CM11 sau CM15
- module control lumini LM12, LW11, AW10, SW10, LM12W, LM15, etc.
- module control prize AM12, AM12W
- module control siguranțe AD10D

Control robinete, verificare temperatură

- 1*placa microcontroler Cerebot(Digilent)
- Module PmodTMP(Digilent) pentru citirea temperaturii(max. 5 bucăți)
- Module PmodHB3(Digilent) pentru controlul robinetelor(max. 7 bucăți)
- Electrovalva Gardena(max. 7 bucăți)

6.2 Software necesar

6.2.1 Clienți

Pentru folosirea aplicației pe sistemul clientului trebuie să ruleze un sistem de operare Microsoft Windows Vista, Microsoft Windows XP SP3 sau Microsoft Windows Server 2008. Sistemul de operare poate rula atât pe 64 de biți cât și pe 32 de biți.

Pentru deschiderea aplicației clientul va folosi browser-ul Internet Explorer 8.

6.2.2 Locație centrală

În locația centrală serverele trebuie să aibă instalat un sistem de operare Windows Server 2008 sau Windows Server 2003. Sistemul de operare poate rula atât pe 64 de biți cât și pe 32 de biți. Pe serverul de baze de date trebuie să fie instalat SQL Server 2008.

6.2.3 *Locații remote*

În locația remote software-ul necesar depinde de numărul de servere folosite.
Configurație cu 1 server:

- Serverul de supraveghere video și control
  - Sistem de Operare: Windows XP, Windows Vista sau Windows Server 2008 pe 32 de biți.
  - Softul de captură video Geovision V8.2 sau V8.3

Configurație cu 2 servere:

- Serverul de supraveghere video
  - Sistem de Operare: Windows XP, Windows Vista sau Windows Server 2008 pe 32 de biți.
  - Softul de captură video Geovision V8.2 sau V8.3
- Serverul de control
  - Sistem de Operare: Windows XP, Windows Vista sau Windows Server 2008 pe 32 sau 64 de biți

6.3 Utilizare

În următoarele subcapitole voi descrie modul de utilizare a interfeței web a sistemului, mai exact modul de adăugare a unor noi clienți și locații (administrare) și modul în care un client utilizează interfața pentru a-și controla locuința.

Se deschide un browser web după care se va naviga la adresa https://www.smart-homes.ro . Se va deschide pagina de autentificare (Figura 6.1) unde trebuie introduse numele de utilizator și parola după care se apasă „Log In”.

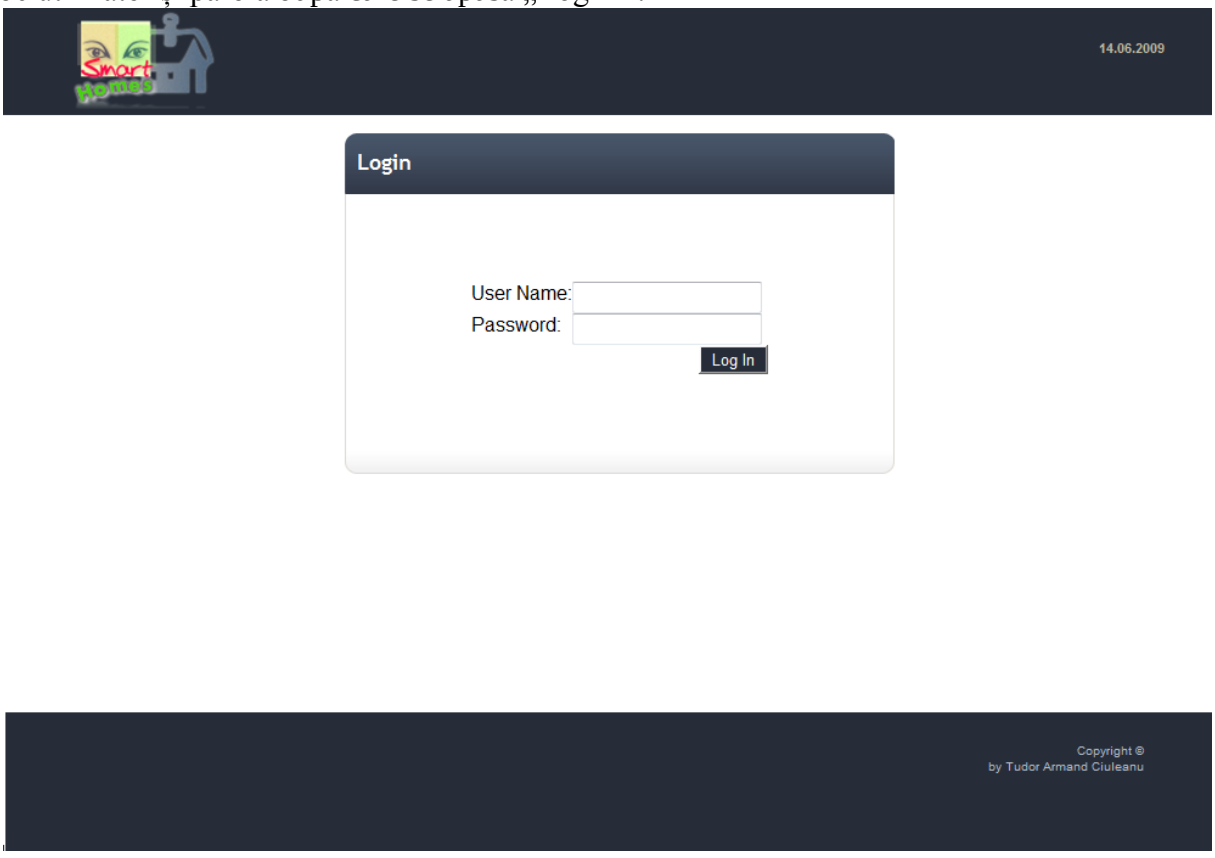


Figura 6.1 Pagina Login.aspx

6.3.1 *Administrare*

Administratorul are dreptul de a adăuga și edita utilizatori, locații și detaliile lor. După autentificare se face redirecționarea la pagina de utilizatori (Figura 6.2). Pe aceasta pagină în tabela „Logins" sunt afișați utilizatorii sistemului. Administratorul nu are dreptul să vizualizeze sau să schimbe parola unui utilizator.

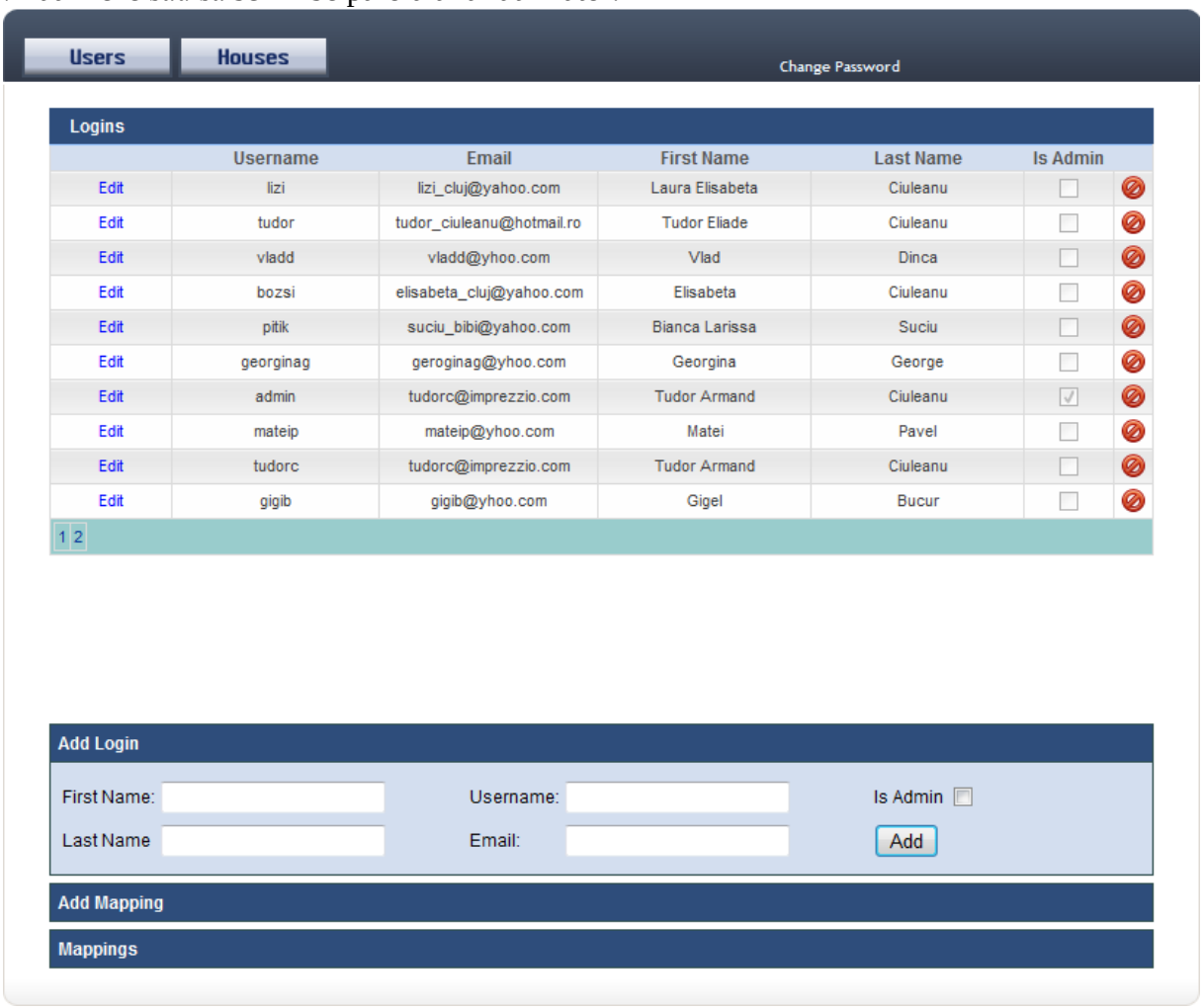


Figura 6.2 Pagina Users.aspx

Pentru a edita datele unui utilizator se va apasă butonul „edit" din linia utilizatorului. După terminare se va apasă butonul „Update". Pentru ștergerea unui utilizator se va apasă butonul  .

Pentru adăugarea unui nou utilizator se va deschide tab-ul Add Login și se vor completa câmpurile după care se va apasă butonul „Add". Toate câmpurile sunt obligatorii, numele de utilizator „username" trebuie să fie unic. În caz contrar un mesaj de eroare va fi afișat și utilizatorul nu va fi adăugat.

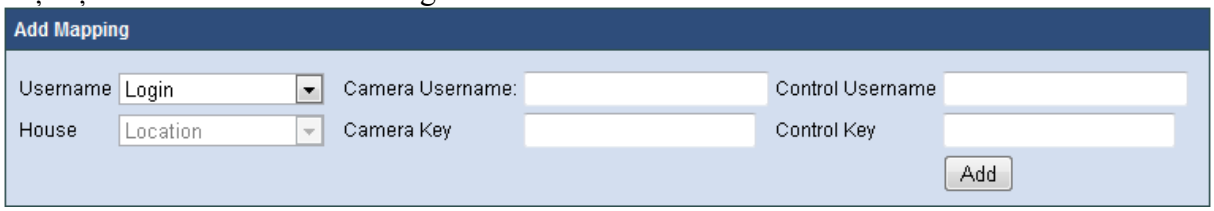


Figura 6.3 Tab-ul de adăugare a mapărilor

Pentru a mapa un utilizator la o locație se va deschide tabul „Add Mappings"(Figura 6.3). Utilizatorul poate fi mapat la orice locație la care nu a mai fost mapat anterior.

Pentru a vedea și șterge mapările utilizatorilor se va deschide tabul „Mappings”(Figura 6.4). Mapările nu pot fi editate ci doar șterse.

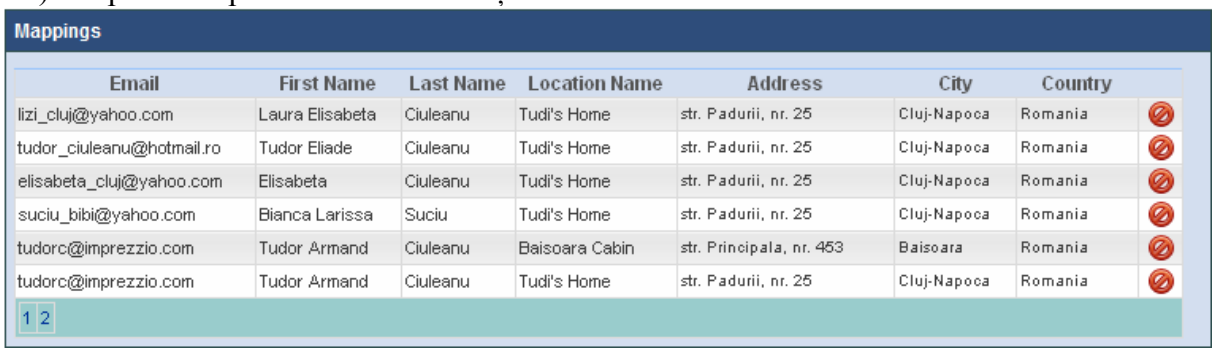


Figura 6.4 Tab-ul de vizualizare a mapărilor

Apasărea link-ului „Houses” deschide pagina de vizualizare / adăugare de locuințe prezentată în Figura 6.5. Această pagină este structurată similar cu cea anterioară. În partea superioară a paginii este un tabel care conține toate casele din sistem și câteva detalii generale despre ele(nume, adresă, oraș, județ, țară, număr de camere video, locația serviciului de camere video și locația serviciului de control). Simbolul trebuie apasăt pentru a vizualiza/edita sau adăuga dispozitive controlabile din locația selectată.

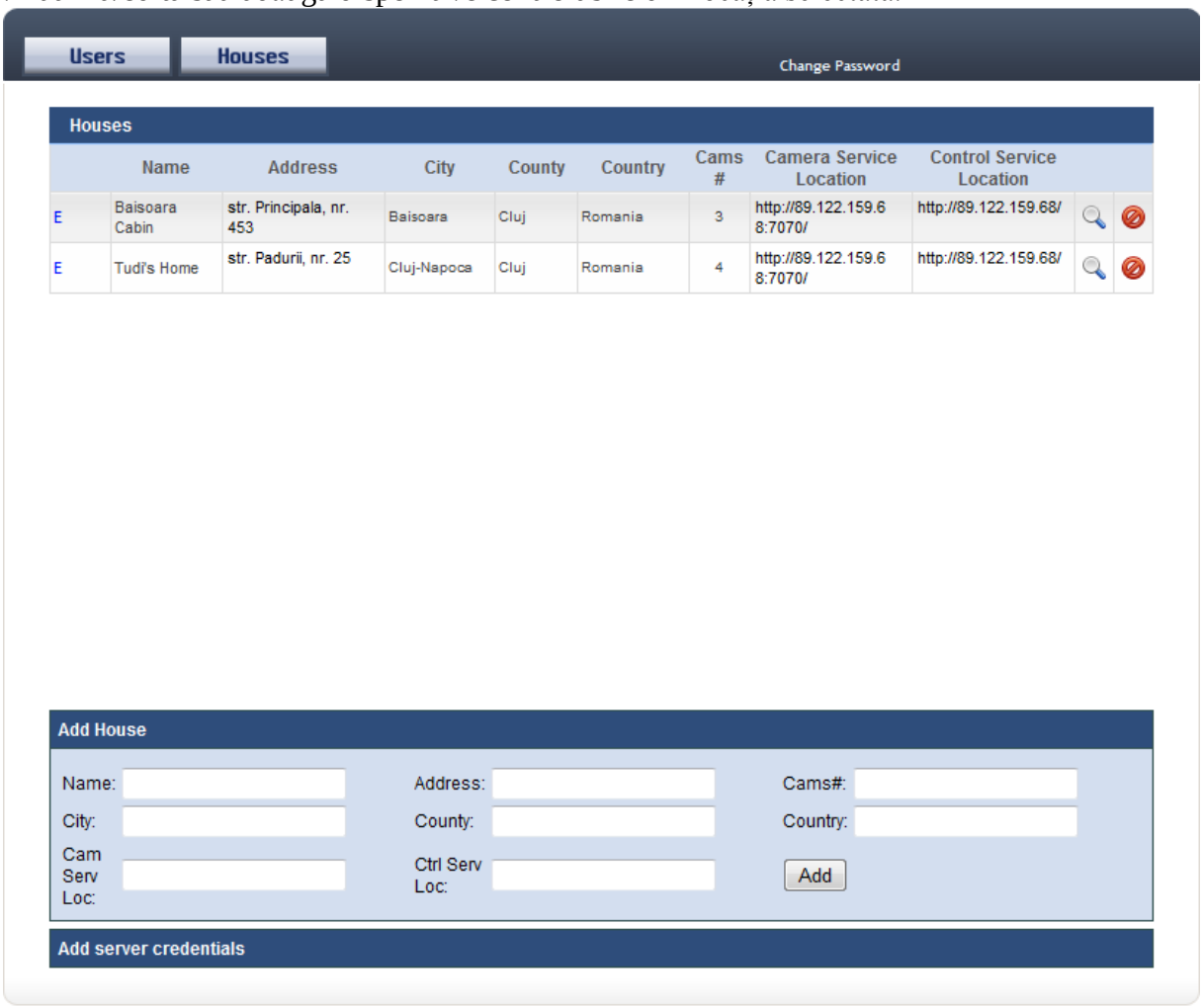


Figura 6.5 Pagina Houses.aspx

Pentru a adăuga o nouă clădire la sistem trebuie completate câmpurile din tab-ul locations. În cazul în care unul dintre câmpuri este completat greșit va apărea un mesaj care

anunță acest lucru. Dacă adăugarea se face cu succes câmpurile vor fi șterse și înregistrarea va apărea în tabelul „Houses”.

Tab-ul „Add server credentials” (Figura 6.6) se folosește pentru adăugarea unui nume de utilizator și a unei chei cu care programele automate se pot autentifica pe serverul „remote”.

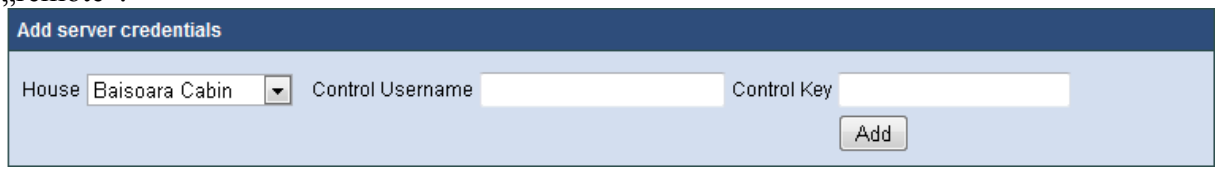


Figura 6.6 Tab-ul Add server credentials

După adăugare noua locație apare în tabela Houses. Se va apasă pentru a naviga la pagina „HouseDetails” de unde se adaugă noi dispozitive pentru clădirea curentă.

Pagina „HouseDetails” (Figura 6.7) conține trei ferestre care se cascadează, „Locations”, „X10Controls” și „Other Controls”.

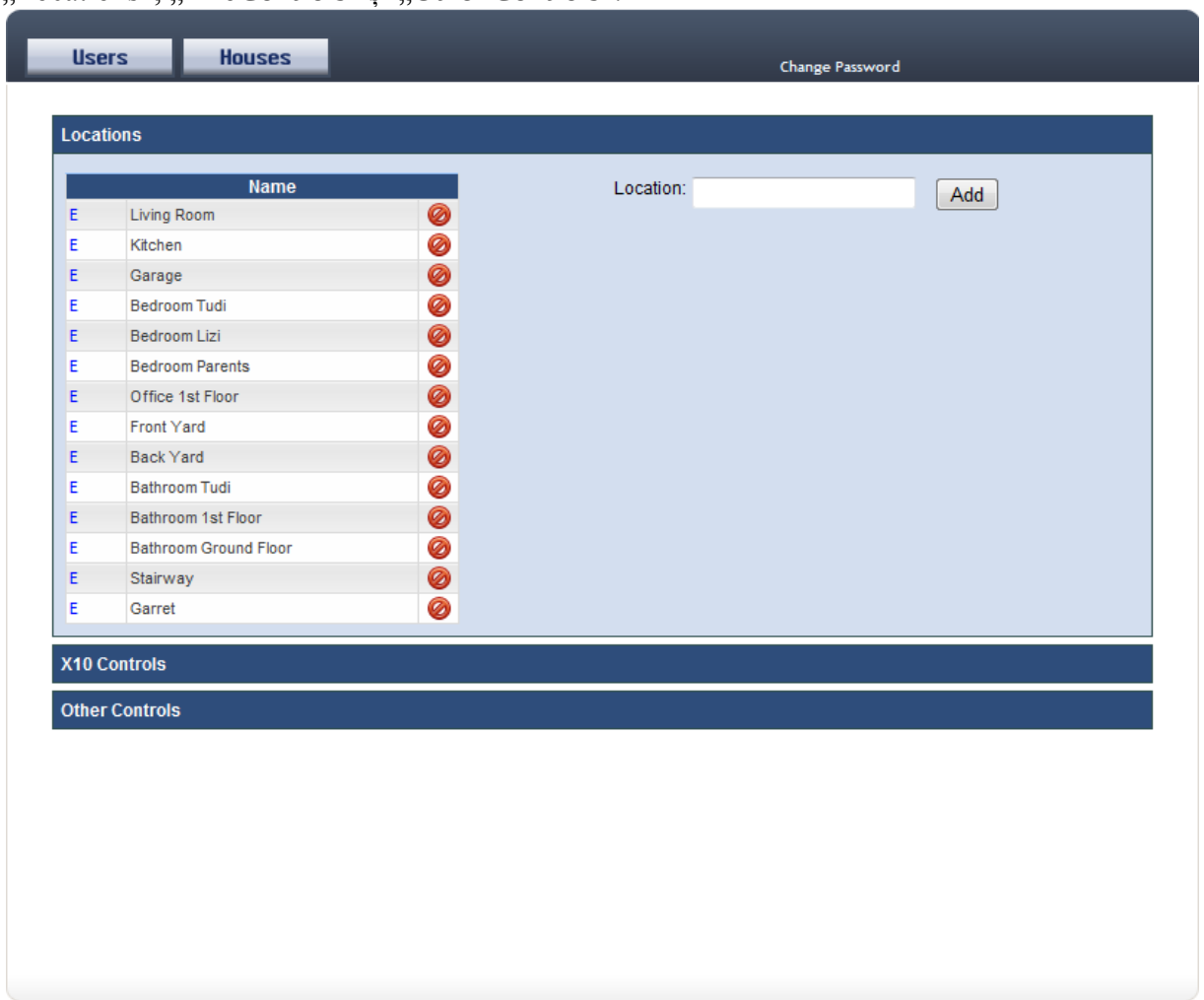


Figura 6.7 Pagina HouseDetails.aspx

Din tab-ul „Locations” se adaugă/editează/șterg zone dintr-o locuință. Pentru adăugarea unei zone se va completa numele zonei în câmpul „Location” și se va apasă butonul Add. Câmpul nu poate fi vid.

Se va deschide tab-ul X10Controls (Figura 6.8) pentru a adăuga/edita/șterge un dispozitiv de control al instalației electrice. Pentru un astfel de dispozitiv trebuie selectat tipul, numele, locația, id-ul caseiX10 (litera de la A la P) și id-ul unității X10 (număr de la 1 la 16).

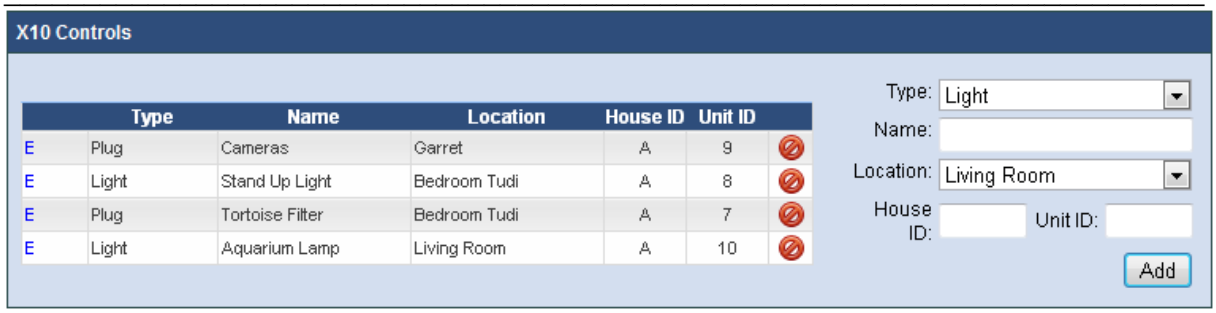


Figura 6.8 Tab-ul cu dispozitivele de tip X10

Se va deschide tab-ul OtherControls(Figura 6.9) pentru a adăuga/edita/șterge un dispozitiv de control conectat la placa de control Cerebot. Câmpurile trebuie introduse corect pentru a trece de validare.

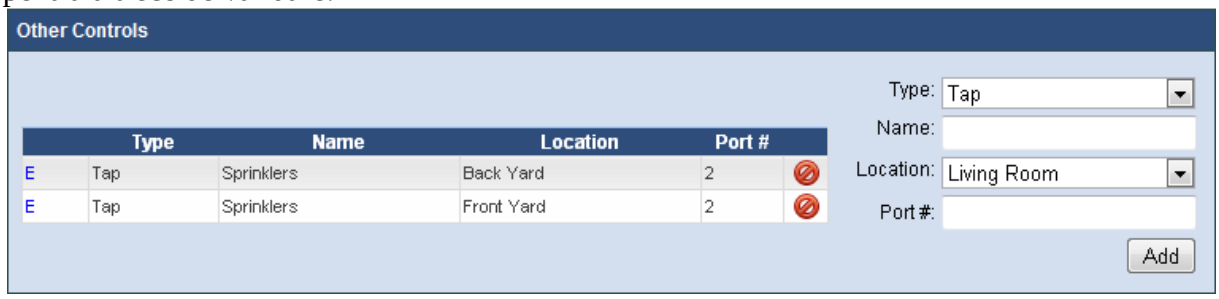


Figura 6.9 Tab-ul cu alte tipuri de dispozitive

6.3.2 Utilizare client

După autentificare dacă utilizatorul are o casă în sistem va fi redirecționat spre pagina de status, dacă are mai mult de o casă va fi redirecționat la pagina de selecție a casei. După selectația casei care va fi controlată se va face redirecționare la pagina Status. În Figura 6.10 este prezentată o captură a părții superioare a paginii de selecție a locuinței care urmează a fi controlată.

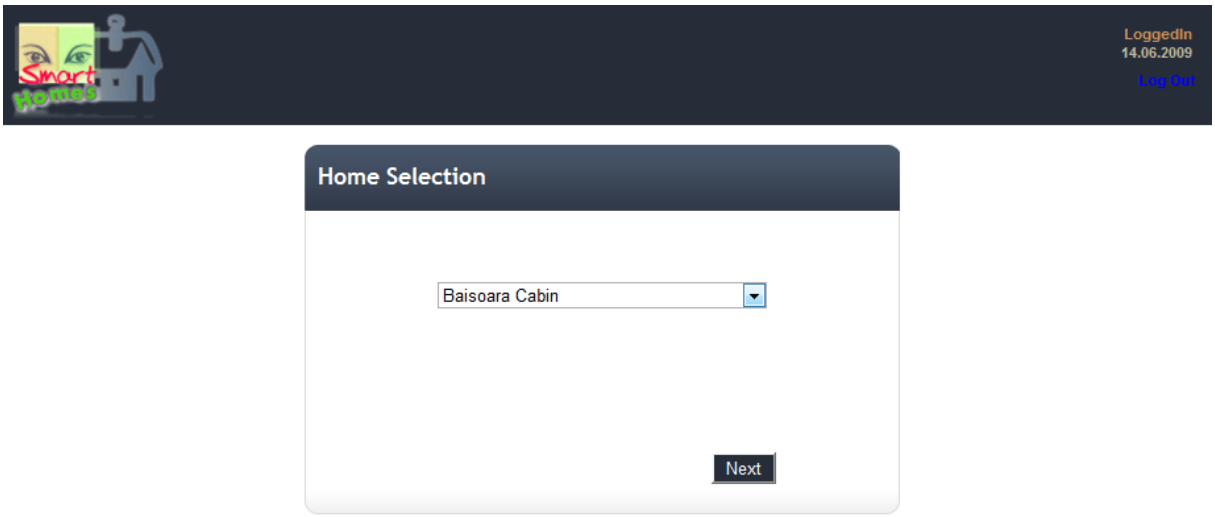
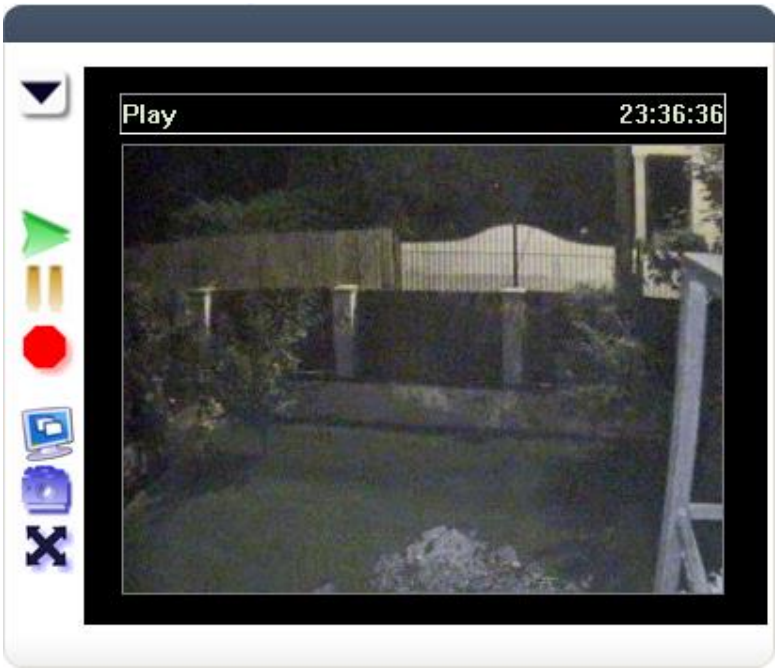


Figura 6.10 Pagina HomeSelection.aspx

Utilizatorii pot naviga între paginile „Surveillance”, „Appliances”, „Scheduling” și „Status” folosind butoanele din partea de sus a ecranului. Pentru a ieși din aplicație se va apasă butonul „Log Out” din coltul dreapta sus al ferestrei.

Indiferent de pagina curentă, prin apasărea butonului din dreapta butoanelor de navigare  (binoclu) se va deschide o fereastră „plutitoare”. Această fereastră (Figura 6.11) conține un control video cu ajutorul căruia se vizualizează imaginile de la una din camerele video din locație.



Comenzile ferestrei sunt:

–selectie camera

–pornire

–oprire

– pauza

–schimbare rezoluție

–snapshot

–fullscreen

Figura 6.11 Fereastra video plutitoare

După autentificare se face redirecționarea pe pagina „Status”(Figura Status) unde se poate observa temperatura citită de senzorii de temperatură.

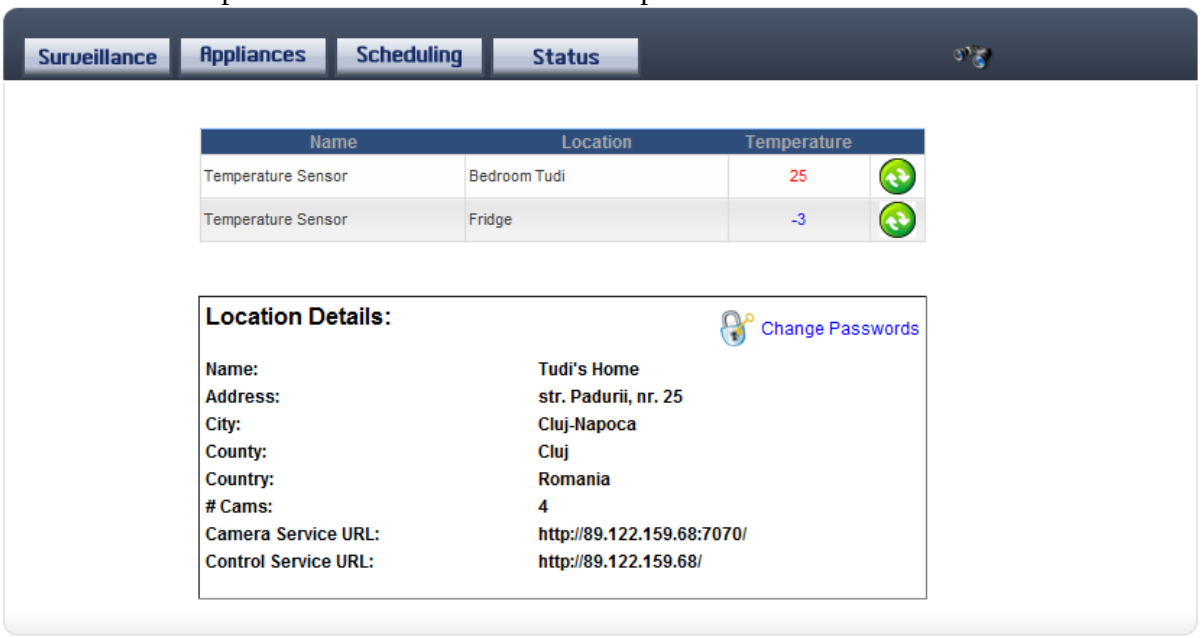


Figura 6.12 Pagina Status.aspx

Pe lângă tabelul cu temperaturi pe această pagină sunt afișate și detaliile locației curente. Temperaturile pozitive sunt afișate cu roșu iar cele negative cu albastru. Prin apasărea butonului  se reactualizează temperaturile. Pentru a schimba cheile de acces la aplicație și

cheile de autentificare la serverul de control și cel video se apasă butonul  „Change Passwords”.

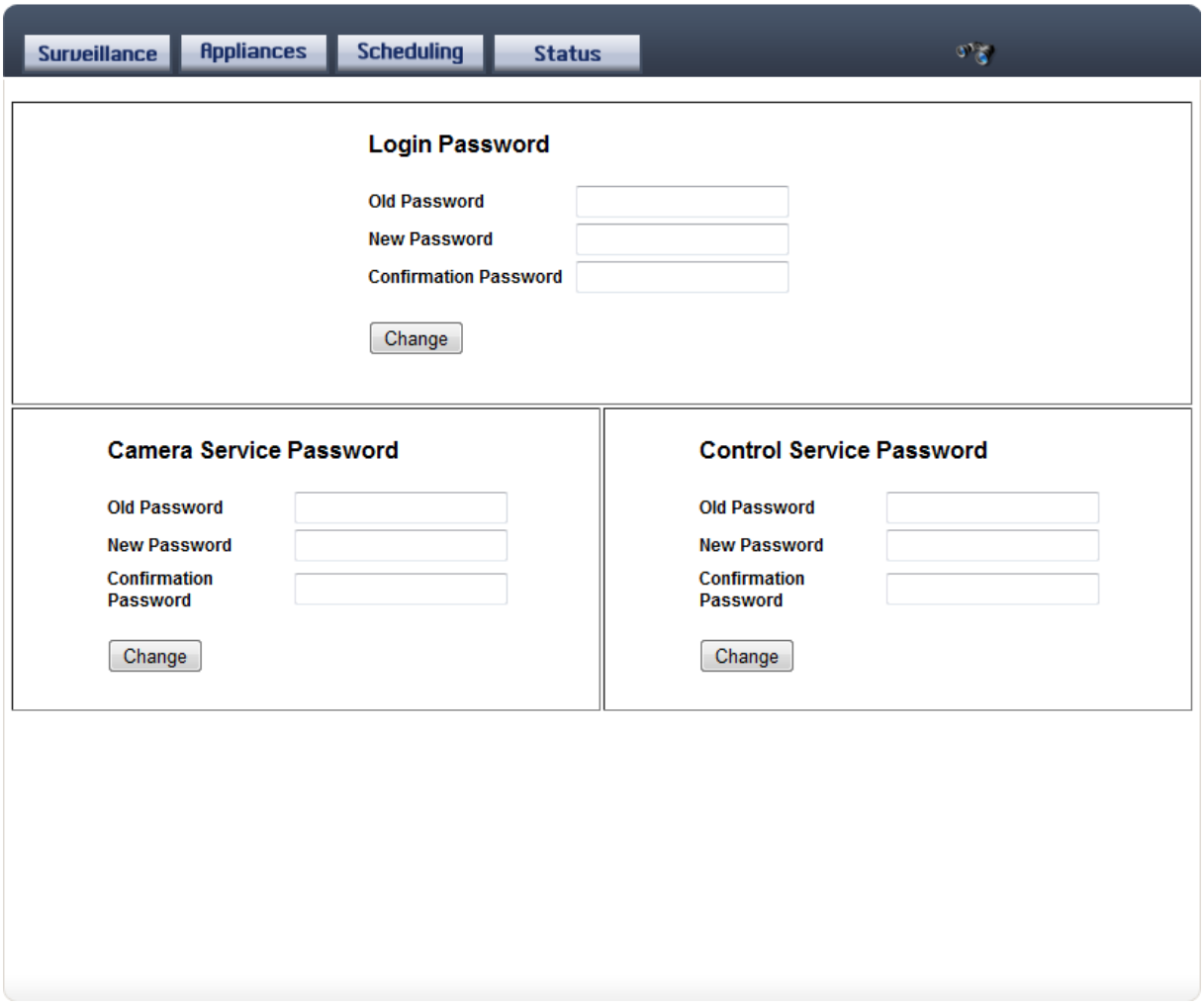


Figura 6.13 Pagina ChangePassword.aspx

În Figura 6.13 este prezentată o captură a ecranului de schimbare a cheilor de acces. Din această pagină se poate schimba cheia de acces la aplicație și cheile de acces la serviciul de supraveghere video și serviciul de control. În cazul în care nu există drepturi de a urmări camerele de supraveghere sau de a controla dispozitive din locație, nu se vor putea schimba cheile de acces la aceste servicii.

Pentru a urmări mai multe camere de supraveghere simultan se apasă butonul „Surveillance” și se va face redirecționarea către pagina de supraveghere. În funcție de numărul de camere video din locație în această fereastră vor apărea de la 1 pâna la 4 ferestre cu funcționalitate similară celei prezentate în Figura 6.11. În Figura 6.14 este prezentată pagina Surevillance.aspx pentru o locație cu 3 camere video. În cazul în care în sistem sunt mai mult de 4 camere video se va folosi butonul de selecție pentru a schimba camera de la care se urmaresc imagini. Fereastra video plutitoare poate fi folosită și în această pagină.

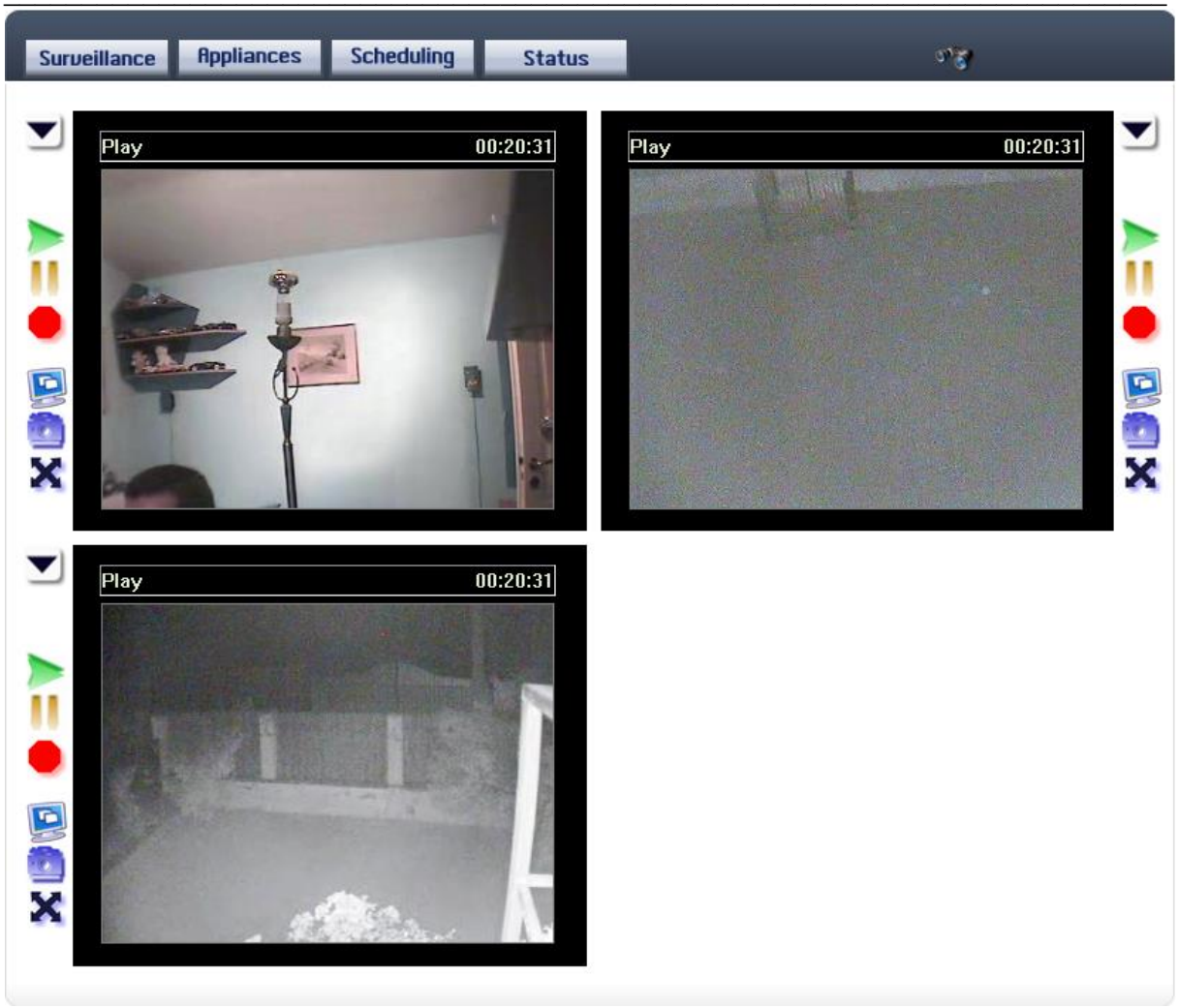


Figura 6.14 Pagina Surveillance.aspx

Pentru a porni sau opri anumite dispozitive din locuință se apasă butonul appliances. În Figura 6.15 se poate observa această pagină cu dispozitive („Appliances.aspx) cu tab-ul de prize deschis.

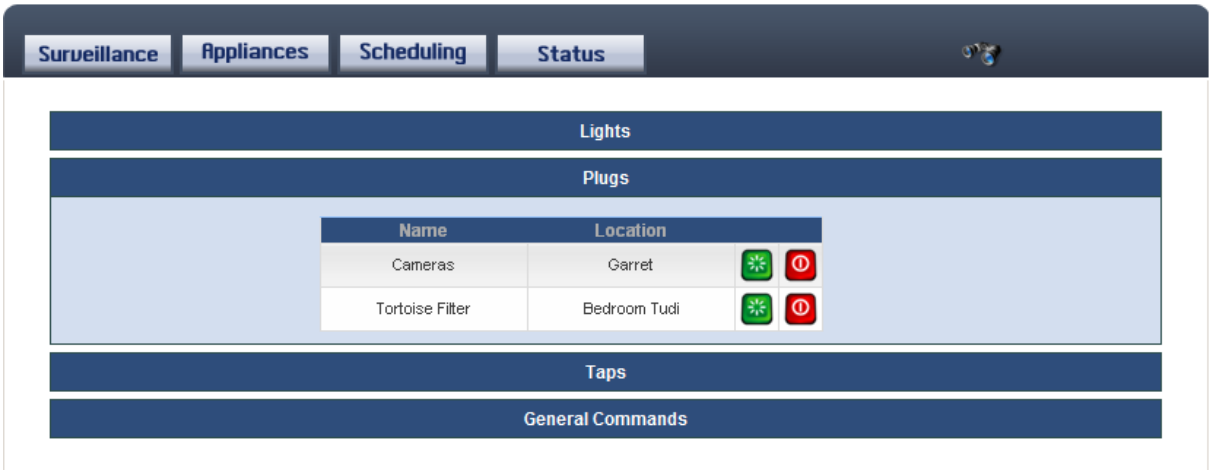


Figura 6.15 Partea superioară a paginii Appliances.aspx

În această fereastră se poate alege între 4 tab-uri care se cascadează:

- Lights – în acest tab se află un tabel cu toate dispozitivele de iluminat conectate la sistemul din locație.

- Plugs – în acest tab se află un tabel cu toate prizele conectate la sistemul din locație.
- Taps – în acest tab se află un tabel cu toate electrovalvele conectate la sistemul din locație.
- General Commands – în acest tab se află un tabel cu toate dispozitivele de iluminat conectate la sistemul din locație.

Toate tabelele de control din această fereastră au un aspect și comportament similar. Câmpurile tabelelor sunt „Name” și „Location” care reprezintă numele și zona în care se află controlul. Pentru a porni un dispozitiv dintr-o tabelă se apasă butonul  iar pentru a-l opri se apasă butonul  . Ultimul tab „Other commands” este diferit de cele anterioare, în acesta aflându-se comenzi generale de oprire sau pornire a tuturor dispozitivelor de un anumit tip.

Pentru adăugarea unui program automat (pornirea/oprirea unor dispozitive la anumite ore) se va apasă butonul „Scheduling”. După apasărea acestui buton se va deschide pagina de programe automate (Figura 6.16)

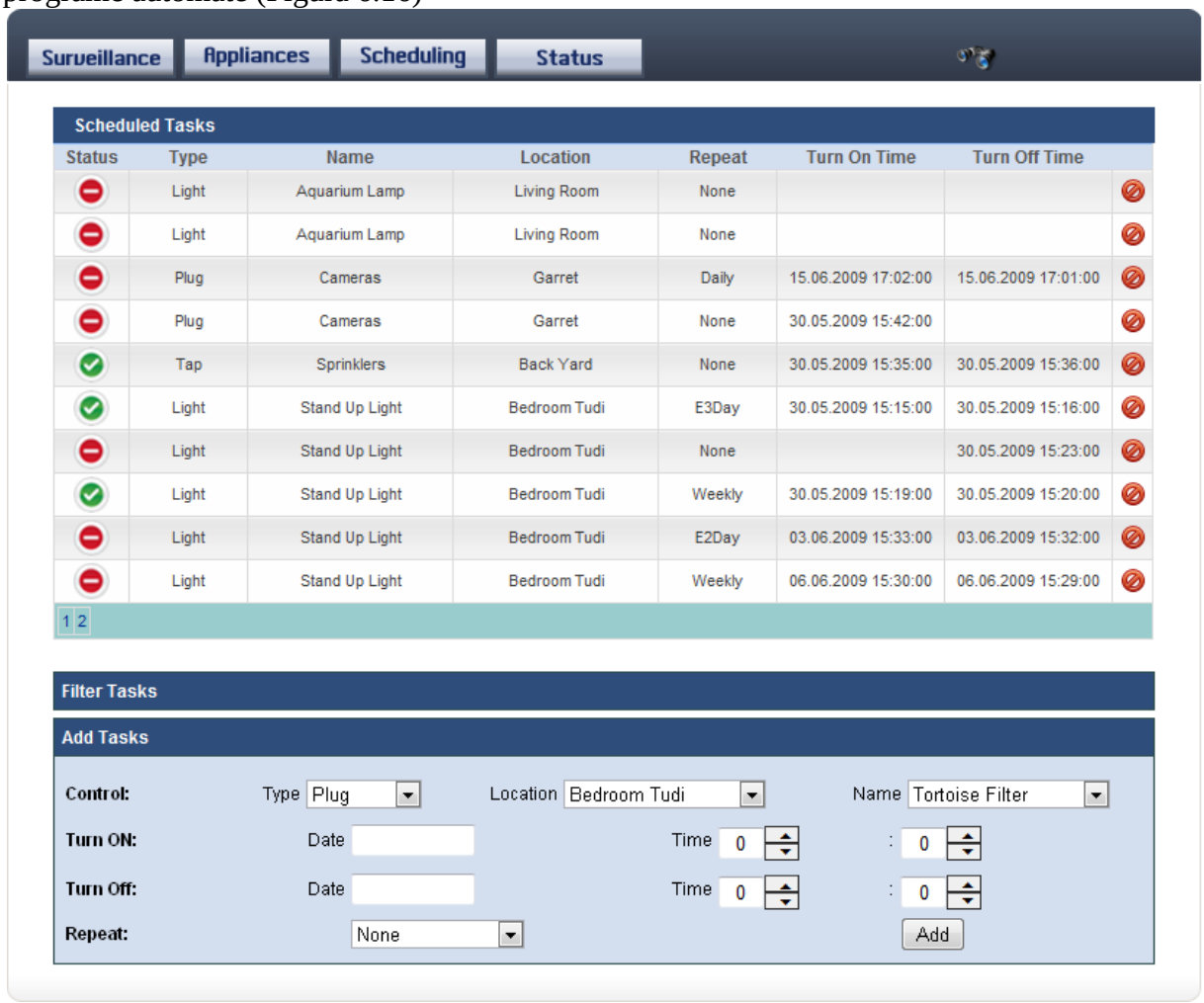


Figura 6.16 Pagina Scheduling.aspx

În această pagină în tabelul superior sunt afișate toate programele active și inactive. Cele active sunt simbolizate prin  iar cele inactive prin  . În acest tabel se afișează tipul controlului, numele, zona, intervalul de repetiție, ora pornirii și ora opririi. Pentru a activa un program se va apasă pe simbolul care arată că este inactiv iar acestuia i se va schimba starea. Pentru a șterge un program se va apasă pe simbolul  .

În partea de jos a ferestrei se află două ferestre care se cascadează. Una dintre ele conține un set de filtre. Aceste filtre pot vor fi setate după care se apăsă butonul filter pentru a se face o afișare selectivă în tabelul superior. Cea de-a doua fereastră „Add Tasks” este folosită pentru adăugarea de noi programe automate. Pentru un program automat se poate seta ora de pornire, ora de oprire sau ambele. Aceste programe pot fi ciclice, pentru aceasta se va seta câmpul repeat la un interval care variază de la o zi la o săptămână cu increment de o zi.

6.4 Condiții de funcționare

Aplicația Smart-Homes are mai multe module componente care se află în multiple locații. Pentru ca un utilizator să poată să acceseze propria casă, module din 3 locații diferite trebuie să funcționeze în același timp. Dacă unul din aceste module nu va funcționa, clientul nu va reuși să își acceseze locuința. Pentru a diminua șansele ca acest lucru să se întâmple la toate locațiile trebuie folosit câte un UPS de capacitate mare. Providerul de Internet trebuie ales în așa fel încât conexiunea să fie stabilă și destul de rapidă.

Unele componente, expuse la intemperii, precum senzorii de temperatură sau camerele video pot înceta să mai funcționeze dacă condițiile devin extreme.

Capitolul 7 Punerea în funcţiune şi rezultate experimentale

7.1 Tehnologia folosită

Pentru implementarea aplicației am folosit mediile de dezvoltare Microsoft Visual Studio 2008 SP1 pentru partea de PC iar pentru partea de microcontroler am folosit AVR Studio 4. Baza de date a fost creată utilizând Microsoft SQL Server 2008 Management Studio.

Pentru implementare am utilizat framework-ul .net 3.5 SP1. Pentru interfața web pe lângă componentele asp, am utilizat și componente din AjaxControlToolkit. Funcționalitatea care se execută client-side a interfeței am implementat-o utilizând JavaScript.

Comunicarea între aplicațiile „remote” și aplicația centrală se face prin intermediul serviciilor web SOAP care au expusă o interfață WSDL. Comunicarea între modulele de comandă Cerebot,X10 și aplicația „remote” se face utilizând standardul RS232.

Imaginile cu reprezentarea semnalelor de control pentru electrovalve au fost captate cu ajutorul unui osciloscop Tektronics TPS 2024.

Pentru scrierea documentației am utilizat Microsoft Office 2007. Diagramele au fost create utilizând Microsoft Office Visio 2007.

7.2 Distribuția aplicației

În locația centrală aplicația trebuie instalată o singură dată, la adăugarea unei noi locuințe în sistem aplicația „remote” trebuie instalată în locația respectivă.

7.2.1 Locația centrală

Pentru punerea în funcțiune a aplicației centrale trebuie executați următorii pași:
Pe serverul de baze de date se vor executa următorii pași:

1. Se va crea o bază de date „SmartHomes” în SQL Server 2008
2. Se va crea un utilizator cu drepturi de scriere și citire pentru baza de date „SmartHomes”, acestui utilizator nu i se vor da drepturi de administrator.
3. Se va da restore în baza de date „SmartHomes” backup-ului „SmartHomes.bak”

Pe serverul aplicației se vor executa următorii pași:

1. Se va crea un director SmartHomes(ex: D:\Websites\SmartHomes)
2. Se va copia aplicația SmartHomes în acest director
3. Se va crea un director virtual în IIS cu numele SmartHomes
4. Se va seta calea directorului virtual către directorul SmartHomes creat la pasul 1
5. Se va activa autentificarea de tip forms și se vor dezactiva celelalte tipuri de autentificări pentru directorul virtual actual
6. Se va schimba Application Pool-ul din Default AppPool în Clasic .net AppPool
7. Se va adăuga un certificat SSL acestui site și se va șterge portul *80. Doar conexiunile HTTPS sunt permise
8. Se va deschide fișierul Web.config din directorul SmartHomes și se vor seta următorii parametrii:
    - WAConnectionString – trebuie setate userul, parola pentru baza de date și url-ul pentru serverul de baze de date
    - SmtpServer – serverul de mail
    - FromEmailAddress – adresa de mail de la care se va primi e-mailul cu parola

- NetworkCredentialUsername – nume de utilizator pentru autentificare la serverul de mail
- NetworkCredentialPassword – parola pentru serverul de mail

7.2.2 Locație „remote”

Configurația plăcii Cerebot:

1. Se va conecta placa Cerebot prin cablul JTAGusb la computer
2. Se va porni aplicația AVRProgrammer și se va programa fișierul CerebotControlApp.hex pe placa Cerebot
3. Se va deconecta cablul JTAG
4. Se va conecta modulul RS232 la portul E al plăcii Cerebot
5. Se vor conecta modulele PmodTMP la portul C.
   - pinul 1 al fiecărui PmodTMP la pinul 1 al Portului C
   - pinul 2 al fiecărui PmodTMP la pinul 2 al Portului C
   - pinul 4 al fiecărui PmodTMP la unul din pinii 3-8 al Portului C
6. Se vor conecta modulele PmodHB3 la portul D.
   - pinul 2 al fiecărui PmodHB3 la unul din pinii 2-8 al Portului D
   - pinul 1 al fiecărui PmodHB3 la pinul 1 al Portului D

Pe serverul aplicației remote se vor efectua următorii pași:

1. Se va crea un director SmartHomes(ex: D:\HomeControl\Services)
2. Se va copia aplicația Remote SmartHomesRA în acest director
3. Se va crea un director virtual în IIS cu numele SmartHomesRA
4. Se va seta calea directorului virtual către directorul Services creat la pasul 1
5. Se va activa autentificarea de tip forms și se vor dezactiva celelalte tipuri de autentificări pentru directorul virtual actual
6. Se va schimba Application Pool-ul din Default AppPool în Clasic .net AppPool
7. Se va adauga un certificat SSL acestui site și se va șterge portul *80. Doar conexiunile HTTPS sunt permise
8. Se va deschide fișierul Web.config din directorul SmartHomes și se vor seta următorii parametrii:
   - Credentials – trebuie setat sub forma “username;password”
   - cerebotPort – trebuie setat numarul portului serial la care este conectată placa Cerebot
   - x10Port – trebuie setat numărul portului serial la care este conect CM11
9. Se va conecta un cablu serial la modulul CM11
10. Se va conecta un cablu serial la sistemul Pmod-ul RS232
11. Se va alimenta placa Cerebot.

7.3 Rezultate experimentale

O folosire tipică a aplicației constă din trei etape importante: adăugarea fizică a dispozitivelor în clădire (instalarea fizică), adăugarea clădirii și a dispozitivelor din aceasta în aplicație prin intermediul interfeței de administrare (instalare virtuala) și folosirea interfeței de acces, de către utilizatorul final. Primii doi pași sunt executați de firma deținătoare a aplicației, ultimul pas revine clientului (utilizatorului final) și se poate repeta de oricâte ori.

Instalarea fizică se face respectând pașii și constrângerile din capitolul anterior, 7.2 Distribuția aplicației. După instalarea fizică un administrator adaugă noua locație, detaliile acesteia, un nou utilizator și mapează clădirea la utilizator.

Utilizatorul folosește cheile de acces primite pentru intrarea în sistem. După verificarea validității cheilor utilizatorul va fi redirecționat către pagina „Status”. Pe această

pagină este afișată temperatura citită de senzorul de temperatură (Figura 3.1).

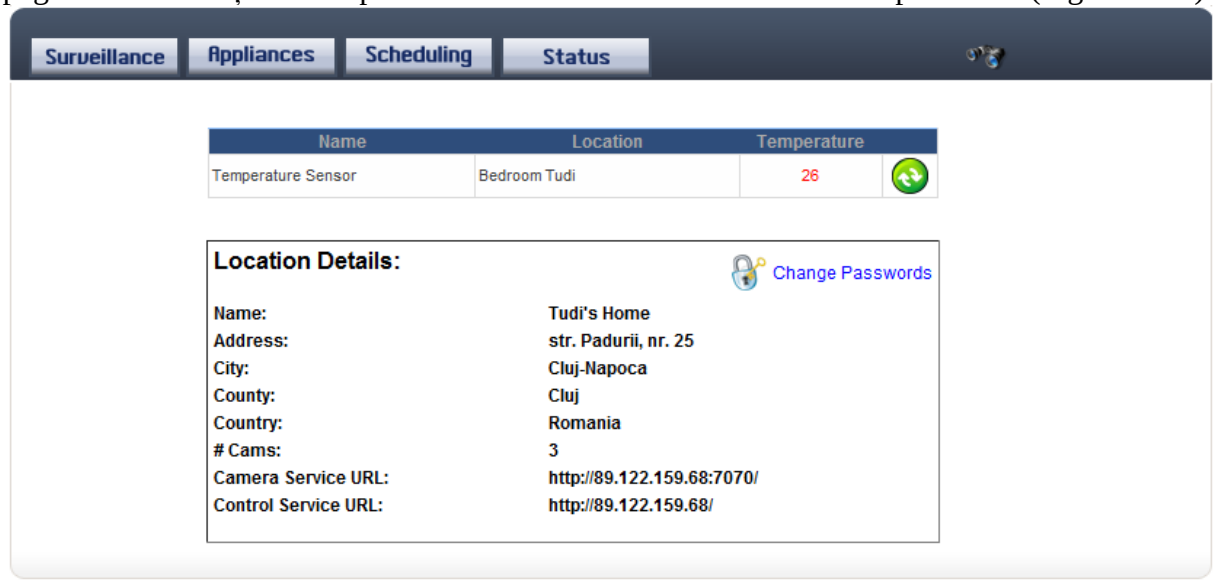


Figura 7.1 Pagina Status.aspx

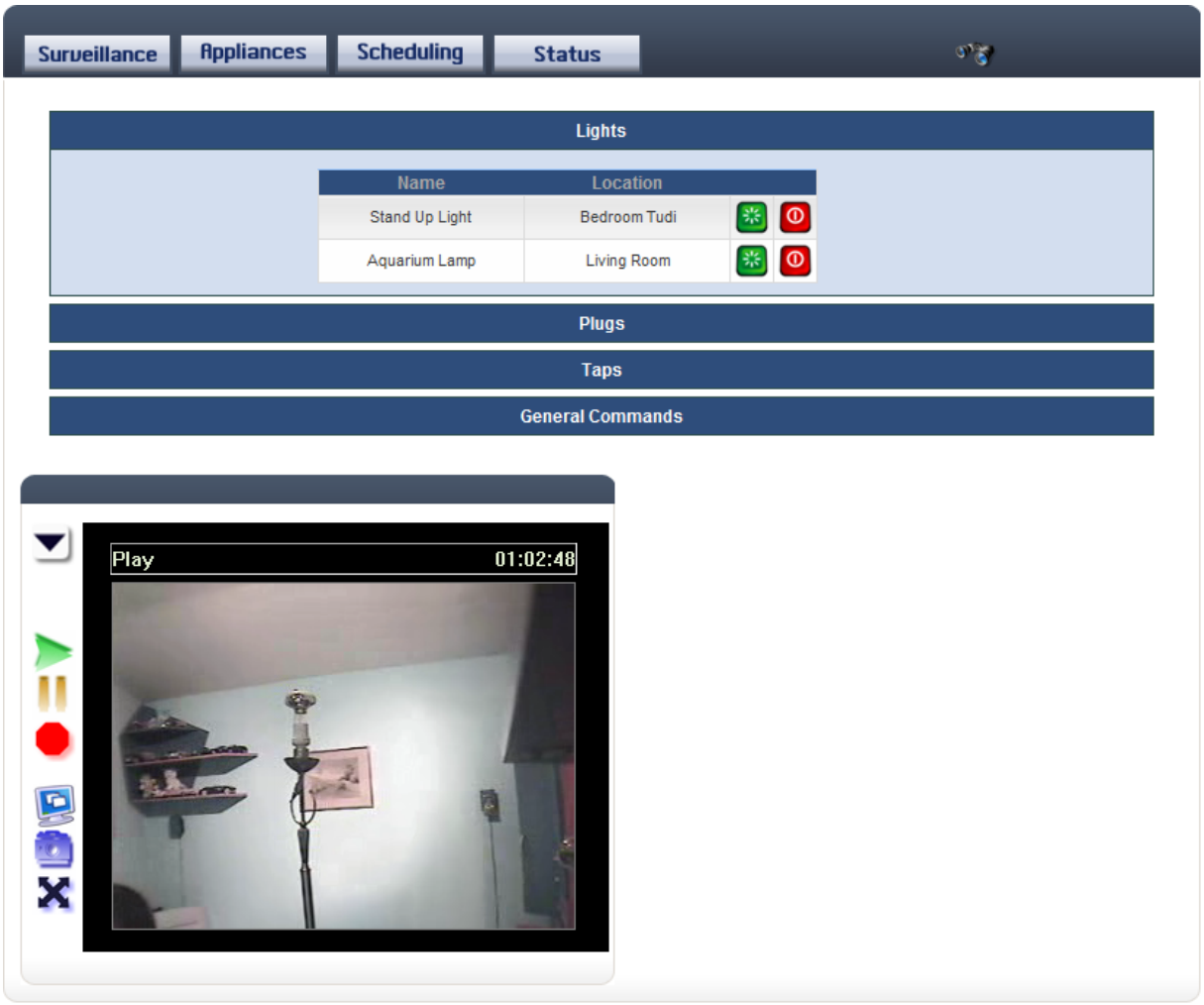


Figura 7.2 Pagina Appliances.aspx cu fereastra plutitoare activă, bec stins

Utilizatorul apăsă butonul de navigare către pagina „Appliances”. Pe această pagina apasă butonul de deschidere a ferestrei video plutitoare după care apăsă butonul play și

„streaming-ul” video pornește. Utilizatorul mută fereastra plutitoare în colțul stânga jos al ferestrei pentru a nu se suprapune cu ferestrele de control (Figura 7.2).

În imagine se poate observa becul din spectrul vizual al camerei video care este oprit. Utilizatorul apasă butonul de pornire din dreptul becului din tabela Lights. În momentul imediat următor se va aprinde becul. Acest lucru se poate observa în Figura 7.3. Efectul acțiunii poate avea o întârziere de 1-2 secunde datorită multitudinii de noduri și medii prin care trebuie să traverseze semnalul(Client -> WAN -> ServerCentral -> WAN -> ServerRemote -> RS232 -> Modul comanda X10 -> Rețea Electrica -> Modul control LM11).

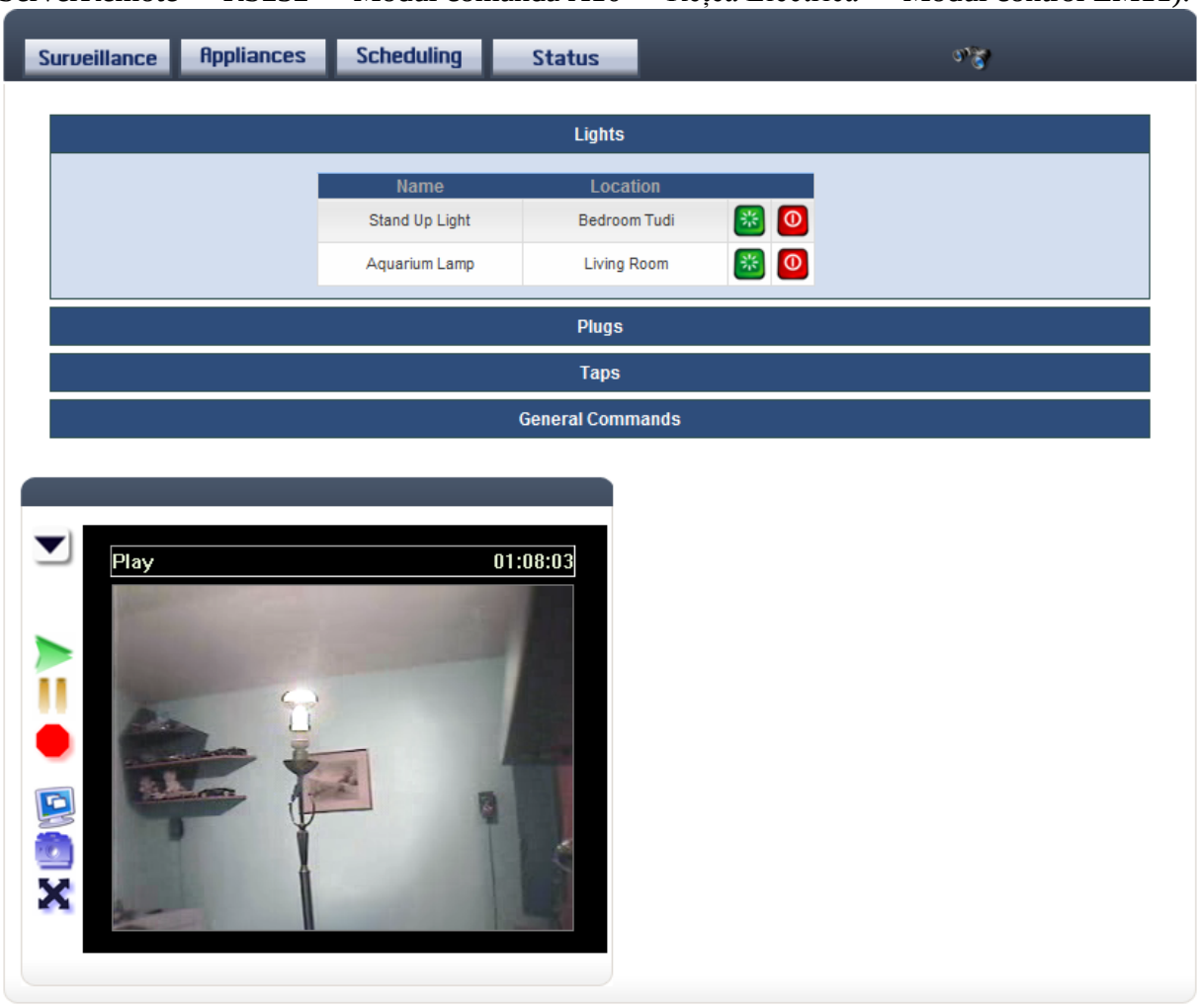


Figura 7.3 Pagina Appliances.aspx, fereastra plutitoare activă, bec aprins

Utilizatorul apasa link-ul „Scheduling” pentru a deschide pagina de programe automate. La deschiderea acestei ferestre pe lângă conținutul normal al ferestrei va apărea și fereastra plutitoare în poziția în care a fost lăsata pe pagina anterioară și în aceeași stare (Figura 7.4).

Utilizatorul apasă icoana cu binoclul pentru a închide fereastra plutitoare. Se adaugă două programe automate. Unul pentru oprirea becului pornit anterior și repornirea lui după terminarea udării și unul pentru pornirea instalației de udare cu timp de funcționare de 5 minute și repetare zilnică. Ambele programe vor avea timpul de pornire cu două minute mai târziu decât timpul actual. După adăugarea programelor se navighează la pagina „Surveillance”. Se va apasă butonul play pentru toate camerele video disponibile. Se va apasă butonul play pentru toate camerele video disponibile. Se setează prima fereastră video in așa fel incât în aceasta să se observe becul iar cea de-a doua sa se observe aspersoarele. În momentul anterior rulării programelor lumina este aprinsă și aspersoarele oprite.

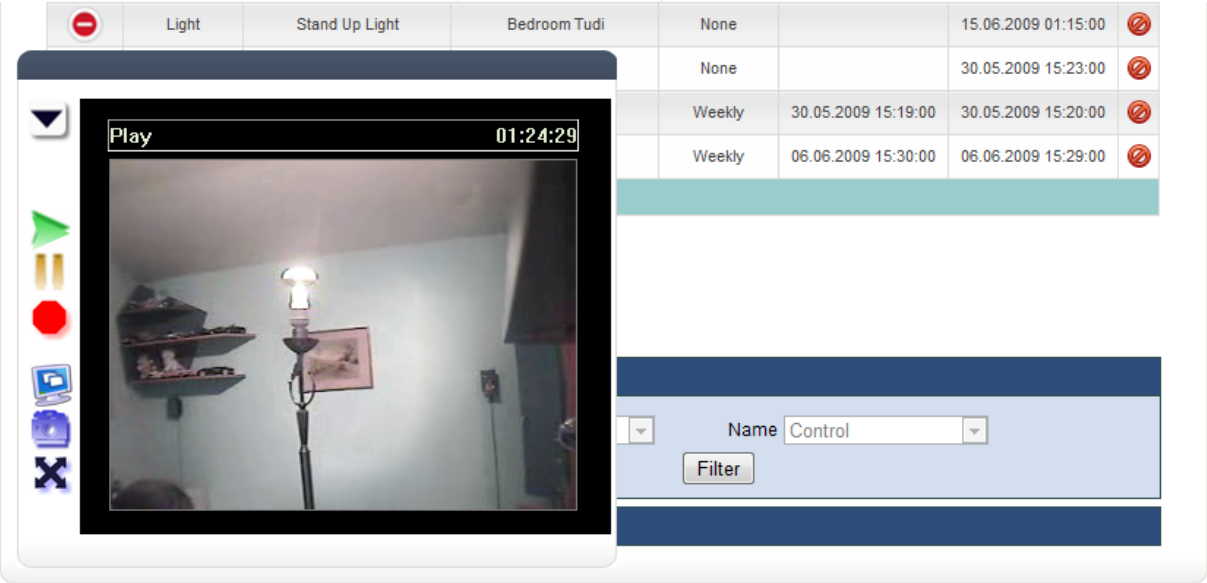


Figura 7.4 Partea inferioară a paginii Scheduling.aspx cu fereastra plutitoare

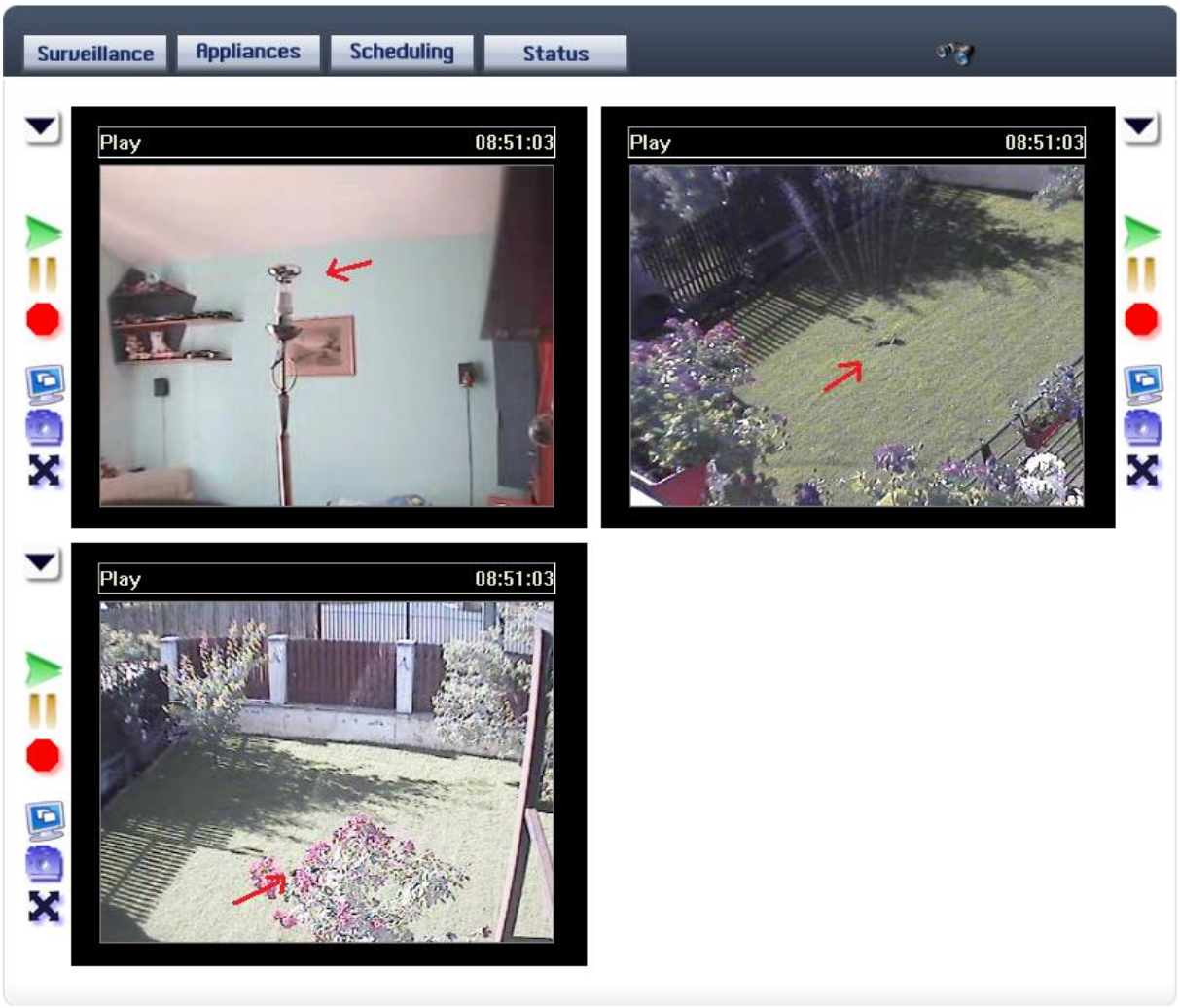


Figura 7.5 Pagina Scheduling.aspx în timpul rulării programelor automate

În Figura 7.5 este prezentată o captură a imaginii de după expirarea celor 2 minute, din timpul excuției programelor automate, lumina este stinsă și sistemul de aspersoare este pornit.

După terminarea ciclului de execuție a programelor automate (Figura 7.6) se revine la pagina „Scheduling” unde se poate observa faptul că data programului de udare s-a schimbat în data zilei următoare. Tot în Figura 7.7 se poate observa că programul automat de stingere a luminii s-a dezactivat iar programul de aspersoare a rămas activ întrucât va rula din nou ziua următoare.

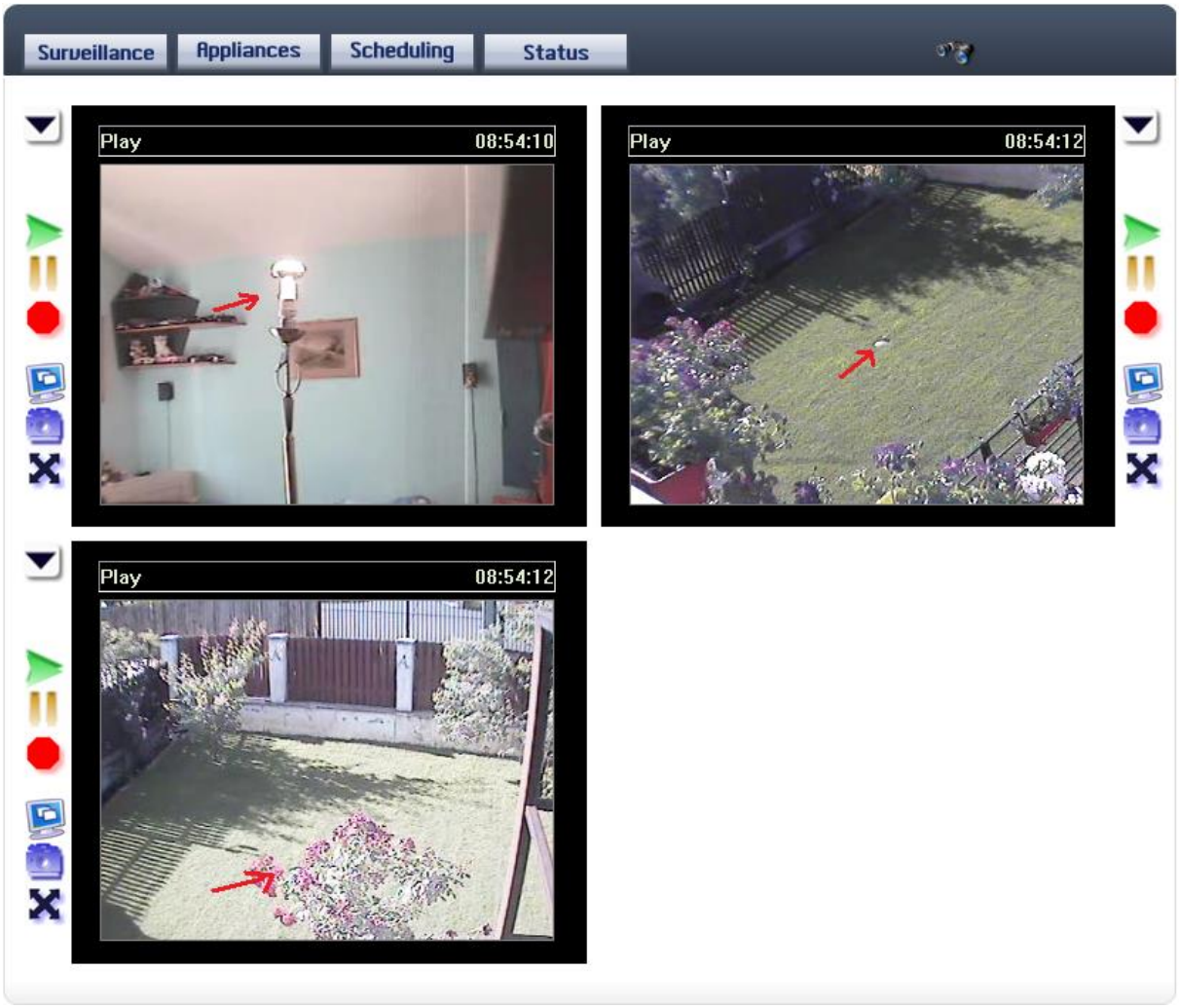


Figura 7.6 Pagina Scheduling.aspx după rularea programelor automate

Figura 7.7 Partea superioară a paginii Scheduling.aspx după rularea programelor

7.4 Rezolvarea dificultăților întâmpinate

Controlul electrovalvelor

Prima dificultate am întâmpinat-o în găsirea unei electrovalve (robinet controlat electronic). Aceste dispozitive le-am găsit la producătorul Gardena, care produce accesorii pentru îngrijirea grădinilor. Electrovalvele Gardena au un dispozitiv de comandă și control. Acesta poate da comenzi instantanee de deschidere sau închidere sau poate fi programat pentru a face acest lucru la o anumită oră din zi. Dispozitivele Gardena nu sunt furnizate împreună cu o documentație tehnică pentru că nu sunt concepute pentru dezvoltatori ci pentru uz casnic. Având în vedere acest lucru, a trebuit să descopăr cum funcționează o electrovalvă, mai exact ce semnal primește de la dispozitivul de control.

În Figura 7.8 puteți observa o electrovalva Gardena (stânga) și sistemul de control al acesteia (dreapta). Pentru a descoperi semnalele de comandă generate de sistemul de comandă al electrovalvei am folosit un osciloscop Tektronix.

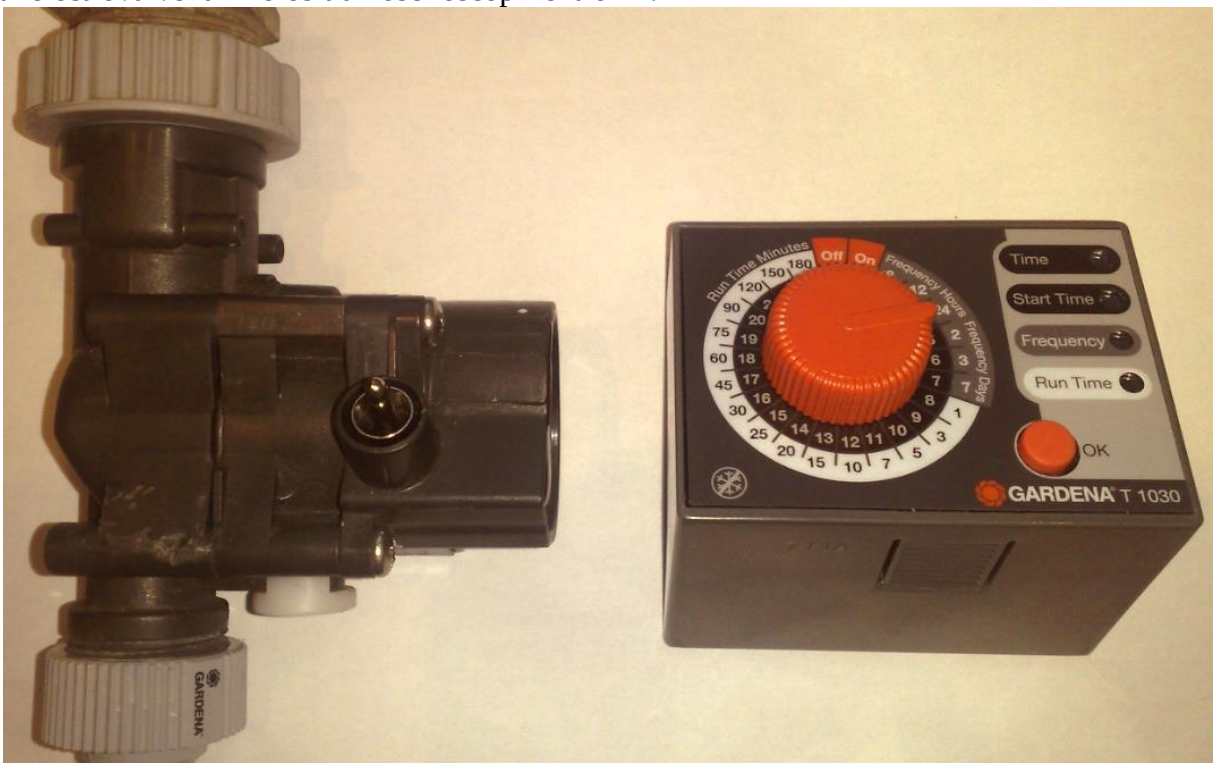


Figura 7.8 Electrovalva (stânga) și sistem de comanda (dreapta)

În Figura 7.9 în partea stânga este o captură a semnalului generat de sistemul de control Gardena pentru pornirea electrovalvei. Acesta este un semnal de -8.8V în punctul cel mai scăzut și are o durată de aproximativ 156ms. Pentru a genera un astfel de semnal am folosit un modul PmodHB3 (punte H) produs de Digilent. Pentru comanda acestui modul am folosit o placă cu microcontroler (Atmel ATMega64) Cerebot produsă de firma Digilent.

În Figura 7.9 în partea dreaptă este o captură a semnalului generat cu ajutorul HB3 pentru deschiderea electrovalvei. În această imagine se pot observa și semnalele de direcție (Galben) și selectie (Albastru) trimise către modulul HB3.

În Figura 7.10 în partea dreaptă se observă semnalul generat de modulul de control pentru închiderea electrovalvei. Acesata are o formă mult mai ciudata decât cel de deschidere, motivul fiind modul de funcționare al electrovalvei, care se bazează pe diferențe de presiune pentru deschiderea și închiderea valvei. Acest semnal nu poate fi generat cu un modul HB3. După ce am studiat modul de funcționare al valvei și semnalul inițial am ajuns la concluzia că

un semnal ca și cel din dreapta figurii ar trebui să închidă valva. Semnalul din dreapta este semnalul prin care modulul HB3 închide electrovalva.

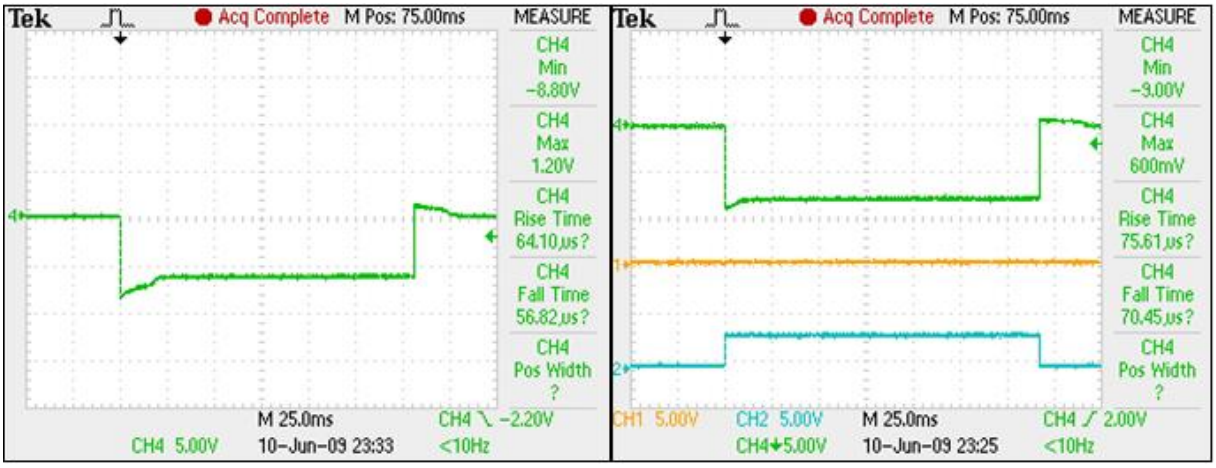


Figura 7.9 Comparație pornire electrovalvă

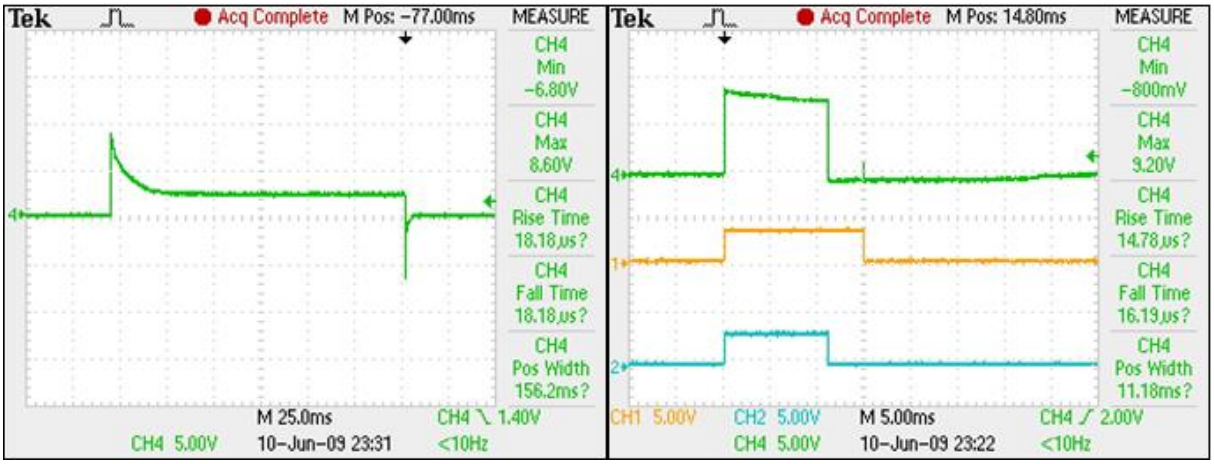


Figura 7.10 Comparație oprire electrovalvă

Comunicație serială în mediu distribuit

În locațiile „remote”, la serverul de control sunt conectate sistemul X10 și placa Cerebot. Acestea sunt conectate prin intermediul porturilor seriale folosind standardul RS232. Comunicația serială nu este foarte rapidă și doar o conexiune poate fi deschisă la un moment dat. Deoarece comenzile vin dintr-un mediu distribuit acestea s-ar putea suprapune, adică s-ar încerca deschiderea a mai multe conexiuni simultan, acest lucru ar eșua și comenzile nu ar fi executate. Pentru a evita o astfel de situație, cele două conexiuni seriale sunt deschise pe evenimentul „Application_Start” în fișierul “Global.asax”. Conexiunile fiind deschise astfel, fiecare thread va folosi aceeași conexiune. Conexiunea se închide în momentul în care se închide aplicația (pe server).

Securitate aplicație

O astfel de aplicație, prin care din orice locație din lume se poate controla orice locuință din sistem trebuie să fie extrem de sigură. Pentru a face acest lucru posibil aplicația are următoarele caracteristici:

- toate comunicațiile se fac prin intermediul protocolului HTTPS
- datele sensibile (utilizatori/parole) sunt ținute în baza de date criptate

- cheia de acces a utilizatorilor nu este ținută în baza de date ci este folosită pentru criptarea id-ului utilizator care este stocat în baza de date.
- administratorul nu știe parolele utilizatorilor, acestea fiind generate automat

Problema a apărut în momentul în care am adăugat programele automate. Acestea trebuie să se conecteze automat la sistemul „remote” și să dea comenzi de pornire/oprire dispozitive la momentele potrivite. Acest lucru înseamnă că o cheie va trebui ținută pe serverul principal. După un studiu aprofundat al problemei am implementat această parte astfel: în codul aplicației am reținut o constantă care reprezintă cheia serverului. Aceasta este transformată în hash(SHA-384), împărțită în 2 segmente de 256 respectiv 128 biți care sunt utilizați pentru a cripta/decripta cheile de acces către serverele remote care se află criptate în baza de date.

Alegerea echipamentului de supraveghere video

Alegerea echipamentului de supraveghere video a fost o problemă destul de mare. Acesta trebuia să îndeplinească următoarele condiții:

- scalabil – să existe o gamă de produse ale aceluiași producător compatibile cu același software și la nivele de preț diferite pentru a se potrivi oricărui tip de locuință
- fiabil
- firma producătoare să ofere suport tehnic – acest lucru era necesar pentru a putea integra sistemul de supraveghere în aplicația de control distribuit al inteligenței ambientale.

După un studiu al pieței și al producătorilor de astfel de echipamente am ales ca și placă de dezvoltare placa GV-600 de la firma producătoare Geovision.

Capitolul 8 Concluzii

8.1 Realizări

Obiectivul lucrării a fost realizarea unui sistem de control a locuinței de la distanță prin intermediul unei locații centrale. Toate obiectivele atat primare cât și secundare au fost îndeplinite.

Aplicaţia îndeplineşte toate cerinţele specificate, atât cele funcţionale, cât şi cele nefuncţionale, iar rezultatul final este mai complex decât specificația inițială.

Cerinţele funcţionale sunt cele descrise de cazurile de utilizare. În urma testării, s-a dovedit faptul că toate cazurile de utilizare sunt executate conform specificaţiilor.

Cerinţele nefuncţionale sunt îndeplinite:

- Uşurinţa în utilizare şi Accesibilitate
  - Interfaţa utilizator a fost proiectată pentru a fi intuitivă, structurată şi prietenoasă. Funcţionalităţile sunt grupate pentru un acces uşor.
  - Administrarea se face în totalitate de către administratorul sistemului reducând astfel nivelul de cunoștințe necesare pentru utilizarea sistemului
  - Întrucât este o aplicație web se poate accesa cu ușurință din orice locație conectată la Internet
- Fiabilitate şi Disponibilitate
  - Sistemul funcţionează corect, fără erori.
  - În cazul apariției unor defecțiuni care duc la erori administratorul de sistem are acces la un fișier de „log” care va ajuta la identificarea mai ușoară a erorilor și cauzelor acestora.
  - Specificațiile de instalare sunt stricte, dacă acestea se respectă disponibilitatea sistemului va fi crescută la maxim
- Performanţă
  - Performanța sistemului pentru un utilizator final depinde de lățimea de bandă (upload) și de latența conexiunii de care dispune locația serverului remote la care se conectează și de lățimea de bandă (download) de care dispune locația din care se face conectarea.
  - Pentru a nu se crea un „bottle-neck” datorită serverului central, streaming-ul video se va face direct între locația remote și client, serverul central fiind utilizat doar pentru aflarea datelor de conectare.
- Suport
  - Toate setările aplicației sunt ținute în fișiere de configurare. Datorită acestui lucru aplicația poate fi instalată în scurt timp și fără mult efort în diverse locații.
  - Toate detaliile unei noi locații sunt ținute în baza de date și se introduc din paginile de administrare, acest lucru face ca adăugarea unei locații să se facă rapid și fără efort
- Scalabilitatea
  - Prin arhitectura implemetntată sistemul devine ușor scalabil

8.2 Avantaje aduse de soluție

În secolul XXI oamenii sunt din ce în ce mai ocupați dar au acces mult mai ușor la tehnologie și informație. Astfel, oriunde s-ar afla aproape totdeauna au în preajma un computer sau un smart-phone cu care să se poată conecta la Internet.

Cu ajutorul acestei aplicații, dacă o persoană se poate conecta la Internet ea se poate conecta la propria casă. După efectuarea conexiunii locuința poate fi supravegheată, luminile și aparatele electrocasnice pot fi pornite sau oprite, chiar și robinetele pot fi deschise sau închise.

Cel mai mare avantaj al acestei soluții este faptul că se pot controla diverse dispozitive din casa cu aceeași aplicație. Pe piață există diverse aplicații pentru control la distanță dar fiecare este specializată pe cate o ramură (supraveghere, control dispozitive electrice, etc.). Soluția 'Smart-Homes' permite conectarea la orice număr de case fără ca utilizatorul să fie nevoit să facă vreo schimbare în configurarea aplicației. Întrucât controlul se realizează prin intermediul unui browser web, aplicația nu trebuie instalată pe mașina de pe care se face conectarea.

În tabelul următor voi compara varianta propusă de soluția 'Smart-Homes' cu variantele tradiționale.

Tabel 8.1

Dezavantajele variantei tradiționale	Avantajele aduse de www.smart-homes.ro
Dispozitive din domenii diferite -> aplicații diferite	Dispozitive din domenii diferite -> aceeași aplicație
Administrarea este făcută de utilizatorul final.	Administrarea este făcută de administratorul din locația centrală.
Acces prin aplicație desktop	Acess prin aplicație web
Instalarea aplicațiilor pe alte computere decât cel personal poate duce la accesul persoanelor neautorizate în sistem.	Sistemul utilizează multiple variante de criptare a cheilor de acces și comunicație securizată (https) pentru a nu permite conectarea persoanelor neautorizate.
Valvele din rețeaua de alimentare cu apă sau gaz se pornesc manual sau cu ajutorul unui programator local.	Aplicația oferă controlul de la distanță a electrovalvelor din reațeaua de alimentare cu apă sau gaz.

8.3 Direcţii de dezvoltare

Un sistem de control de la distanță a unei locuințe trebuie dezvoltat continuu deoarece zilnic apar noi "gadget-uri". Fiecare dintre acestea poate fi controlat de la distanță.

Sistemul a fost gândit în așa fel încât dezvoltările ulterioare să se poată face cu ușurință. Posibile dezvoltări ulterioare:

- Adăugarea unui sistem de mișcare a camerelor video pe axele x, y. Acest lucru se va putea realiza prin utilizarea a 2 motoare(servo sau DC) pentru fiecare cameră video care trebuie controlată. Motoarele se vor conecta la placă Cerebot, implementarea va fi similară cu cea a controlului robinetelor.
- Realizarea unui mecanism de tip „termostat" pe serverele remote. În funcție de senzorii de temperatură instalați în fiecare cameră acest server va porni sau opri caloriferele electrice pentru a păstra temperatura constantă.
- Integrarea sistemului cu sistemul de alarmă a locuinței. Acest lucru ar oferi posibilitatea verificării stării sistemului, armarea/dezarmarea, dezactivarea unor senzori care nu funcționează în parametrii normali.
- Realizarea unui sistem de salvare a imaginilor de pe camerele video pe serverul central în cazul declanșării alarmei în locația remote. Astfel imaginile cu infractorii vor fi într-un loc sigur chiar dacă aceștia distrug serverul sau iau harddisk-ul acestuia.

Bibliografie

[1]. **Chris Britton, Peter Bye.** *IT Architecture & Middleware: Strategies for Building Large, Integrated Systems,* Pearson, 2004. ISBN-13: 9780321246943.

[2]. **George Coulouris, Jean Dollimore, Tim Kindenberg.** *Distributed Systems - concepts and design,* Addison-Weseley, 2007. ISBN: 0201-619-180.

[3]. **Armando Roy Delgado, Rich Picking, Vic Grout,** *Remote-Controlled Home Automation Systems with Different Network Technologies.***.** Wrexham, UK : Centre for Applied Internet Research (CAIR), University of Wales, NEWI, 2005.

[4]. **Digilent Inc.** *Cerebot Reference Manual.* Pullman, WA, SUA, Digilent Inc., 2009.

[5]. **Digilent Inc.** *PmodTMP Reference Manual.* Pullman, WA, SUA, Digilent Inc., 2008.

[6]. **Digilent Inc.** *Digilent PmodHB3 2A H-Bridge Reference Manual.* Pullman, WA, SUA, Digilent Inc., 2007.

[7]. **Bill Evjen, Scott Hanselman, Farhan Muhammad, Srinivasa Sivakumar, Devin Rader,** *Professional ASP.NET 2.0.,* Wrox, 2005, ISBN: 0-7645-7610-0.

[8]. **Ingrid Van Den. Hoogen,** *Deutsch's Fallacies, 10 Years After,* http://java.sys-con.com/read/38665.html, 2004.

[9]. **Michael Howard, David LeBlanc,** *Writing Secure Code.,* Microsoft, 2002.ISBN: 0-7356-1588-8.

[10]. **Eric Ledoux,** *Cm11.dll. Reference Manula*. www.dwell.net/Cm11, 2009.

[11]. **Fernando Moraes, Alexandre Amory, Juracy Jr. Petrini** *Sistema Cliente-Servidor para Supervisão de Processos através da Web,* http://www.ee.pucrs.br/~amory/Trabalho_de_Conclusao/trabalho_de_conclusao.html, 2000.

[12]. **Brad A. Myers,** *Taking handheld devices to the next level,* IEEE Computer Society, 2004.

[13]. **Christian Nagel, Bill Evjen, Jay Glynn, Morgan Skinner, Karli Watson, Allen Jones.** *Professional C# 2005,* Wrox, 2006, ISBN: 0-7645-7534-1.

[14]. **P.Rigole, Y. Berbers, T. Holvoet,** *A UPnP software gateway towards EIB home automation.* Cancun, Mexico, CST 2003

[15]. **Saumil Shah, Shreeraj Shah, Stuart McClure.** *Web Hacking,* Addison Wesley, 2002.

[16]. **Charles W. Sullivan,** *CM11A (X10) Protocol Document,* http://wanderingsamurai.net/electronics/cm11a-x10-protocol-document, 2006.

[17]. **Wikipedia.** *Home automation*, http://en.wikipedia.org/wiki/Home_automation, Wikipedia, 30.05.2009.

[18]. **Nicholas C. Zakas** *Professional JavaScript for Web Developers.*, Wrox, 2005. ISBN: 0 - 7645 -7908 - 8.

Acronime

API – Application Programming Interface

CORBA – Common Object Request Broker Architecture

EIB – European Instalation Bus

LAN – Local Area Network

MVC – Model View Controller

PDA – Personal Digital Assistant

RMI – Remote Method Invocation

SNMP – Single Network Management Protocol

SOAP – Simple Object Access Protocol

TCP – Transmission Control Protocol

UPnP – Universal Plug and Play