UNIVERSITATEA „ALEXANDRU IOAN CUZA” IAŞI

**FACULTATEA DE INFORMATICĂ**



LUCRARE DE LICENŢĂ

**Secure Bit Exchange**

**propusă de**

***Ioana Alexandra Tănase***

**Sesiunea:** *Iulie, 2016*

**Coordonator ştiinţific**

**Conferențiar Dr. Ștefan Ciobâcă**

**UNIVERSITATEA ALEXANDRU IOAN CUZA IAŞI**

**FACULTATEA DE INFORMATICĂ**

**Secure Bit Exchange**

***Ioana Alexandra Tănase***

**Sesiunea:** *Iulie, 2016*

**Coordonator ştiinţific**

**Conferențiar Dr. Ștefan Ciobâcă**

DECLARAŢIE PRIVIND ORIGINALITATE ŞI RESPECTAREA DREPTURILOR DE AUTOR

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

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

Iaşi, *data*

Absolvent *Ioana Alexandra Tănase*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(semnătura în original)

DECLARAŢIE DE CONSIMŢĂMÂNT

Prin prezenta declar că sunt de acord ca Lucrarea de licență cu titlul „*Secure Bit Exchange”*, codul sursă al programelor şi celelalte conţinuturi (grafice, multimedia, date de test etc.) care însoţesc această lucrare să fie utilizate în cadrul Facultăţii de Informatică.

De asemenea, sunt de acord ca Facultatea de Informatică de la Universitatea „Alexandru Ioan Cuza” Iași să utilizeze, modifice, reproducă şi să distribuie în scopuri necomerciale programele-calculator, format executabil şi sursă, realizate de mine în cadrul prezentei lucrări de licenţă.

Iași, *data*

Absolvent *Ioana Alexandra Tănase*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(semnătura în original)

**CUPRINS**

**Inroducere**

**Contribuții**

1. **Prezentarea sistemului de operare Android**
   1. Scurt istoric
   2. Android 6.0 Marshmallow
   3. Concluzii
2. **Tehnologii folosite**
   1. Android Studio IDE
   2. Limbajul de programare Java
   3. Concluzii
3. **Descrierea aplicației**
   1. Analiză și proiectare
   2. Arhitectură
   3. Implementare
      1. Aplicația de bază
      2. Serviciul NSD
      3. Comunicarea Server - Client
      4. Commitment Protocol
      5. Agreement Protocol
   4. Tutorial

**Concluzii**

**Bibliografie**

**INTRODUCERE**

Ultimii zece ani au reprezentat un mediu prosper pentru industria telefoanelor mobile ce ruleaza sub sistemul de operare Android. Încă de la lansarea platformei, noiembrie 2007, aspectul open source al acesteia a stârnit o mulțime de contradicții, dar s-a dovedit a fi un mare plus al acestui sistem.

Atât portabilitatea, cât și mâna liberă a utilizatorului spre content puternic customizabil, reprezintă aspecte unice ale sistemului de operare Android. Datorită acestor facilități împreună cu permisibilitatea Magazinului Play de a accepta promovarea aplicațiilor la un preț redus, appstore-ul a fost asaltat cu milioane de programe soft.

La scurt timp au început să apară diferite aplicații menite să ușureze munca de zi cu zi a oamenilor. Pe lângă altele, au apărut diferite instrumente soft care să permită menținerea scorului pentru jocurile de masă. Popularitatea lor a început să crescă în ultimul timp, dar toate dispun de o slabă securitate și un nivel scăzut de interactivitate. Câteva exemple de astfel de aplicații: Simple Score Sheet[[1]](#footnote-1), 500 Scorer[[2]](#footnote-2), Yatzee Scorer[[3]](#footnote-3).

Am decis așadar, în urma motivelor expuse anterior, crearea unui software dezvoltat pe platforma Android care să combine interacțiunea oamenilor cu interesul pentru tehnologie prezentat de aceștia.

Obiectivul principal al acestei lucrări este de a facilita întrunirile cu scop de amuzament oferind un mediu lipsit de oboseala calculelor, bazat pe încredere, ferit de orice tentativă de a trișa. Propun în acest scop o aplicație care să satisfacă aceste două direcții.

Proiectul de față este realizat cu ajutorul uneia dintre cele mai puternice instrumente de dezvoltare a programelor software, Java, rulat de Android Studio IDE v2.1.2.

Lucrarea este structurată pe trei părţi: Prezentarea sistemului de operare Android, Tehnologii utilizate şi Descrierea aplicaţiei.

Prima parte este formată din două capitole în care se descrie pe scurt sistemul de operare Android și avantajele acestuia.

Cea de-a doua parte surprinde o expunere concisă a tehnologiilor utilizate în implementarea aplicației.

În cea de-a treia parte este prezentată aplicația din mai multe puncte de vedere. Primul capitol face analiza generală a functionalităților descriind modul în care interacționează serviciul Network Service Discovery cu protocolul de comunicare, sub schema Server - Client.

Al doilea capitol reprezintă aspecte legate de modalitatea de implementare a soluției.

Cel de-al treilea și ultim capitol face o prezentare amănunțită a programului din două perspective: sever și client.

**CONTRIBUȚII**

Ideea construirii unei aplicații Android a fost propusă de mine. Alegerea tematicii a fost realizată în colaborare cu domnul profesor Ștefan Ciobîcă, acesta elaborând varianta finală.

Elementele de teorie folosite în construirea proiectului survin în proporție majoritară din partea profesorului meu coordonator, pe când soluțiile practice folosite în dezvoltarea lucrării provin în principal din propria experiență profesională, dobândită programând pe platforma Android, completată de noțiuni de Programare Orientată Obiect și Programare Avansată Java, acumulate pe parcursul anilor din facultate.

Sub îndrumare constantă am implementat un serviciu ce poate fi descoperit pe infrastructura Wi-Fi, ușurând astfel comunicarea între Server și Client, bazată pe protocolul de securitate de BitAgreement.

Colaborarea coordonator – student s-a produs sub forma de întâlniri săptămânale împreună cu numeroase convorbiri purtate cu ajutorul e-mailul, comunicarea realizându-se astfel într-un mod transparent și eficient.

Toți acești factori au contribuit la o bună și prosperă dezvoltare a prezentei lucrări și a aplicației practice.

1. **Prezentarea sistemului de operare Android**
2. **Scurt istoric**
3. **Introducere**

În încercarea de a surprinde toată evoluţia sistemului de operare Android, la ora actuală cel mai popular dintre cele dedicate dispozitivelor portabile, se va atrage atenţia asupra tuturor modificărilor esenţiale din istoria platformei.

Construit în jurul unui nucleu Linux, întocmai precum Chrome OS sau Mac OS X, Android a fost iniţial efortul unei companii omonime – Android Inc., achiziţionată de Google în 2005. Nu a durat mult şi în 2007, sub atenta supraveghere a celor de la Google, s-a înfiinţat Open Handset Alliance, un concern creat pentru a stimula evoluţia tehnologică în domeniul dispozitivelor portabile şi, odată cu ea, a sistemului de operare care va rula pe noile dispozitive aparţinând acestei clase de produse. În același an a fost lansat sistemul de operare Android, într-o formă nu foarte apropiată de cea pe care o cunoaştem astăzi.

Încă de la lansare, noul OS a aparţinut comunităţii, fiind declarat open source. Totuşi, a fost creată şi o instituţie intitulată Android Open Source Project, având drept scop supravegherea atentă şi dezvoltarea permanentă a proiectului.

1. **Caracteristici**

Câteva dintre caracteristicile de bază ale sistemului de operare ce merită menționate:

1. Platforma este adaptabilă la configuraţii mai mari, VGA, biblioteci grafice 2D, biblioteci grafice 3D bazate pe specificaţia OpenGL ES 1.0 şi configuraţii tradiţionale smartphone.
2. Software-ul de baze de date SQLite este utilizat în scopul stocării datelor.
3. Android suportă tehnologii de conectivitate incluzând GSM/EDGE, CDMA, EV-DO, UMTS, Bluetooth și Wi-Fi.
4. Navigatorul de web disponibil în Android este bazat pe platforma de aplicații open source WebKit.
5. Software-ul scris în Java poate fi compilat în cod masina Dalvik și executat de mașina virtuala Dalvik, care este o implementare specializată de mașină virtuală concepută pentru utilizarea în dispozitivele mobile, deși teoretic nu este o Mașină Virtuală Java standard.
6. Android acceptă urmatoarele formaturi media audio/video/imagine: MPEG-4, H.264, MP3, AAC, OGG, AMR, JPEG, PNG, GIF.
7. Sistemul poate utiliza camere video/foto, touchscreen, GPS, accelerometru, și grafică accelerată 3D.
8. Mediul de dezvoltare include un emulator de dispozitive, unelte de depanare, profilare de memorie și de performanță, un plug-in pentru mediul de dezvoltare Eclipse.
9. Similar cu App Store-ul de pe iPhone, Piața Android este un catalog de aplicații care pot fi descarcăte și instalate pe hardware-ul țintă prin comunicație fără fir, fără a se utiliza un PC. Inițial au fost acceptate doar aplicații gratuite. Aplicații contra cost sunt disponibile pe Piața Android începând cu 19 februarie 2009.
10. Are suport nativ pentru multi-touch, dar această funcționalitate este dezactivată (posibil pentru a se evita încalcarea brevetelor Apple pe tehnologia touch-screen). O modificare neoficială, care permite multi-touch a fost dezvoltată.
11. **Dezvoltarea de software**

Primele aprecieri cu privire la dezvoltarea aplicațiilor pentru platforma Android au fost amestecate. Problemele citate includeau bug-uri, lipsa de documentație, infrastructura de testare inadecvată, și lipsa unui sistem de gestionare a problemelor public. Google a anunțat un sistem de gestionare a problemelor la data de 18 ianuarie 2008.În decembrie 2007, fondatorul startup-ului mobil MergeLab, Adam MacBeth, a declarat: *"Funcționalitatea lipsește, este prost documentată sau pur și simplu nu funcționează... Este clar că nu este gata pentru prime time."*

În ciuda acestui fapt, aplicațiile pentru Android au început să apară deja în săptămâna următoare celei în care a fost anunțată platforma. Prima aplicație publică a fost [jocul Snake](https://ro.wikipedia.org/w/index.php?title=Snake_(joc_video)&action=edit&redlink=1). Telefonul Android Dev este un dispozitiv cu SIM și hardware neblocate care este destinat dezvoltatorilor avansați. Cu toate că dezvoltatorii pot utiliza un dispozitiv de consum achiziționat de pe piață pentru a-și testa și a utiliza aplicațiile, unii dezvoltatori pot alege să nu utilizeze un dispozitiv de pe piață, preferând un aparat neblocat sau fără contract.

**Software Development Kit**

SDK-ul Android include un set complet de instrumente de dezvoltare.Acestea includ un [program de depanare](https://ro.wikipedia.org/w/index.php?title=Debugger&action=edit&redlink=1), [biblioteci](https://ro.wikipedia.org/w/index.php?title=Bibliotec%C4%83_software&action=edit&redlink=1), un [emulator de dispozitiv](https://ro.wikipedia.org/w/index.php?title=Emulator&action=edit&redlink=1) (bazat pe [QEMU](https://ro.wikipedia.org/w/index.php?title=QEMU&action=edit&redlink=1)), documentație, mostre de cod și tutoriale. Platformele de dezvoltare sprijinite în prezent includ calculatoare bazate pe [x86](https://ro.wikipedia.org/wiki/X86) care rulează [Linux](https://ro.wikipedia.org/wiki/Linux) (orice distribuție Linux desktop modernă), [Mac OS X](https://ro.wikipedia.org/wiki/Mac_OS_X), orice distribuție Windows. Cerințele includ, de asemenea, [Java Development Kit](https://ro.wikipedia.org/w/index.php?title=Java_Development_Kit&action=edit&redlink=1), [Apache Ant](https://ro.wikipedia.org/w/index.php?title=Apache_Ant&action=edit&redlink=1), și [Python](https://ro.wikipedia.org/wiki/Python_(limbaj_de_programare)). Mediul de dezvoltare (IDE) suportat oficial este Android Studio[[4]](#footnote-4), bazat pe IntelliJ IDEA[[5]](#footnote-5), deși dezvoltatorii pot folosi orice editor de text pentru a edita fișiere XML și Java și apoi să utilizeze unelte din linia de comandă pentru a crea, construi și depana aplicații Android.

1. **Platforme Android**

În tabelul 1 de mai jos puteți vedea istoricul versiunilor Android, denumirea comercială și distribuția acestora în funcție de dispozitivele care au accesat magazinul Play Store în a doua parte a lunii august 2013.Pe 25 Iunie, Google a anunțat Android Lollipop, acesta fiind cel mai mare update pe care l-a suferit platforma Android de la Ice Cream Sandwich și până în prezent. Android Lollipop 5.0 este disponibil începând cu toamna anului 2014. Ultimul update, Android 6.0 Marshmallow, s-a lansat în toamna anului 2015.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Versiune** | **Nume de cod** | **Data lansării** | **API level** | **Distribuție** |
| **6.0** | *Marshmallow* | Google I/O 2015 | 23 |  |
| **5.0-5.1** | *Lollipop* | Google I/O 2014 | 21 | 9% |
| **4.4** | *KitKat* | 31 Octombrie 2013 | 19 | 39.8% |
| **4.3** | *Jelly Bean* | 24 iulie 2013 | 18 | 5.5% |
| **4.2** | *Jelly Bean* | 13 noiembrie 2012 | 17 | 18.1% |
| **4.1** | *Jelly Bean* | 9 iulie 2012 | 16 | 36.6% |
| **4.0.3–4.0.4** | *IceCream Sandwich* | 16 decembrie 2011 | 15 | 5.3% |
| **3.2** | *Honeycomb* | 15 iulie 2011 | 13 | 0.1% |
| **3.1** | *Honeycomb* | 10 mai 2011 | 12 | 0% |
| **2.3.3–2.3.7** | *Gingerbread* | 9 februarie 2011 | 10 | 5.7% |
| **2.3–2.3.2** | *Gingerbread* | 6 decembrie 2010 | 9 | 5,7% |
| **2.2** | *Froyo* | 20 mai 2010 | 8 | 2.4% |
| **2.0–2.1** | *Eclair* | 26 octombrie 2009 | 7 | 0.3% |
| **1.6** | *Donut* | 15 septembrie 2009 | 4 | 0% |
| **1.5** | *Cupcake* | 30 aprilie 2009 | 3 | 0% |

Tabel 1: Istoricul versiunilor Android[[6]](#footnote-6), August 2013

1. **Actualizări**

Android 6.0, denumit și Marshmallow, este cea mai nouă versiune a sistemului de operare[[7]](#footnote-7). Schimbările aduse nu sunt atât de mari în comparație cu celelalte update-uri. Marshmallow reprezintă, mai degrabă, extinderea caracteristicilor de bază și funcționalitatea Android Lollipop.

Deși un preview al softului a fost anunțat pe data de 28 mai 2015, versiunea finală nu a fost lansată decât pe 29 septembrie, împreună cu noile smartphone-uri Nexus.

Marshmallow aduce o transformare a telefonului în propriul portofel. Această aplicație permite înregistrarea cardurilor, chiar a celor ce nu provin din industria bancară. Mulțumită popularității sistemului de operare acest nou feature promite potențial.

Când vine vorba de design, Android Lollipop a dus termenul de inovație la noi limite, pe când Marshmallow aduce o reorganizare a sistemului de permisiuni pentru aplicații la fel de promițătoare. Este un mare bonus pentru securitate și confidențialitate. Noul sistem îi permite utilizatorului să aleagă informațiile la care are acces orice aplicație instalată.

Schimbarea în materie de securitate nu se oprește aici. Noua versiune vine cu un senzor de scanare al amprentelor. Dezvoltatorii vor putea folosi mai eficient suportul pentru așa ceva, iar securizarea aplicațiilor va putea fi posibilă cu ajutorul acestui senzor.

Android 6.0 include suport pentru porturi de tipul USB-C, foarte populare la acest moment. Caracteristica importantă a acestui USB este bidirecționalitatea, ceea ce face ca bateria să se încarce de până la trei ori mai repede.

Căutarea online este cheia celor de la compania Google. Marshmallow propune un nou tip de căutare, lăsând de-o parte motorul de search. În schimb se va folosi aplicația Google Now, ce se comportă ca un scaner contextual al intregului ecran. Dacă cineva te invită la film, Now on tap va scana informațiile și îți va afișa orice legătură găsită cu respectivul film: trailere, case de bilete, etc, transformând procesul de căutare într-o experiență mult mai plăcută și mai economisitoare de timp.

Când Now on Tap se va dezvolta suficient de bine, va reuși să înțelegă ce face o aplicație fără ajutor uman. Va ști deci ceea ce utilizatorul dorește de la o aplicație, fără ca el să îi ceară ceva precis.

1. **Concluzii**

După cum se poate vedea, platforma Android are multe beneficii peste iOS, dar și unele slăbiciuni. Pentru dezvoltatori, un prim avantaj este limbajul Java, care este foarte accesibil, datorită istoriei îndelungate și a multitudinii de unelte de dezvoltare, care ajută în crearea unei proces end-to-end foarte bine pus la punct.

În plus, platforma este open-source, astfel încât cei care doresc să creeze ROM-uri custom o pot face fără probleme. Tot pentru Java sunt prezente și foarte multe biblioteci care facilitează o serie de tehnici moderne de dezvoltare, cum ar fi Inversion of Control sau Annotations.

Costurile de dezvoltare sunt și ele mai mici. O aplicație Android poate fi dezvoltată pe un calculator cu sistem de operare Ubuntu, care este gratuit, reducând costurile licențelor și a hardware-ului la minimum. Un alt aspect pozitiv este lipsa rigorii la publicarea unei aplicații pe Google Play, făcând ca ea să fie mult mai repede accesibilă și ca update-urile să fie livrare în timp mai scurt.

Totuși, una dintre problemele cu care se confruntă platforma este fragmentarea, având în vedere mai ales ritmul accelerat de dezvoltare. Multitudinea de producători de terminale mobile care implementează versiuni customizate de Android, cum ar fi Samsung sau HTC cu SenseUI, contribuie la scaderea ratei de adopție, destul de mică în comparație cu iOS. Cu toate că doar două versiuni sunt dominante pe piață (2.3 și 4.0), există între ele diferențe destul de mari, în special la implementarea user experience, versiunea 4 aducând multe facilitați și componente noi.

Google oferă o documentație[[8]](#footnote-8) generoasă și foarte bine pusă la punct, iar comunitatea online de dezvoltatori este foarte activă, cu multe tutoriale și exemple disponibile gratuit.

1. **Tehnologii folosite**
   1. **Android Studio IDE**

Mediu de dezvoltare[[9]](#footnote-9) oficial al programării sub platforma cu același nume, Android Studio asigură cele mai rapide instrumente necesare construirii de aplicații pentru fiecare tip de dispozitiv Android.

Iată câteva dintre funcționalitățile oferite:

* IntelliJ IDE + Android Studio plugin (editor de cod intelligent)
* Android SDK Tools (
* Unelte de debugging
* Sistem instant build/deploy
* Emulator rapid cu sitem de operare Android ce include Google Play Services
* Template-uri de cod
* Integrare GitHub
* O versiune a platformei Android

Toate acestea sunt oferite la un nivel de performanță ridicat și având drept suport flexibilitatea proiectelor.

* 1. **Limbajul de programare Java**
     1. **Prezentare**

Java reprezintă un limbaj de programare de nivel înalt, apărut mai ales ca urmare a eforturilor firmei JavaSoft, ulterior SUN Microsystems. Aceasta a încercat să dezvolte un limbaj de programare pentru echipamentele inteligente (telefonul, cuptor cu microunde, carduri, detectoare de fum şi a altor dispozitive mici) în cadrul aşa numitului proiect Green. Mai mult decât atât, datorită provocării lansate de Internet, era nevoie de un limbaj care să permită utilizarea aplicaţiilor pe o mulţime de platforme.

Limbajul a apărut in August 1991 și îl are drept autor pe James Gosling. Autorul, Patrick Naughton şi Bill Joy sunt membrii echipei pentru implementarea proiectului Green, proiect care se ocupa de programarea dispozitivelor inteligente. Cei trei au pus bazele interpretorului Java, scris în C.

Tot ce a urmat după aceea a fost dezvoltarea ideii pentru ca noi astăzi să avem parte de un nou limbaj, superior din multe puncte de vedere limbajului C++. Astăzi, toate instrumentele necesare dezvoltării şi rulării programelor Java sînt incluse în ceea ce se numeşte Java Development Kit, pe scurt JDK.

* + 1. **Avantajele limbajului**

Dintre caracteristicile principale ale limbajului amintim:

* **simplitate**, Java este uşor de învăţat, caracteristicile complicate (supraîncărcarea operatorilor, moştenirea multiplă, şabloane) întâlnite în alte limbaje de programare sunt eliminate.
* **robusteţe**, elimină sursele frecvente de erori ce apar în programare prin eliminarea pointerilor, administrarea automată a memoriei şi eliminarea fisurilor de memorie printr-o procedură de colectare a 'gunoiului' care rulează în fundal. Un program Java care a trecut de compilare are proprietatea că la execuţia sa nu "crapă sistemul".
* **orientat pe obiecte**, elimină complet stilul de programare procedural; se bazează pe încapsulare, moştenire, polimorfism.
* **uşurinţă** în ceea ce priveşte programarea în reţea.
* **securitate**, este cel mai sigur limbaj de programare disponibil în acest moment, asigurând mecanisme stricte de securitate a programelor concretizate prin: verificarea dinamică a codului pentru detectarea secvenţelor periculoase, impunerea unor reguli stricte pentru rularea programelor lansate pe calculatoare aflate la distanta, etc.
* este **neutru** din punct de vedere arhitectural.
* **portabilitate**, cu alte cuvinte Java este un limbaj independent de platforma de lucru, aceeaşi aplicaţie rulând, fără nici o modificare, pe sisteme diferite cum ar fi Windows, UNIX sau Macintosh, lucru care aduce economii substanţiale firmelor care dezvoltă aplicaţii pentru Internet. Sloganul de bază este: „Write once, run anywhere”.
* **compilat şi interpretat**.
* asigură o **performanţă ridicată** a codului de octeţi.
* conţine o librărie de clase şi interfeţe pentru domenii specifice cum ar fi programarea interfeţelor utilizator (JFC, AWT, Swing), programare distribuită (comunicare TCP/IP, CORBA, RMI etc.)
* permite programarea cu **fire de execuţie** (multithreaded).
* **dinamicitate**.
* este modelat după C şi C++, trecerea de la C / C++ la Java făcându-se foarte uşor.
* face diferenţa între literele mici şi mari (este **case sensitive**).
* permite dezvoltarea aplicaţiilor pentru Internet – crearea unor documente Web îmbunătăţite cu animaţie şi multimedia.
* Development Kit (JDK) este **disponibil gratis.**
  1. **Concluzii**

Tehnologiile folosite în dezvoltarea aplicației „Card Games Scorer” sunt Android Studio IDE v.2.1.2 și Java SE Development 8[[10]](#footnote-10). Întreaga lucrare este scrisă sub API level echivalent versiunii de Android Jelly Bean, ce manifestă funcționalitate pe aproximativ 80% din dispozitive.

Am optat pentru acest mediu de dezvoltare deoarece, pe lândă faptul ca este cel oficial în materie de programare pe platforma Android, el oferă o mulțime de avantaje față de plug-in-ul ADT pentru Eclipse[[11]](#footnote-11).

Alegerea ultimei versiunii de Java, SDK 8, este evidentă. Acest update a adus atât inovație cât și stabilitate pentru dezvoltatori.

1. **Descrierea Aplicației**
   1. **Analiză și proiectare**
      1. **Soluționarea problemei**

Competiția este resimțită tot mai puternic în zilele noastre. Odată cu spiritele aprinse și doritoare de câștig se pot strecura, din păcate, și participanți neserioși (trișori) la numeroasele evenimente provocatoare din viața noastră.

Date aceste fapte, menținerea scorului unor concursuri, competiții sau jocuri amicale poate fi o sarcină cu un nivel ridicat de dificultate. Se pot găsi infinite modalități pentru ca un invidid să își asume câștigul pe nedrept.

Implicând tehnologia introducem un nou tip de trișor în peisaj – supranumit uneori hacker. În realitate acesta poate fi pur și simplu un exploator ce doar a avut puțin noroc și nu chiar calitățile necesare spargerii de coduri sau reguli. Pentru a crea o aplicație care să dețină un schimb de informații sigur, am analizat o multitudine de protocoale de securitate. Acestea funcționează pe diferite principii, dar în principal pe schimb de noțiuni criptate.

Pentru ca acest protocol să poate fi pus în aplicare a fost solicitată o structură de server – client a aplicației. Cum conexiunea prin socketuri de una singură poate provoca dificultăți a fost necesară implicarea unui API pentru a mijloci și stabiliza fluxul de informație.

În concluzie, am survenit cu o idee de rezolvare pentru problema prezentată urmând a fi prezentată mai în detaliu în capitolul care urmează.

* + 1. **Expunerea ideii de rezolvare**

Înainte de a specifica o posibilă soluție a problemei am decis că ar fi util să putem fixa cerințele aplicației „Card Games Scorer”. Prin urmare, regăsim drept funcționalități de bază următoarele:

1. Detectarea altor dispozitive ce rulează aplicația drept server în vederea stabilirii unei conexiuni sau crearea unui astfel de serviciu în cazul în care descoperirea nu returnează nicio legătură posibilă;
2. Determinarea cărei părți părtașă la conexiune îi revine dreptul de start al jocului;
3. Menținerea evidenței scorului într-un mod amical, ambele părți trebuind să aprobe o schimbare a punctajului la un anumit moment dat.

Pe lângă aceste funcționalități principale am realizat posibilitatea de a configura denumirile celor doi participanți la joc, precum și posibilitatea de a salva istoricul punctajelor.

Găsirea dispozitivului ce își însușește meseria de server se realizează cu ajutorul API-ul Android, Network Service Discovery. Folosirea API-ului în aplicație permite utilizatorilor să identifice alte dispozitive la nivel local care suportă serviciul cerut de soft-ul în cauză. În caz contrar, cu ajutorul aceleiași tehnologii se poate transforma aplicația curentă într-un dispozitiv ce înregistrează tipul de serviciu dorit, urmând apoi să aștepte realizarea unei conexiuni.

Entitatea cu drept de start la joc este determinată folosind un protocol de commitment bazat pe criptografie. Fiecare participant alege un bit random 0 sau 1. Folosind o metodă de criptare cu cheie simetrică, amândoi recurg la codificarea acestuia. Clientul trimite bitul codat serverului și vice-versa. Fac schimb și de cheile lor folosite la criptare și decodează bitul oponentului. Alegerea participantului cu drept de start se alege, așadar, prin evaluarea expresiei XOR dintre cei doi biți. 0 indică câștigul serverului, iar 1 pe cel al clientului. Țin să menționez ca prin această metodă este exclusă orice tentativă de trișare a celor doi candidați.

Păstrarea scorului se poate face pe oricare dintre cele două dispozitive. La sfârșitul unei runde sau a unui joc (în funcție de modalitatea de joc aleasă) se cere salvarea modificărilor făcute pe parcurs. În acest moment fiecare dintre cei doi părtași trebuie să își exprime acordul cu noile valori. Prin acest acord nu se garantează eliminarea înșelăciunii, dar se cere o a doua opinie ce poate exclude evenimentele nefericite de păcălire.

* 1. **Arhitectură**

Aplicația a fost gândită având drept suport modelul Server – Client. Uneori astfel de conexiuni pot ridica diverse provocări de stabilitate. Drept soluție am ales folosirea unui API[[12]](#footnote-12) care pune la dispoziția dezvoltatorilor posibilitatea de a crea servicii proprii cu scopul de a conecta dispozitive prin Wi-Fi.

Network Service Discovery (prescurtat NSD în ceea ce urmează) garantează:

1. ușurință de integrare;
2. securitate a datelor;
3. stabilitate în formarea de conexiuni.

La baza folosirii acestui instrument stau doar trei listeneri care utilizați cu succes constituie 90% din funcționalitate. Putem considera astfel ușurința cu care putem dezvolta aplicații prin mijlocul NSD.

Se pot realiza conexiuni numai între dispozitive ce rulează același tip de serviciu și sub același pseudonim. Aceste două informații nu sunt publice pentru utilizatori și pot fi ridica dificultate în a fi descoperite de către aplicații autorizate. Așa putem conchide că sistemul este ridicat peste un spațiu securizat.

Odată pus în practică, API-ul furnizează toate cele necesare formării unei conexiuni prin intermediul socketelor; timpul de execuție restrâns și o legătură asigurată contribuind la stabilitatea sitemului.

Acest serviciu de descoperire constă în trei pași: înregistrare, descoperire și soluționare.

Primul pas îl reprezintă înregistrarea serviciului propriu asupra rețelei de Internet Wireless. Din acest moment alte dispozitive vor putea găsi broadcastul acestui instrument.

În cazul în care vrem să descoperim serivicii deja înregistrate, ne folosim de listenerul de discovery al API-ului. La găsirea unor servicii se poate înainta la pasul de resolve. Dacă ambele părți, atât serviciul cât și clientul rulează sub același tip de conexiune NSD, soluționarea legăturii peste rețea va fi una pozitivă.

După ce pasul de soluționare a avut loc, se pot folosi informațiile prelucrate până în acest moment de API-ul nostru (host și port) în vederea stabilirii unei conexiuni bazată pe comunicarea între sockete.

În figura 1 putem observa mai în detaliu succesiunea evenimentelor ce duce la crearea și stabilirea unui serviciu NSD.

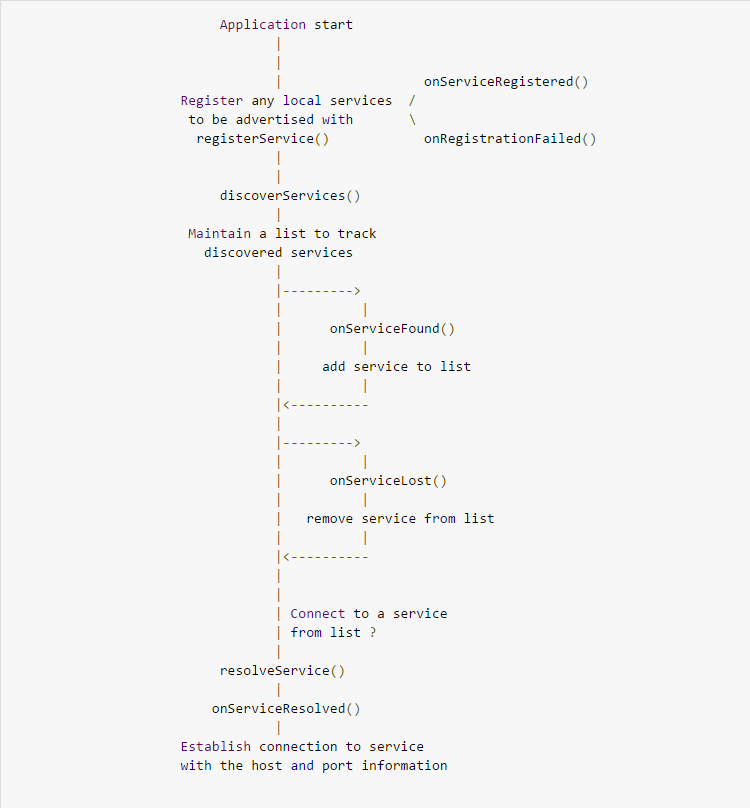


Figura 1: Detalierea ciclului de viață al API-ului

Ulterior serviciului, am folosit un server TPC/IP iterativ pentru a comunica cu clienții. Motivul pentru care am optat pentru un astfel de model de server este natura aplicație; ea va putea înregistra o singura conexiune Server – Client la un anumit moment având în vedere că vizează menținerea scorului în jocuri de câte doi playeri.

Odată activ, serverul așteaptă sosirea requesturile clienților. În momentul în care legătura dintre părțile participante se stabilește, clientul poate iniția startul protocolului de commitment pentru a soluționa dispozitivul ce are dreptul de a fi primul la joc.

Protocolul constă în schimb de informații codate între părtași. Fiecare dintre aceștia vor accesa printr-un task asincron[[13]](#footnote-13) serviciul web Random.org[[14]](#footnote-14) și vor genera astfel câte un bit random (0 sau 1). Acesta va fi codat folosind un algoritm de criptare cu cheie simetrică și trimis părții adeverse. Odată primiți cu succes, clientul și serverul fac schimb și de cheile lor de criptare. Cu ajutorul acestora reușesc să decodifice bitul trimis inițial.

Câștigătorul, sau norocosul ce va avea drept să fie primul, va fi ales în funcție de rezultatul generat de evaluarea expresiei logice XOR[[15]](#footnote-15). În cazul în care valoarea disjuncției este egală cu 0 serverul va fi desemenat primul jucător, altfel, clientul va avea drept de start în caz contrar (expresia va indica valoarea 1).

Pentru a întelege mai ușor modul de funcționare al acestui protocol am recurs la desenarea unui diagrame UML care descrie întregul schimb de informații.

În figura 2 se poate vedea modul în care comunică serverul cu clientul în scopul implementării protocolului de commitment.

Cu ajutorul acestui protocol se elimină una dintre problemele ce pot surveni în timpul alegerii primului jucător în cadrul oricărei competiții. Totul se bazează pe noroc pur și nu există mijloc de a trișa sau de a prezice rezultatul.



Figura 2: Diagramă a protocolului de bit agreement

* 1. **Implementare**

În acest capitol voi descrie în detaliu procesul tehnic care a dus la asamblarea aplicației. Deschiderea în spațiul de cod va fi făcută de schema fișierelor sursă, mai ales organizarea lor în cadrul softului.

În figura 3 se regăsește schema codului sursă pentru soluția aplicaței Secure Bit Exchange.

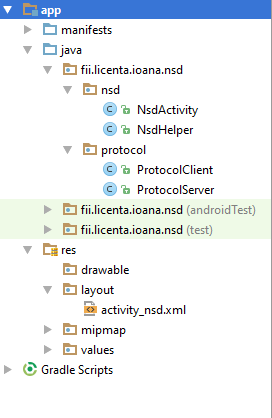


Figura 3: Organizarea codului sursă

Clasele au fost împărțite în două pachete:

* Nsd – conține activitatea de bază și clasa respunzătoare API-ului NSD
* Protocol – include serverul și clientul TCP / IP
  1. **Tutorial**

La rularea aplicației se va decide ce tip de terminal se instanțiază. Serviciul denumit BitExchange este deja existent se va porni terminalul de client, în cazul în care nu se poate găsi instrumentul dorit se rulează aplicația sub modelul serverului.

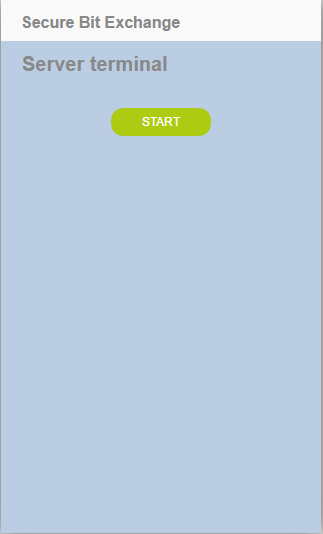
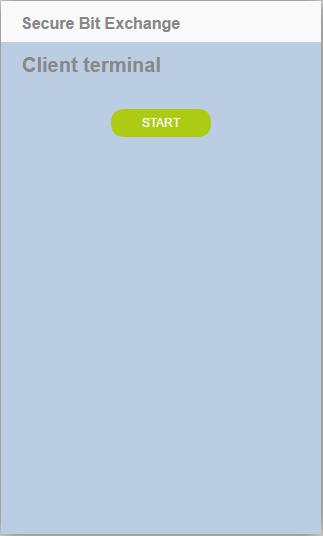
 

Figura 3: Pornirea aplicației

Pe partea de server se inițializeză serviciul BitExchange și se începe broadcastul lui asupra rețelei Wi-Fi.

Pe partea de client este început procedeul de descoperire al serviciului și se pregătește de inițierea conexiunii.

Când acești pași au avut loc, iar soluționarea serviciului a fost una evaluată cu succes, butonul ‘Start’ devine vizibil. Odată apăsat de pe terminalul server, acesta crează serverSocketul și așteaptă conexiunea din partea clientului. Clientul așteaptă și urmează să apese el însuși butonul pentru a realiza legătura între dispozitive.

Când conexiunea a fost una reușită, pe ecran va apărea o secțiune albă ce va surveni ca scop afișajul anumitor informații necesare la buna desfășurare a protocolului.

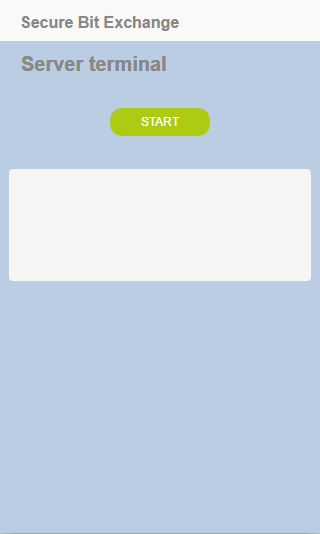
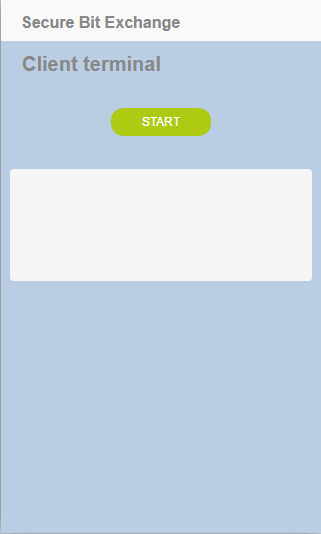
** **

Figura 4: Serviciul setat corect și inițierea conexiunii TCP/IP

Dacă legătura între sockete a fost realizată cu succes, în partea destinată afișării text vor începe să apară rezultatele.

Primul pas îl reprezintă alegerea a unui bit random de către fiecare user, posibilă cu ajutorul API-ului Random.org. Prima linie a ariei de text este rezervată afișării acestui bit.

Se generează o cheie unică de criptare pentru fiecare participant.

În continuare, algoritmul de criptare folosit encodează informația și arată rezultatul ei pe ecran.

Se face schimbul de biți codați și se așteaptă confirmarea utilizatorului.

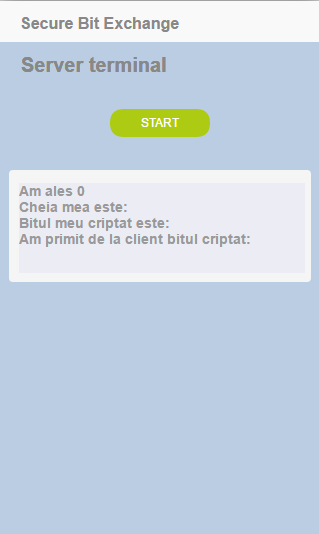
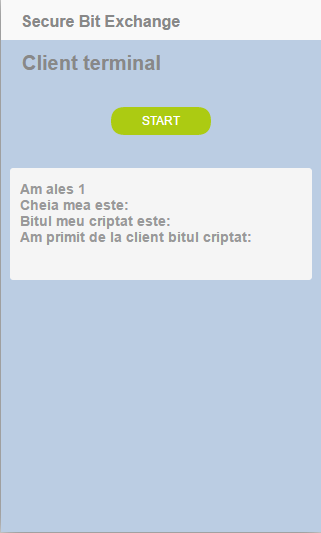
** **

Figura 5: Se realizează schimbul de biți codați

Următorul pas îl reprezintă luarea la cunoștință de ambele părți faptul că schimbul de informații a decurs cu succes și că sunt de acord să trimită reciproc cheile lor secrete necesare în procesul de decriptare.

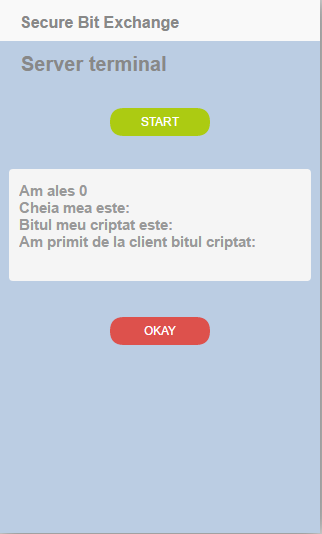
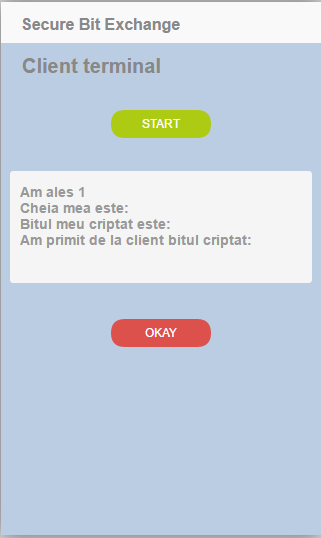
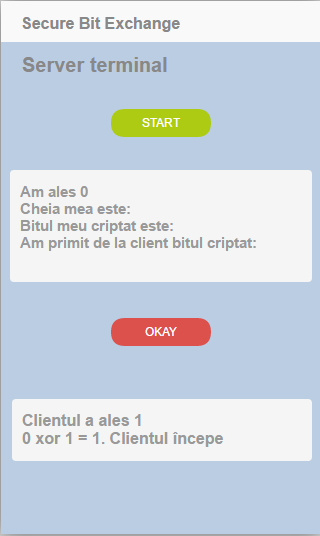
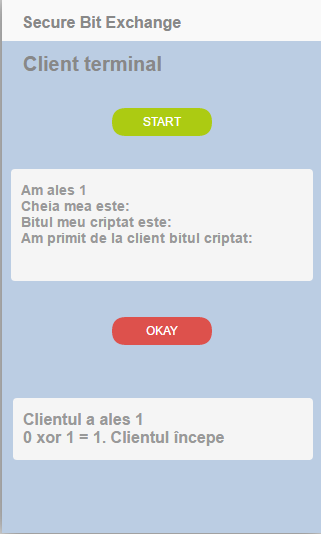
** **

Figura 6: Se așteapă confirmarea utilizatorului

În final, după terminarea transmisiei și închiderea conexiunilor fiecare participant procesează rezultatul expresiei xor între biții aleși. Aceste informații sunt și ele precizate pe ecran, în josul display-ului. Cazul în care s-a ajuns la valoarea 0, serverul este favorabil, clientul având drept de start atunci când rezultatul este 1.

** **

**CONCLUZII**

Obiectivul principal al acestei lucrări a fost căutarea unui mijloc de a angaja oamenii la a se pune de acord în privința unei bune desfășurări a unui joc de cărți. În acest scop au fost necesare soluții anti-trișare și anti-exploatare.

Consider că aplicația a reușit să atingă toate funcționalitățile propuse, implementarea lor ridicând uneori dificultăți.

Un mare hop ce a trebuit depășit încă de la primele linii de cod a fost stabilizarea API-ului NSD și realizarea unei conexiuni stabile. Odată ce aceste două funcții au fost stăpânite totul a decurs mult mai fluent. Am văzut cum implementarea unei criptări relativ simplă poate aduce atât de multe beneficii și poate asigura un spațiu sigur peste care se pot construi o multitudine de alte functionalități.

O posibilă direcție de dezvoltare este crearea unei baze de date care să stocheze informații despre jocurile trecute, incluzând totodată un sistem de logare în aplicație. Se poate extinde și refolosi criptare în a sercuriza procesul de login.

Bineințeles o altă idee de îmbunătățire este aceea de a modulariza aplicația încât să poată susține scorul pentru mai multe tipuri de jocuri, nu numai de cărți. Implementarea unui zar poate fi folositoare în acest scop.

Pentru a fi accesibilă și mai multor persoane care nu cunosc bine limba suportată, s-ar putea include mai multe limbi în componența aplicației cum ar fi Franceză, Italiană, Germană, etc.

Conchidem astfel că tema propusă este una de actualitate, în ton cu tendințele curente și ce lasă loc de oportunități și prosperitate.

**BIBLIOGRAFIE**

**[1] Ctrl-D.ro:** <http://ctrl-d.ro/inspiratie/procesul-de-dezvoltare-a-platformei-android/>

**[2] Ctrl-D.ro:** <http://ctrl-d.ro/tutoriale/hello-android-dezvoltarea-primei-aplicatii/>

**[3] PlayTech.ro:** <http://playtech.ro/2012/istoria-android-de-la-1-0-la-4-0-in-cativa-pasi-simpli-partea-i/>

**[4] PlayTech.ro:** <http://playtech.ro/2012/istoria-android-de-la-1-0-la-4-0-in-cativa-pasi-simpli-partea-i/>

**[5] PlayTech.ro:** <http://playtech.ro/2015/ce-noutati-aduce-android-6-0-marshmallow/>

**[6]** **Code.tusplus.com:** <http://code.tutsplus.com/tutorials/learn-java-for-android-development-introduction-to-java--mobile-2604>

**[7] Wikipedia:** <https://ro.wikipedia.org/wiki/Android_(sistem_de_operare)>

**[8] Wikipedia:** <https://ro.wikipedia.org/wiki/Java_(limbaj_de_programare)>

**[9] Wikipedia:** <https://en.wikipedia.org/wiki/Commitment_scheme>

1. Simple Score Sheet : https://play.google.com/store/apps/details?id=de.einedigitalewelt.spielblock [↑](#footnote-ref-1)
2. 500 Scorer: https://play.google.com/store/apps/details?id=com.antsapps.tennotrumps [↑](#footnote-ref-2)
3. Yatzee Scorer: https://play.google.com/store/apps/details?id=com.beyondit.yatzeescorer [↑](#footnote-ref-3)
4. Android Studio: <https://developer.android.com/studio/index.html> [↑](#footnote-ref-4)
5. IntelliJ IDEA: <https://www.jetbrains.com/idea/> [↑](#footnote-ref-5)
6. Wikipedia: <https://ro.wikipedia.org/wiki/Android_(sistem_de_operare)> [↑](#footnote-ref-6)
7. Adevărul: <http://adevarul.ro/tech/gadget/totul-despre-android-60-marshmallow-aduce-nou-telefoane-primesc-1_56137fd3f5eaafab2c5bfdf5/index.html> [↑](#footnote-ref-7)
8. Android: <https://developer.android.com/index.html> [↑](#footnote-ref-8)
9. Integrated Development Environment guide: <http://mashable.com/2010/10/06/ide-guide> [↑](#footnote-ref-9)
10. Java SE Development Kit Guide: <https://docs.oracle.com/javase/8/> [↑](#footnote-ref-10)
11. Eclipse plugin ADT: <https://developer.android.com/studio/tools/sdk/eclipse-adt.html> [↑](#footnote-ref-11)
12. Application Programming Interface: <https://ro.wikipedia.org/wiki/Application_Programming_Interface> [↑](#footnote-ref-12)
13. Documentație AsyncTask: <https://developer.android.com/reference/android/os/AsyncTask.html> [↑](#footnote-ref-13)
14. Serviciul Random.org bazat pe sunetele din atmosferă: <https://www.random.org/> [↑](#footnote-ref-14)
15. XOR (Disjuncție exclusivă): <https://ro.wikipedia.org/wiki/Disjunc%C8%9Bie_exclusiv%C4%83> [↑](#footnote-ref-15)