UNIVERSITATEA “ALEXANDRU IOAN CUZA” DIN IAȘI

FACULTATEA DE INFORMATICĂ

LUCRARE DE LICENȚĂ

**Desktop la distanta. Controlul unui dispozitiv Windows de la distanta utilizand Android**

propusă de

**Sălceanu Sergiu-Paul**

**Sesiunea:** iunie, 2020

Coordonator științific

**Lect. Dr. Traian Vidrașcu Cristian**

UNIVERSITATEA “ALEXANDRU IOAN CUZA” DIN IAȘI

FACULTATEA DE INFORMATICĂ

**Desktop la distanta. Controlul unui dispozitiv Windows de la distanta utilizand Android**

**Sălceanu Sergiu-Paul**

**Sesiunea:** iunie, 2020

Coordonator științific

**Lect. Dr. Traian Vidrașcu Cristian**

Declaratie originalitate

Declaratie drepturi de utilizare

**Cuprins**

1. Introducere
2. Descrierea Problemei
3. Grad de Noutate
4. Contributii
5. VNC (Virtual Network Computing)
6. RFB (Remote Frame Buffer)
7. RDP (Remote Desktop Protocol)
8. Chromoting, Chrome Remote Desktop
9. CustomRemote
   1. Interfata aplicatiei server si client (butoane yay)
   2. Conexiunea in aplicatie (socket, network discovery)
   3. Preluarea inputului de catre aplicatie client
   4. Emularea inputului de catre aplicatie server
10. Concluzie
11. Bibliografie

**Introducere**

In prezent, exista din ce in ce mai multe de persoane ce detin mai multe dispozitive electronice care au un rol important in viata lor de zi cu zi. Procentajul de persoane ce au acces la un calculator a crescut in ultimii ani, odata cu accesibilitatea acestora. La nivel global, aproape 50% din populatia globului are acces la un calculator. Acest procentaj este mai mare in regiunile mai dezvoltate. Smartphone-urile sunt, in mod similar, din ce in ce mai raspandite. Aceste dispozitive functioneaza individual, dar sunt facute sa poata interactiona unele cu altele. Este clar ca in cazul in care cineva are mai multe dispozitive, va intampina situatii in care ii va fi benefica interactiunea dintre acestea.

Conceptul de desktop la distanta este un tip de interactiune dintre dispozitive. Acesta consta in receptionarea input-ului de la un dispozitiv si transmiterea acestuia la dispozitivul “gazda”, in acelasi timp oferind feedback de la “gazda” sub forma a unui flux video in timp real cu continutul ecranului, sau informatii precum datele despre fisierele prezente pe acesta.

In aceasta lucrare va fi prezentata o aplicatie Android menita sa fie utilizata alaturi de o aplicatie Windows pentru a permite controlul unui calculator Windows la distanta. Deja exista un numar mare de aplicatii de tip “Remote Desktop” care au ca tinta atat Windows, cat si alte sisteme de operare. Aplicatia prezentata difera prin existenta unei modalitati de a crea aranjari de butoane cu functii specificate de utilizator. Astfel, este simulata o “telecomanda personalizata” pentru calculator.

**Descrierea Problemei**

Exista numeroase probleme ale caror solutie beneficeaza din abilitatea de a controla un calculator de la distanta. Desi numarul de aplicatii ce ofera aceasta posibilitate este mare, fiecare are avantaje si dezavantaje in functie de scopul utilizarii acesteia. Astfel, nu exista o solutie care sa fie superioara altora din toate punctele de vedere.

* Depanarea, sau “troubleshooting-ul”, este un process unde “Remote Desktop” este utilizat frecvent. Cineva poate prelua controlul calculatorului unui utilizator ce are o problema fara a fi nevoie prezenta acestuia la locatia dispozitivului. Acest tip de acces il ajuta in a rezolva problemele mai repede decat ar putea prin explicarea pasilor ce trebuie urmati celui din fata calculatorului.. Chiar si simpla partajare a ecranului poate fi de ajutor in astfel de situatii.
* Performanta dispozitivului controlat poate fi un motiv pentru utilizarea “Remote Desktop”. Pot fi folosite aplicatii ce necesita putere mare de procesare pe masina “server”, in timp ce utilizatorul interactioneaza cu ele de la distanta. Astfel, este crescut gradul de portabilitate al calculatorului, in functie de conexiunea dintre cele doua dispozitive.
* Dispozitivul ce controleaza calculatorul “gazda” poate lua locul unui dispozitiv periferic de intrare. Astfel, se poate simula input de la mouse si tastatura fara ca acestea sa fie conectate la calculator. In plus, se pot utiliza functionalitati prezente doar la acel dispozitiv. De exemplu, daca este un smartphone sau tableta, se vor putea folosi controale touch. In cazul in care telefonul este controlat de pe calculator, se pot efectua operatii precum raspunderea la un apel fara folosirea directa a telefonului.
* Un mod de utilizare asemanator cu cel al depanarii este colaborarea la un proiect in timp real. Daca cineva se afla la calculatorul gazda si cineva il controleaza prin “Remote Desktop”, se poate coopera pentru realizarea unui proiect. Totusi, aceasta problema este rezolvata mai eficient de editoare colaborative in timp real (ex. Google Docs).

**Grad de Noutate**

Tehnologia necesara pentru “Remote Desktop” a existat de mult timp, deci exista un numar mare de aplicatii ce se ocupa de acest tip de conexiune. O componenta a acesteia este partajarea ecranului, existenta in cadrul multor aplicatii cu alt scop principal (ex. Skype, Zoom).

Pentru administrarea dispozitivelor la distanta, un factor important este securitatea. Astfel, solutiile ce permit controlul dispozitivelor dincolo de reteaua locala au, in general, traficul securizat (ex. TeamViewer se utilizeaza de un schimb de chei RSA, si sesiunile sunt criptate in AES-256; Chrome Remote Desktop are sesiuni criptate prin SSL).

O zona unde viteza de raspuns si calitatea imaginii sunt importante este cea de Game Streaming. Recent, au aparut numeroase servicii ce ofera abilitatea de a controla un dispozitiv de la distanta pentru a juca jocuri. Astfel, cerintele pe care trebuie sa le indeplineasca utilizatorul nu mai sunt cerintele de sistem ale jocului, ci o conexiune la internet stabile si plata serviciului ce permite accesul la un dispozitiv cu puterea de procesare grafica necesara. Servicii ce ofera Game Streaming includ Google Stadia, GeForce Now, Shadow. Un serviciu asemanator este Steam Remote Play, doar ca acesta utilizeaza calculatorul utilizatorului ca “gazda”, deci nu ofera decat conexiuna dintre doua dispozitive detinute de utilizator. Steam Remote Play poate fi folosit, de asemenea, pentru a permite cuiva sa joace ceva ce are doar multiplayer local pe internet cu cineva care primeste control asupra calculatorului sau.

Dispozitivele Android au, de asemenea, multe optiuni pentru controlul desktopului la distanta. Acestea includ Microsoft Remote Desktop, Chrome Remote Desktop, TeamViewer. Pe langa acestea, exista un numar mare de aplicatii pe Magazin Play.

Diferenta dintre aplicatia prezentata si cele deja existente e abilitatea de a crea aranjari de butoane cu diferite instructiuni, in locul utilizarii unui sablon pre-existent. Astfel, poate functiona ca un panou de control pentru combinatii de taste, crescand eficienta utilizatorului.

**Contributii**

* Studierea limbajului C#
* Studierea library-ului Windows Forms
* Studierea sectiunii Windows API ce se ocupa cu inputul utilizatorului
* Studierea limbajului Java
* Studierea platformei Android si API-ului oferit de aceasta
* Crearea unei conexiuni server-client intre Windows si Android

# VNC (Virtual Network Computing)

VNC (Virtual Network Computing) este o tehnologie ce permite utilizatorilor sa vada si sa interactioneze cu un desktop printr-un program server folosind un program client pe alt calculator. VNC poate fi utilizat atat pe Windows, cat si pe Linux. Protocolul VNC utilizeaza RFB (Remote Frame Buffer) - transmite mesaje de tip eveniment de la client la server si primeste feedback grafic de la ce consta in date despre pixelii de pe ecran. Serverul VNC utilizeaza in mod implicit portul 5900. In functie de informatiile de pe ecran, e posibila nevoia unei viteze mari de transfer de date, fiindca aceste sectiuni sunt retrimise doar daca difera fata de cadrul anterior. Astfel, in cadrul unei interfete grafice, nu se vor transmite multe informatii grafice, ferestrele fiind statice. Pe de alta parte, transmiterea unui film va necesita retrimiterea unui numar ridicat de date pentru aproape fiecare cadru.

# RFB (Remote Frame Buffer)

RFB (Remote Frame Buffer) este protocolul utilizat pentru transmiterea informatiilor grafice in cadrul VNC. RFB nu este un protocol complex. Versiunea initiala a fost una simpla, dar au fost adaugate functionalitati precum transferul de fisiere si tehnici mai eficiente de compresie a fisierelor si securitate pe parcursul evolutiei acestuia. Fiind open-source, exista si numeroase extensii neoficiale ale acestuia.

Imediat dupa conexiunea client-server, e transmisa o secventa de mesaje handshake pentru a determina versiunea RFB utilizata, tipul de securitate utilizat, si verificarea unei eventuale parole. Intai, se trimite un mesaj de tip ProtocolVersion de la server la client, semnificand cea mai recenta versiune a RFB de pe server. Clientul raspunde cu un mesaj similar, care decide versiunea utilizata. Acesta reprezinta cea mai apropiata versiune de cea specificata de server, dar nu va fi utilizata una mai recenta decat cea oferita de server. Un exemplu de mesaj ce specifica o versiune este:

RFB 003.008\n

Mesajul e trimis in bytes si interpretat ca un string de caractere ASCII, unde grupele de cifre reprezinta numerele majore si minore de versiune. Mesajul dat ca exemplu indica RFB versiunea 3.8.

Pentru a decide nivelul de securitate al conexiunii, serverul trimite un array cu tipurile de securitate disponibile, si clientul un singur byte care indica cel ce va fi utilizat. In urma acestui procedeu, se procedeaza in functie de cerintele acelui tip de securitate.

In final, sunt trimise mesaje care specifica daca serverul sa deconecteze ceilalti clienti RFB, si serverul specifica informatii precum latimea si lungimea ecranului in pixeli, si numele serverului.

Partea de display a protocolului este administrata de un singur tip de instructiune, care cere actualizarea pixelilor dintr-o zona dreptunghiulara specificata. Cantitatea de date transmisa depinde, astfel, de compresia datelor si de modul in care sunt selectate aceste regiuni. Viteza de transmitere a cadrelor deponde de conexiunea clientului, fiindca nu sunt trimise date grafice decat in urma cerintei explicite de la un client. Nu sunt trimise actualizari nesolicitate ale ecranului.

Celelalte evenimente de tip input sunt bazate pe un model standard ce include o tastatura si un mouse cu mai multe butoane. Este detectat input-ul de la dispozitivele periferice ale client-ului, si emulat pe server.

Mesajele client-to-server disponibile includ:

* SetPixelFormat: Specifica formatul in care sa fie trimisa informatia despre pixeli in urma mesajelor de tip FramebufferUpdateRequest.
* FramebufferUpdateRequest: Notifica serverul ca un client este interesat de o anumita zona de ecran. In mod normal, serverul raspunde trimitand un mesaj de tip FramebufferUpdate. Uneori, acelasi mesaj poate fi trimis ca raspuns la mai multe mesaje de tip FramebufferUpdateRequest. Un parametru specifica daca clientul cere o actualizare a informatiei din regiunea specificata sau daca le-a pierdut. In cazul in care le-a pierdut, serverul va trimite informatii chiar daca acestea nu s-au modificat.
* KeyEvent: Specifica apasarea sau finalul apasarii unei taste. Tastele sunt specificate folosind valori “keysym” definite de X11, un protocol de afisare de interfete grafice. De exemplu, literele mici de la a la z au valori de la 0x0061 la 0x007a. Pot aparea complicatii atunci cand sunt utilizate aranjari diferite ale tastaturii. Daca un caracter necesita apasarea tastei Shift la client, dar nu la server, atunci serverul va trebui sa ignore tasta Shift pentru un rezultat corect. De exemplu, caracterul ‘#’ necesita apasarea tastei Shift in formatul US, dar nu in formatul UK. Trebuie de asemenea tratate special cazurile tastelor de tip modificator (Control, Shift, Alt) in cazul combinatiilor de taste.
* PointerEvent: Specifica miscarea pointerului, sau apasarea sau eliberarea unui buton. Sunt oferite coordonatele noi ale pointer-ului, si un byte unde fiecare bit specifica starea butonului cu indicele acestuia. Scrollwheel-ul mouse-ului este controlat de indicii 4 si 5. Apasarea si eliberarea unuia dintre acestia reprezinta miscarea scrollwheel-ului un pas.
* ClientCutText: RFB suporta sincronizarea textului din “cut buffer” dintre client si server. Acesta e reprezentat de selectia ce poate fi mutata de la o fereastra la alta, fara a folosi clipboard-ul. Prin sincronizarea acestuia, se poate muta text dintr-o fereastra de pe dispozitivul client la una de pe dispozitivul server. Un dezavantaj este ca nu se poate copia decat text din setul de caractere Latin-1.

Mesajele server-to-client disponibile includ:

* FramebufferUpdate: Contine o secventa de regiuni rectangulare cu date despre pixeli, trimisa doar ca raspuns de la un FramebufferUpdateRequest.
* SetColorMapEntries: Contine un “color map” ce asociaza valorile pixelilor trimise de server unor culori. Astfel, se poate utiliza un numar variat de culori.
* Bell: Daca exista unul in implementarea clientului, acesta va face un semnal audio.
* ServerCutText: Componenta server pentru sincronizarea textului din “cut buffer”.

Codificarea de pixeli ce poate fi utilizata de orice client RFB este “Raw”, unde informatii individuale despre fiecare pixel sunt transmise de la stanga la dreapta, de sus in jos. Este transmis un array de tip PIXEL, deci de dimensiuni egale cu produsul dintre latime, lungime, numarul de octeti ocupati de PIXEL Alte codificari includ:

* CopyRect: Determina cand aceleasi date sunt prezente intr-o pozitie diferita. Clientului ii sunt trimise doar coordonate x si y, si acesta preia informatiile din cadrul anterior. CopyRect e util atunci cand o fereastra este mutata, sau cand este utilizata o bara scroll.
* RRE: Partitioneaza regiuni dreptunghiulare in subregiuni dreptunghiulare cu pixeli de o singura culoare. Este transmisa o valoare Vb, care reprezinta pixelul ce apare cel mai des in regiune. Apoi, e transmis N, urmat de N subregiuni dreptunghiulare. Fiecare regiune contine o tupla (v,x,y,w,h). v difera de Vb si e valoarea pixel din regiune, x, y, w, h sunt utilizate pentru a determina pozitia si dimensiunea regiunii pe ecran. Astfel, pentru afisarea iamginii, clientul poate umple o regiune cu pixelul Vb, si umple fiecare subregiune cu pixelul v corespunzator.
* Hextile: Partitioneaza regiuni dreptunghiulare in subregiuni 16 x 16. In cazul in care regiunea nu are dimensiunile divizibile cu 16, subregiunile din randul si coloana finala for fi mai mici. Este aplicat RRE pe fiecare subregiune. Similar cu RRE, exista o valoare de pixel de fundal. Totusi, aceasta nu este universala pentru toata regiunea, ci e transmisa de la o subregiune 16 x 16 la alta. Aceasta poate fi suprascrisa de un parametru BackgroundSpecified al subregiunii. Exista de asemenea un parametru ForegroundSpecified care determina culoarea oricarei regiuni generate de RRE pe subregiune.
* TRLE: Partitioneaza regiuni in mod similar cu Hextile, dar foloseste un tip CPIXEL in loc de PIXEL, memorand culoarea intr-un format mai compact. Inainte de fiecare subregiune, este trimis un byte care determina dimensiunea paletei de culori folosita. Exista valori speciale care indica reutilizarea paletei de culori a ultimei regiuni care avea una.
* ZRLE: Asemanator cu TRLE, dar utilizeaza compresie zlib, subregiunile iau dimensiunea de 64 x 64 pixeli, si nu se pot reutiliza palete de culori intre acestea.

**RDP (Remote Desktop Protocol)**

Remote Desktop Protocol este un protocol proprietar creat de Microsoft. Acesta ofera o interfata grafica pentru conexiunea la un alt calculator. Se utilizeaza implicit portul 3389. Componenta server este o parte a sistemului de operare Windows, dar este oferit un API printr-o multime de apeluri de functii catre Wtsapi32.dll. Astfel, pe langa o suita de aplicatii client oficiale create de Microsoft (pentru Windows, Android, macOS, iOS), poate fi creata o aplicatie client de alt dezvoltator. In mod similar cu VNC, in cadrul Remote Desktop Services, majoritatea procesarii este realizata pe dispozitivul gazda. Un utilizator poate accesa fisiere, resurse, si executa aplicatii prin intermediul aplicatiei client. Pot exista mai multi clienti simultan la un singur server. Traficul dintre client si server este criptat folosind RC4. Un utilizator poate sa se deconecteze dintr-o sesiune de desktop la distanta fara a o finaliza. Astfel, poate sa o continuie fara vreo interventie de la dispozitivul server, in cazul in care e intrerupta neasteptat de la o problema de conexiune. Alte functionalitati ale RDP includ sincronizarea clipboard-ului, abilitatea de a imprima documente folosind o imprimanta de pe dispozitivul client, si functionalitati de reducere a traficului prin crearea unui cache de bitmap-uri.

**Chrome Remote Desktop, Chromoting**

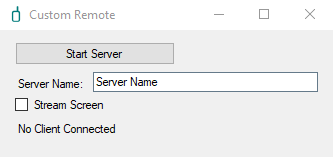
Chrome Remote Desktop, si protocolul proprietar Chromoting difera fata de cele prezentate anterior prin faptul ca atat aplicatia client cat si aplicatia server utilizeaza ca si interfata un browser de tip Chromium. Traficul se realizeaza pe porturile 443 si 3478. Este necesar ca utilizatorul sa detina un cont de Google pentru a utiliza atat aplicatia client cat si aplicatia server, si nu pot fi transmise fisiere intre dispozitive prin intermediul conexiunii. Codificarea ecranului este realizata utilizand VP8, un format de compresie video achizitionat de Google. Acesta e utilizat si in cadrul fisierelor video WebM.

**CustomRemote**

CustomRemote este alcatuit dintr-o aplicatie server Windows, si o aplicatie client Android. Acesta permite trimiterea de input simplu catre un calculator din reteaua locala de la un dispozitiv Android.

**Interfata CustomRemote**

Serverul CustomRemote utilizeaza Windows Forms pentru a afisa informatii minimale despre starea serverului si conexiunea cu clientul.



Serverul poate fi pornit si oprit manual, si i se poate oferi un nume pentru a il diferentia de alte servere in aplicatia Android. De asemenea, poate fi minimizat in bara de notificari a Windows.

In aplicatia Android exista o sectiune dedicata conexiunii la server, iar restul categoriilor au rolul obtinerii de input pentru a fi trimis acestuia.

**Conexiunea dintre client si server**

Aplicatia utilizeaza un socket cu port 5800 pentru transmiterea instructiunilor de la client la server, si un socket cu port 5600 pentru a facilita abilitatea de a fi descoperita de catre client pe reteaua locala. Nu este nevoie ca IP-ul serverului sa fie introdus in aplicatia Android, deoarece contine o functionalitate ce permite descoperirea serverelor de pe reteaua locala. In urma gasirii unui socket deschis la portul 5600, IP-ul sau va aparea in lista de servere, alaturi de numele trimis in urma conexiunii. Aceasta va fi, apoi, inchisa. Modul in care sunt descoperite IP-urile serverelor este prin inlocuirea ultimului octet al IP-ului local al clientului Android cu fiecare dintre celelalte 255 de posibilitati. Aplicatia nu a fost testata in situatii in care exista retele locale cu mai mult de 255 de dispozitive. Dupa descoperirea sa, un server poate fi adaugat intr-o lista de favorite, pentru a nu fi necesara cautarea sa din nou atunci cand este dorita reconectarea.

Odata conectat la server, clientul trimite siruri de caractere de forma:

INSTRUCTIUNE:PARAM1,PARAM2,...

Unde “INSTRUCTIUNE” reprezinta numele comenzii, urmat de doua puncte si parametrii acesteia separati prin virgula. Un exemplu de comanda este “MD”, care deplaseaza pointerul din pozitia curenta la distanta specificata de cei doi parametri ce reprezinta coordonatele. De exemplu:

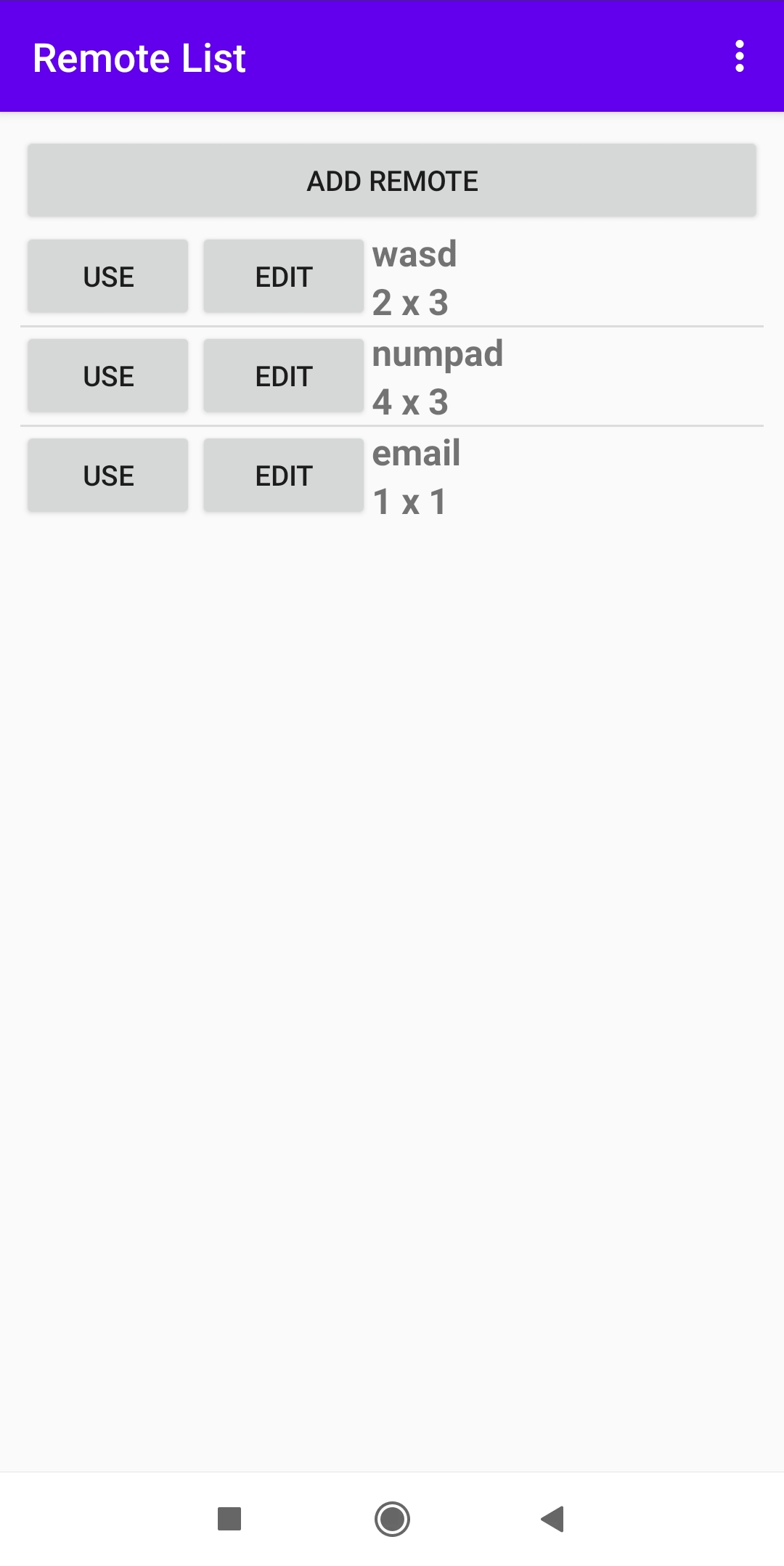
MD:10,20

Instructiunea urmatoare deplaseaza mouse-ul cu 10 pixeli pe axa x, si 20 de pixeli pe axa y. In cazul unei pozitii invalide (coordonate finale negative sau mai mari decat dimensiunea ecranului) pointerul va fi mutat in cea mai apropiata pozitie valida.

**Preluarea inputului de la client**

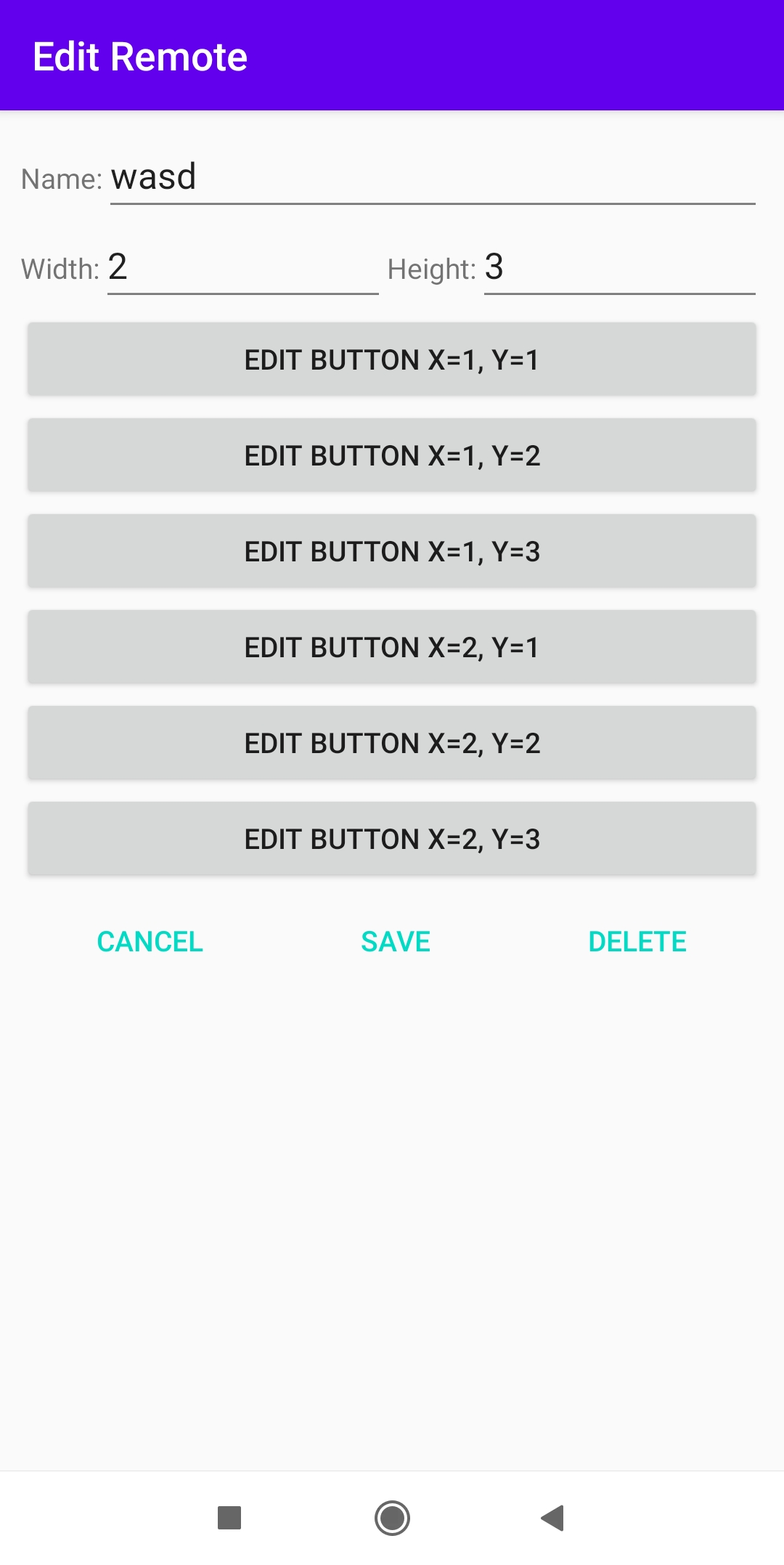
Inputul de la client poate fi preluat din sectiunea ce contine o simulare de trackpad, si sectiunea remotes, unde are posibilitatea de a crea aranjari de butoane personalizate. Simularea de trackpad urmareste inputul touch al utilizatorului, si compara diferenta dintre pozitia acestuia in momente diferite. Aceasta diferenta este trimisa la server si este utilizata pentru a deplasa pointerul in aceeasi directie, la distante proportionale. Inputul text este preluat folosind o casuta unde poate fi introdus text. Sunt monitorizate schimbarile de text si trimise caracterele introduse catre aplicatia server.

Interfata pentru “remote-uri” personalizate este urmatoarea:



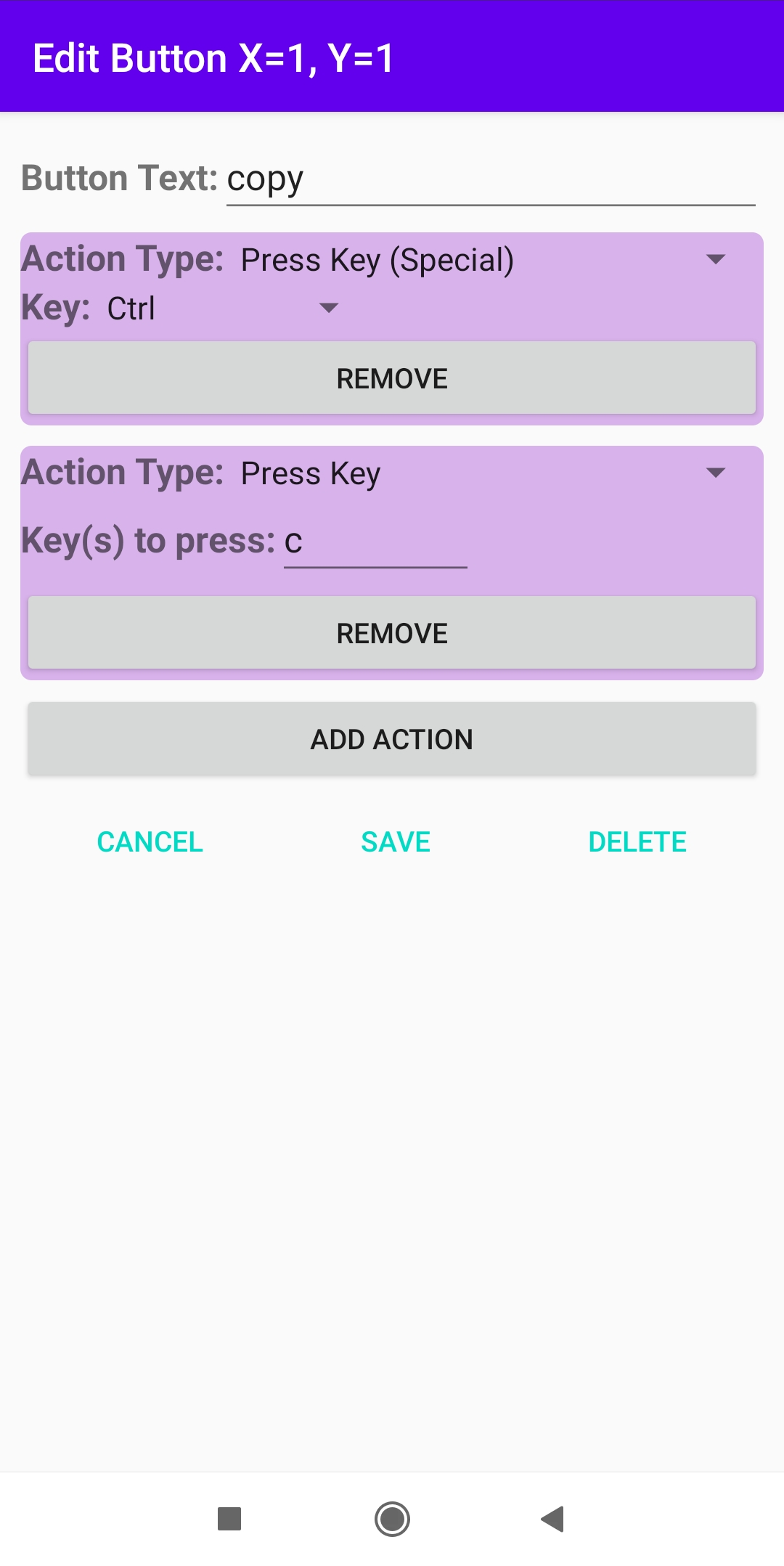
De aici, utilizatorul nu poate trimite input catre server propriu-zis, dar poate accesa si modifica remote-uri unde poate face acest lucru. Este utilizat API-ul SharedPreferences din Android pentru memorarea informatiilor despre remote-urile create, in fisiere JSON. De aici, se pot crea remote-uri noi, cerandu-se un nume, o latime si o inaltime. Aceste proprietati sunt vizibile din lista si pot fi modificate ulterior.

Interfata pentru editarea unui remote este urmatoarea:



Aici, sunt generate in mod dinamic butoane ce conduc la setarile fiecarui element al remote-ului. In timpul utilizarii, butoanele sunt amplasate intr-un tablou bidimensional. In interfata editarii, acestea pot fi identificate folosind coordonatele lor. In cazul in care unui buton nu este asignata macar o actiune, acesta nu va aparea in remote. Momentam, aplicatia are o limita superioara de 20 pentru latime sau inaltime. In urma imbunatatirii interfetei ce contine lista de butoane, s-ar putea creste aceasta limita. Momentan, modificarea butoanelor devine dificila atunci cand sunt intr-un numar mare. Un remote poate avea, teoretic, pana la 400 de butoane. In cazul in care utilizatorul doreste mai multe functii, poate crea un nou remote, numarul acestora neavand o limita superioara.

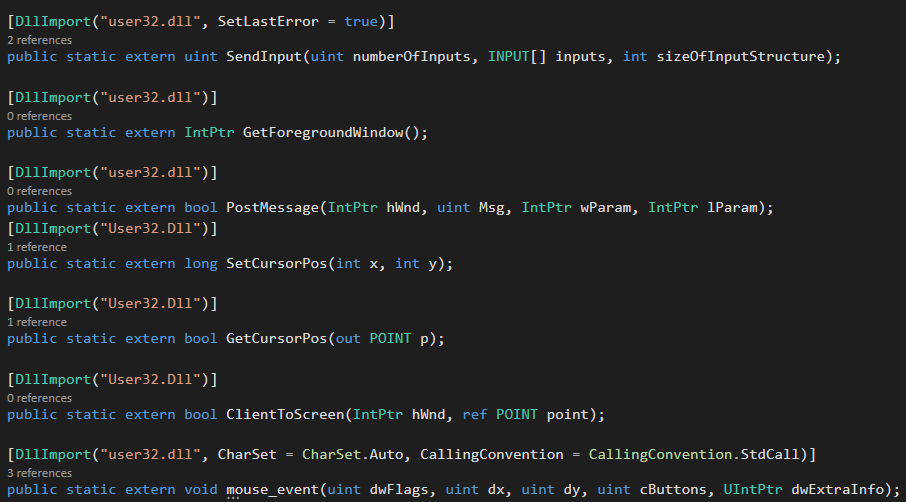
Interfata pentru editarea unui buton este urmatoarea:



Poate fi configurat textul ce apare pe buton in timpul utilizarii remote-ului. Sub acesta exista o lista de actiuni. In urma apasarii butonului, aceste actiuni vor fi efectuate in ordine secventiala. Pot fi configurate apasari ale tastelor si utilizarea mouse-ului. De asemenea, pot fi introduse intreruperi (“Action Type: Wait”). Astfel, se pot configura pauze fixe de timp intre input (ex. Clickuri cu pauze de exact o secunda intre ele).

**Utilizarea inputului de catre server**

Odata ce serverul parseaza mesajele trimise de catre client, apeleaza functii din Windows API corespunzatoare instructiunii dorite. Pentru utilizarea acestora in cadrul unei aplicatii C# in Windows Forms, e necesara importarea individuala a metodelor dorite. Acestea se gasesc in USER32.DLL. Majoritatea continutului acestei biblioteci consta in metode ce asista la crearea unei interfate grafice Windows, dar exista si un numar mare de metode ce permit emularea input-ului utilizatorului.



Declararea metodelor se realizeaza folosind atributul DllImport.

**Concluzie**

In ciuda existentei a mai multor protocoale de comunicare pentru desktop la distanta, in cadrul dezvoltarii unei aplicatii trebuie tratate un numar mare de cazuri particulare, mai ales in situatiile in care diferentele dintre dispozitivele client si server sunt mari. O alta dificultate in crearea unei astfel de aplicatii apare cand trebuie gestionate situatiile in care calitatea conexiunii scade, sau distanta dintre dispozitive este extrem de mare. Timpul ridicat de raspuns poate face mult mai grea utilizarea eficienta a dispozitivului gazda.

**Bibliografie**

* <https://tools.ietf.org/html/rfc6143>
* <https://developer.android.com/reference>
* <https://docs.microsoft.com/en-us/windows/win32/termserv/remote-desktop-protocol>
* <https://winprotocoldoc.blob.core.windows.net/productionwindowsarchives/MS-RDPBCGR/%5bMS-RDPBCGR%5d.pdf> - [MS-RDPBCGR]: Remote Desktop Protocol: Basic Connectivity and Graphics Remoting