28.01.2017r.

Karina Szepczyńska
Michał Krawczyk

Sprawozdanie z Podstaw Teleinformatyki

Informatyka, studia niestacjonarne, sem. VII

Repozytorium: *https://github.com/SzepczynskaKrawczyk/external-ip-monitor*

I. Opis zadania.

Stworzenie aplikacji umożliwiającej wykorzystanie monitora podłączonego do innego komputera jako ekranu rozszerzającego pulpit jednostki głównej.

II. Założenia realizacyjne.

System operacyjny komputera głównego, któremu rozszerzamy pulpit, to Linux (*Ubuntu 16.10* z zainstalowanym oprogramowaniem *Java 1.8, xrandr, maim* oraz biblioteką *libx11-protocol-other-perl*), a drugiej stacji roboczej to Windows (*Windows 10* z zainstalowanym oprogramowaniem *.NET Framework 4.5.2*).

Komunikacja pomiędzy jednostkami PC odbywa się przy użyciu sieci TCP/IP.

III. Napotkane problemy i ograniczenia.

Początkowym założeniem była implementacja zadania wyłącznie na systemie Windows, jednakże po wnikliwej analizie zauważyliśmy, że wiąże się to z napisaniem własnego sterownika typu *Filter-Hook Driver* w modelu *WDDM* (Windows Display Driver Model), co niestety nie jest w zasięgu naszych możliwości czasowych i implementacyjnych.

Rozszerzenie pulpitu roboczego osiągnęliśmy wykorzystując wbudowany w system Linux program *xrandr*. Niesie to za sobą ograniczenie w postaci ciągłego pulpitu na dwóch monitorach (w przypadku uruchomienia jakiejkolwiek aplikacji w trybie pełnego ekranu, pojawi się ona na obydwu monitorach). Dodatkowo, na zewnętrznym monitorze wyświetla się obszar o wysokości ekranu głównego.

Pierwsze próby implementacji aplikacji typu serwer na stacji z systemem operacyjnym Linux, wykazały problem z generowaniem zrzutu ekranu z zawartym kursorem (dodatkowo - z wszystkimi typami kursorów, np *I-Beam*). Zarówno C# (jego implementacja na Linuxie - *Mono*) jak i Java nie posiadają wbudowanych mechanizmów, które pozwoliłyby to osiągnąć. W pierwszym podejściu znaleźliśmy program *shutter*, który spełniał nasze wymagania. Generował on poprawne zrzuty z zawartym kursorem (wymagało to doinstalowania pakietu *XFixes*), aczkolwiek wydajność stała na bardzo niskim poziomie. Wygenerowanie jednego pliku trwało około *1300* milisekund. Ostatecznie zmieniliśmy program na *maim*, dla którego stworzenie zrzutu ekranu trwa około *24* milisekundy.

Kolejną próbą zwiększenia płynności zastosowanego rozwiązania, było

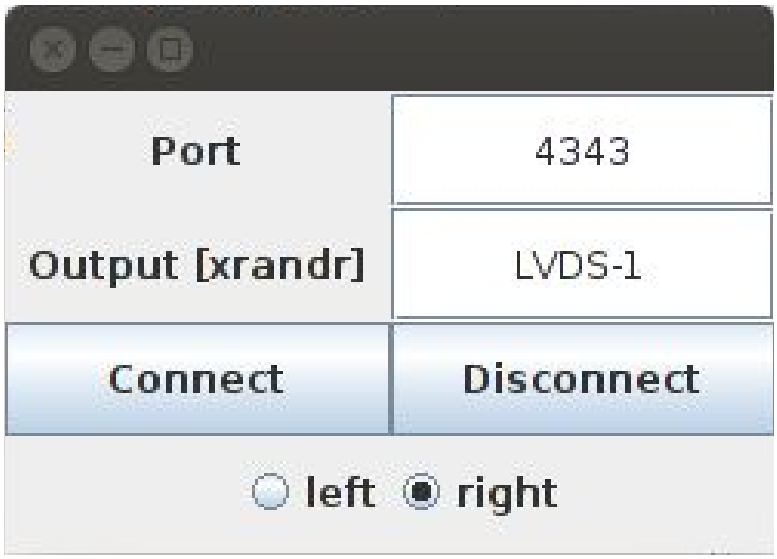
zaproponowane przez Prowadzącego projekt, wysyłanie różnicy pomiędzy aktualnie wykonanym zrzutem ekranu i poprzednio wysłanym do klienta. Metoda ta okazała się jednak mniej wydajna niż wysyłanie pliku w formacie *JPG* - cały plik zajmował średnio *60kB* a różnica w formie tekstowej skonwertowana do tablicy bajtów około *400kB* (również wysyłanie pliku *PNG* zawierającego tylko różniące się piksele było wolniejsze).

IV. Opis działania programu.

Projekt składa się z dwóch programów: jeden to uruchamiany na Linuxie serwer, a drugi to uruchamiany na Windowsie klient. Po poprawnym połączeniu *socketów*, klient wysyła do serwera swoją rozdzielczość w formacie *WIDTH*x*HEIGHT*. Następnie serwer po otrzymaniu danych, przetwarza je i wykonuje odpowiednie polecenia programu *xrandr* w celu ustawienia wymaganej rozdzielczości, która zapewni pełen obszar roboczy dla monitora wbudowanego i monitora zewnętrznego (biorąc pod uwagę wybrane ustawienie czy monitor zewnętrzny ma być po lewej czy po prawej stronie). Po wykonaniu tych czynności serwer i klient wchodzą w pętlę nieskończoną, którą można przerwać klikając *Disconnect* w interfejsie aplikacji na Linuxie. W pętli tej serwer generuje zrzut ekranu, który następnie wysyła do klienta w postaci listy bajtów. Aplikacja na Windowsie odbiera dane i wyświetla je w odpowiednim oknie, które zostało tak zaprojektowane aby zajmowało cały ekran i nie pokazywało kursora myszki pochodzącego z systemu klienta.

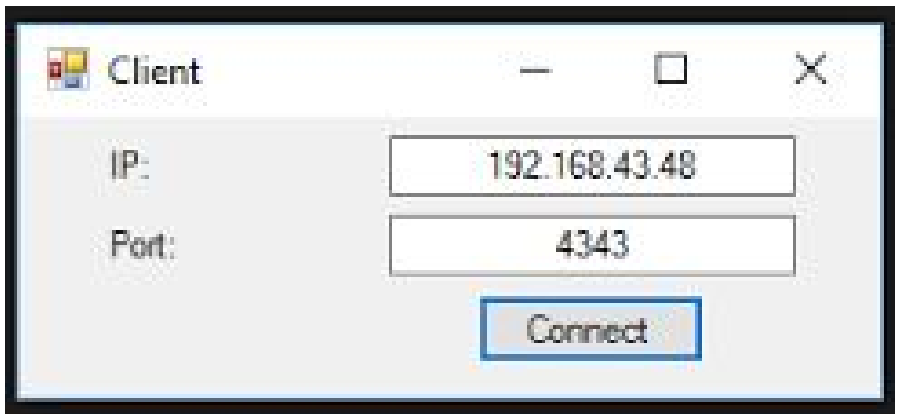
V. Użytkowanie aplikacji.

W celu uruchomienia rozszerzania ekranu dla danego komputera, należy podłączyć komputer do wspólnej sieci Wi-Fi i uruchomić na nim aplikację napisaną w języku Java. Następnie należy wykonać z *Terminala* polecenie *xrandr* i sprawdzić jaką nazwę posiada aktualnie podłączony do komputera monitor. Znalezioną nazwę należy wpisać w oknie aplikacji w miejscu *Output [xrandr]* (*Rysunek 1)*. Oprócz tego możemy wybrać port na jakim będą komunikować się urządzenia oraz czy zewnętrzny monitor ma być po lewej czy po prawej stronie urządzenia. Ostatnim krokiem na Linuxie jest naciśnięcie przycisku *Connect*.



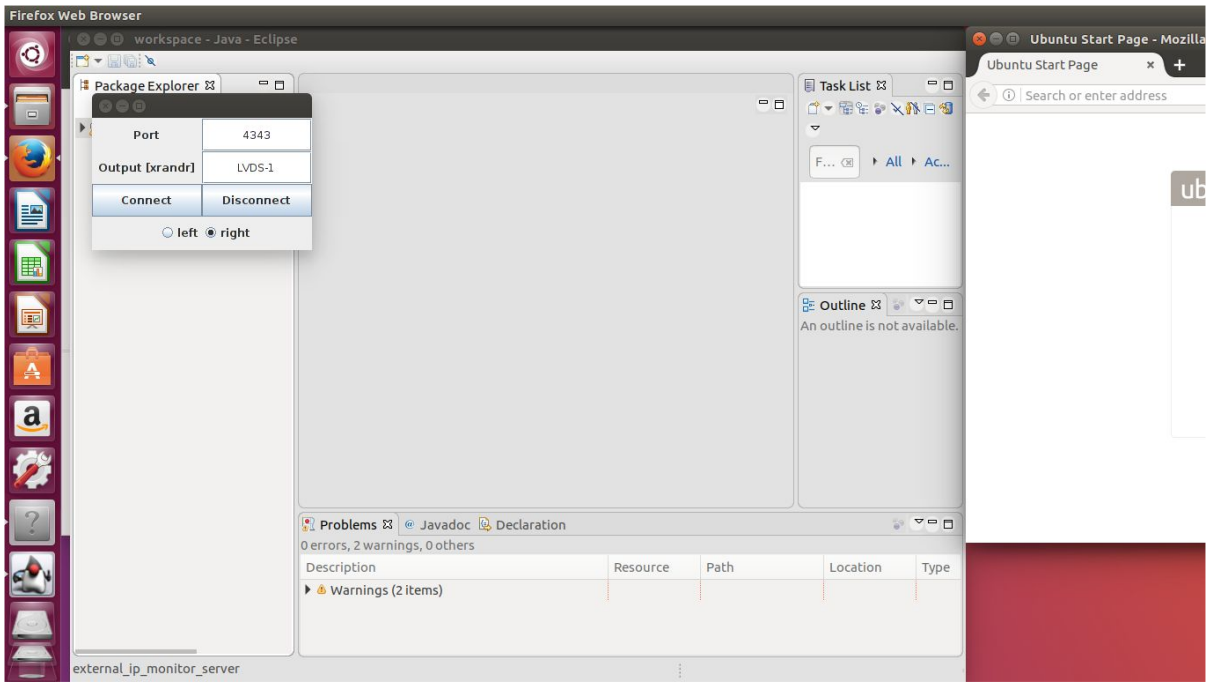
Rysunek 1 Okno aplikacji serwera

Na Windowsie należy podłączyć komputer do wspólnej sieci Wi-Fi i uruchomić aplikację napisaną w C#. W wyświetlonym oknie wpisujemy adres *IP* komputera głównego oraz port do komunikacji (*Rysunek 2)*. Ostatnim krokiem jest naciśnięcie przycisku *Connect*.

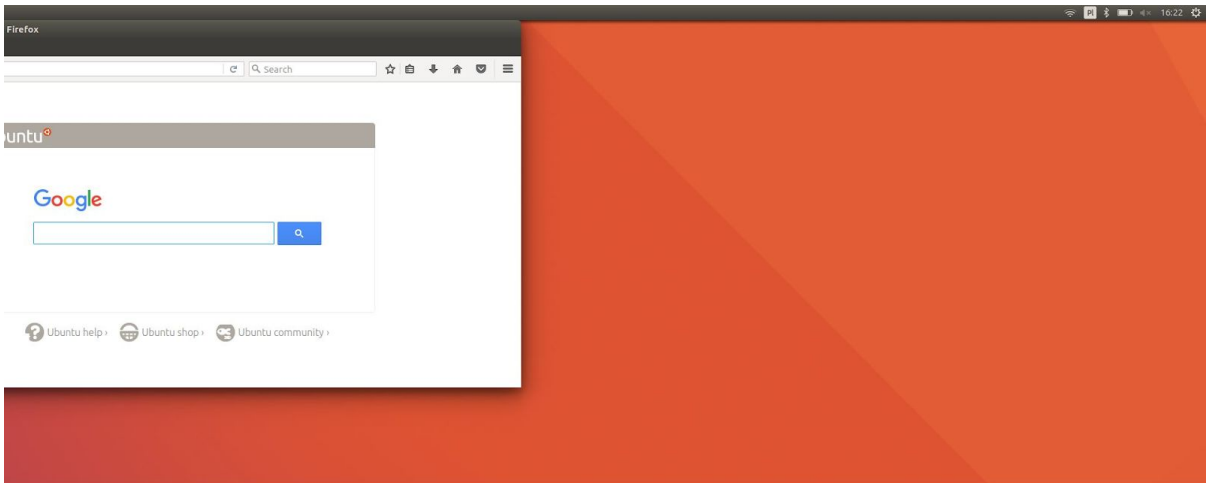


Rysunek 2 Okno aplikacji klienta

Rysunki 3 i 4 przedstawiają zrzuty obrazujące rozszerzony pulpit.



Rysunek 3 Lewa część ekranu - Linux



Rysunek 4 Prawa część ekranu - Windows

W celu zakończenia rozszerzania ekranu należy nacisnąć przycisk *Disconnect* na serwerze, co spowoduje powrót rozdzielczości do stanu sprzed uruchomieniem aplikacji oraz rozłączenie klienta.