

Diplomarbeit

Bidirektionale Videosprechanlage

Erweiterung einer RaspberryPi basierten Videogegensprechanlage

Höhere Technische Bundeslehr- und Versuchsanstalt Anichstraße

Abteilung Elektrotechnik

Ausgeführt im Schuljahr 2019/20 von:

Andreas Grain 5AHET (HV)
Matthias Mair 5AHET

Betreuer:

DI(FH) Mario Prantl

Innsbruck, am 2020-02-24

Abgabevermerk:

Datum:

Betreuer:

Andreas Grain / Matthias Mair

--

Erklärungen

Eidesstattliche Erklärung

Wir erklären an Eides statt, dass wir die vorliegende Diplomarbeit selbständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche erkenntlich gemacht haben.

Ort, Datum

Andreas Grain

Ort, Datum

Matthias Mair

Gender-Erklärung

Aus Gründen der besseren Lesbarkeit wird in dieser Diplomarbeit die Sprachform des generischen Maskulinums angewendet. Es wird an dieser Stelle darauf hingewiesen, dass die ausschließliche Verwendung der männlichen Form geschlechtsunabhängig verstanden werden soll.

Inhaltsverzeichnis

Erklärungen	i
Eidesstattliche Erklärung	i
Gender-Erklärung	i
Inhaltsverzeichnis	ii
Kurzfassung/Abstract	v
Projektergebnis	vii
1 Einleitung	1
1.1 Funktionalität	1
1.2 Aufgabenteilung	2
I Hardware-Erweiterung - Matthias Mair	3
2 Probleme des Ist-Standes	5
3 Platine	7
3.1 Spannungsversorgung	7
3.2 Mikrofon-Verstärkerschaltung	7
3.3 Lautsprecher-Verstärkerschaltung	7
3.4 Mikrocontroller	7
3.4.1 Programmierung	7
3.4.2 Watchdog	7
II Smartphone-App - Andreas Grain	9
4 Auswahl Rahmenwerk	11
4.1 Vergleich der einzelnen Rahmenwerke	11
4.2 Build-Umgebung	11
4.3 Gedanken	13
5 Programm-Dokumentation	15
5.1 Übersicht	15

5.2	Portable Project	15
5.3	PiBell.Android	15
5.4	PiBell.iOS	15
6	RTSP	17
7	LibVLC Sharp	19
8	Push-Benachrichtigung	21
8.1	Push Notification Service	21
8.2	Azure Notification Hubs	21
9	Entwicklung	23
10	Testen	25
III	Server	27
11	Grundlagen	29
11.1	GStreamer	29
12	Live555 Proxy	31
IV	Appendix	33
A	Verwendete Entwicklungswerkzeuge	35
B	Lastenheft	37
C	Zeitplanung	39
D	Kostenübersicht	41
E	Fertigungsdokumentation	43

--

Kurzfassung/Abstract

Die vorliegende Diplomarbeit beschäftigt sich mit der Erweiterung einer RaspberryPi (RPi)-basierten Videosprechanlage um einer Smartphone-Applikation, sowie der grundlegenden Überarbeitung der Stations-Hardware. Zusätzlich soll ein Linux-basierter, aus dem Internet erreichbarer Server zum Verteilen der Video-Streams eingerichtet werden.

Dieses Projekt basiert auf einer früheren Diplomarbeit von _ und _ an der HTL Anichstraße aus dem Jahr _, welche ein voll funktionsfähiges Videosprechanlagensystem hervorbrachte. Jedoch wurde damals die Hardware-Entwicklung sehr vernachlässigt.

Die Hardware-Erweiterung besteht grundsätzlich aus zwei Teilen: zum einen werden die vielen Elektronik-Bausteine in einer zentralen Platine vereint, was eine einfachere und kostengünstigere Fertigung erlaubt. Zusätzlich wird die aktuelle Hardware der Station um einen Watchdog-Timer erweitert, welcher im Fehlerfall die Anlage zurücksetzt.

Softwareseitig wird die Anlage um eine Smartphone-App erweitert, mit welcher der Fernzugriff auf das System ermöglicht werden soll.

This diploma thesis deals with the extension of a RPi-based video intercom system by a smartphone application, as well as the fundamental revision of the station hardware. In addition, a Linux-based server, accessible from the Internet, will be set up to distribute the video streams.

This project is based on an earlier diploma thesis by _ and _ at the HTL Anichstraße from the year _, which produced a fully functional video intercom system. However, at that time the hardware development was rather neglected.

The hardware extension basically consists of two parts: on the one hand, the many electronic components are combined in a central circuit board, which allows easier and more cost-effective production. In addition, the current hardware of the station is extended by a watchdog timer, which resets the system in case of an error.

On the software side, the system will be extended by a smartphone app, which will enable remote access to the system.

Projektergebnis

--

1 Einleitung

Jedes moderne Wohngebäude ist inzwischen mit einer Gegensprechanlage ausgestattet. Eine solche Anlage erlaubt es dem Wohnungsbesitzer mit jemandem vor der Haustür zu reden, bevor dieser hereingelassen wird. Manche Wohnungen besitzen bereits eine Videosprechanlage, die nicht nur den Ton, sondern zusätzlich ein Video von außen dem Bewohner bereitstellt. Dem Gast wird jedoch immer noch nur der Ton von innen übertragen. Diese Applikation ist ortsgebunden, d.h. es müssen sowohl der Gast, als auch der Bewohner vor der entsprechenden Station stehen.

Im Zuge einer früheren Diplomarbeit an der HTL Anichstraße entwickelten _ und _ eine Videogegensprechanlage, welche mehrere Innenstationen und die bidirektionale Video- und Tonübertragung unterstützt. Die Idee ist, dass in einem Wohnungskomplex pro Partei eine Innenstation verbaut wird, sowie eine Außenstation am Hauseingang. Die einzelnen Innenstationen, also Wohnungsparteien, können dann nicht nur mit der Außenstation, sondern auch mit anderen Innenstationen per Video und Ton kommunizieren. Die Stationen wurden mit Hilfe eines RPi realisiert, um die Kosten gering zu halten.

Am Anfang des 5. Schuljahres bat uns unser FI-Professor DI(FH) Mario Prantl an, dieses Projekt durch eine Smartphone-App und eine Hardware-Überarbeitung zu verbessern. Für die Entwicklung der App sollte das Xamarin-Rahmenwerk verwendet werden, damit die Applikation auf sowohl Android-, als auch iOS-Geräten lauffähig ist. Der Benutzer soll über die App das Video der entsprechenden Station abrufen und mit dem Gesprächspartner sprechen können. Die Hardware-Überarbeitung hat mehrere Ziele; zum Einen soll dadurch die Komplexität des internen Aufbaus einer Station verringert werden. Zum Anderen soll mit Hilfe eines Watchdog-Timers das Langzeit-Betriebsverhalten verbessert werden.

1.1 Funktionalität

Ortsunabhängiger Anlagenzugriff Um dem Anwender mehr Komfort bieten zu können, beschränkt sich die Anlage und dessen Funktion nicht mehr auf die fest installierte Station, sondern ist auch über ein mobiles Gerät bedienbar. Dies bietet den Vorteil, dass auch wenn man sich gerade nicht in der Nähe der Innenstation befindet Gäste empfangen werden können.

Paketdienst Die Paketübergabe erfolgt üblicherweise persönlich. Wenn der Empfänger nicht zuhause ist, nimmt der Paketdienst die Ware wieder mit, und deponiert sie bei einer Paketabholstelle und hinterlässt dem Empfänger lediglich eine Benachrichtigung.

Diese Vorgangsweise ist besonders ärgerlich, wenn man den Paketboten um wenige Minuten versäumt hat oder nicht schnell genug zur Innenstelle gelangen konnte. Beispielsweise befindet sich der Bewohner gerade auf dem Dachboden oder im Keller. Mit einer Smartphone-App ist eine sofortige Kontaktaufnahme mit dem Postboten möglich und verhindert eine unnötige zusätzliche Bearbeitung des Paketversandes.

Sicherheit Die neu gewonnene Ortsunabhängigkeit der Anlage bietet eine höhere Einbruchssicherheit, da man jederzeit Überblick über gewünschte bzw. ungewünschte Besucher hat. Über die Smartphone-Applikation ist ein Öffnen des Türschlosses aus mehreren Gründen deaktiviert.

- Wenn der Benutzer nicht zuhause ist, ist eine Türöffnung sowieso irrelevant.
- Wenn der Benutzer zuhause ist, muss er sowieso zur Eingangstür gehen, um den Besuch in Empfang zu nehmen. Die im Eingangsbereich installierte Station dient hier zur Öffnung der Haustür.

1.2 Aufgabenteilung

Andreas Grain ist der Projekt-Hauptverantwortliche und zuständig für die App-Programmierung. Dies beinhaltet die Einarbeitung in und Verwendung des Xamarin-Rahmenwerks zur Erstellung einer Multiplatform-Smartphone-Applikation. Diese soll den Video-Stream der gewählten Station abrufen und anzeigen, sowie den Mikrofon-Ton des Mobilgerätes zur Anlage zurückschicken. Weiters ist er für die Einarbeitung in das GStreamer-Rahmenwerk zur zentralen Verarbeitung der Video- und Audio-Daten der Stationen und Mobilgeräte über einen linuxbasierten Server zuständig.

Matthias Mair

Teil I

Hardware-Erweiterung - Matthias Mair

2 Probleme des Ist-Standes

--

3 Platine

3.1 Spannungsversorgung

3.2 Mikrofon-Verstärkerschaltung

3.3 Lautsprecher-Verstärkerschaltung

3.4 Mikrocontroller

3.4.1 Programmierung

3.4.2 Watchdog

Teil II

Smartphone-App - Andreas Grain

4 Auswahl Rahmenwerk

4.1 Vergleich der einzelnen Rahmenwerke

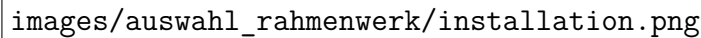
4.2 Build-Umgebung

Aufgrund obiger Aufstellung wurde Visual Studio 2019 der Firma Microsoft als Entwicklungsumgebung gewählt. Diese steht für Schüler und Private gratis auf der Seite <https://visualstudio.microsoft.com/> zum Download zur Verfügung. Visual Studio ist das offizielle Werkzeug für die Programmierung mit dem Xamarin-Rahmenwerk.

Die minimalen Hardware-, bzw. Software-Anforderungen können der Microsoft-Dokumentation des Visual Studio entnommen werden:

Unterstützte Betriebssysteme	<ul style="list-style-type: none"> - Windows 10, Version 1703 oder höher (LTSC und S werden nicht unterstützt) - Windows Server 2019 - Windows Server 2016 - Windows 8.1 (mit Update 2919355) - Windows Server 2012 R2 (mit Update 2919355) - Windows 7 SP1 (mit neuesten Windows-Updates)
Hardware	<ul style="list-style-type: none"> - 1,8-GHz-Prozessor oder schneller; Quad-Core oder besser empfohlen - 2 GB RAM; 8 GB RAM empfohlen - Speicherplatz auf der Festplatte: Mindestens 800 MB, je nach installierten Features bis zu 210 GB des verfügbaren Speicherplatzes. Eine typische Installation erfordert 20–50 GB freien Speicherplatz. - Festplattengeschwindigkeit: Zur Verbesserung der Leistung installieren Sie Windows und Visual Studio auf einem Festkörperlaufwerk (SSD). - Grafikkarte, die eine Auflösung von mindestens 720p (1280 x 720) unterstützt. Visual Studio funktioniert am besten mit einer Auflösung von WXGA (1366 x 768) oder höher.

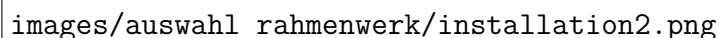
Abbildung 4.1: Soft- und Hardware-Anforderungen Visual Studio



images/auswahl_rahmenwerk/installation.png

Abbildung 4.2: Auswahl des Xamarin-Rahmenwerks bei der Installation

Keine weiteren Einstellungen oder Auswahlen sind zwingend erforderlich, allerdings empfiehlt es sich noch zusätzlich das ".NET desktop developmentPaket installiert. Mit diesem kann man neue Technologien des .NET Frameworks in einer einfacheren Konsolenanwendung ausprobieren und kennenlernen.

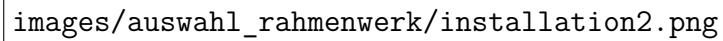


images/auswahl_rahmenwerk/installation2.png

Abbildung 4.3: Zusatzpaket für Desktop-Entwicklung mit .NET

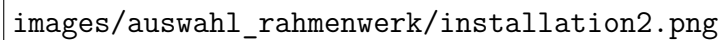
Nach Fertigstellung der Installation von Visual Studio 2019 erscheint dieses am Desktop und im Startmeü als neues Icon. Zusätzlich zum Xamarin-Rahemnwerk sind die notwendigen Android-SDK-Komponenten auszuwählen und zu installieren. Diese sind notwendig ...

Nach dem Start von Visual Studio muss eine neue SSolutionängelegt werden. Diese beinhaltet alle Programmteile, welche für die Xamarin-Entwicklung benötigt werden.



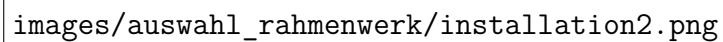
images/auswahl_rahmenwerk/installation2.png

Abbildung 4.4: Anlegen einer neuen Solution



images/auswahl_rahmenwerk/installation2.png

Abbildung 4.5: Auswahl der benötigten Android Plattformen



images/auswahl_rahmenwerk/installation2.png

Abbildung 4.6: Auswahl der benötigten Android Tools

4.3 Gedanken

--

5 Programm-Dokumentation

5.1 Übersicht

Grundsätzlich besteht das Software-Projekt aus drei Teilen:

PiBell ist das portable Projekt, auch .NET-Standard-Projekt genannt. Dieses beinhaltet den ganzen plattformunabhängigen Code, sowie die Definition der UI-Oberfläche mittels XAML-Dateien.

PiBell.Android beinhaltet allen Code, der plattformspezifisch für Android geschrieben wurde. Unter anderem ist hier enthalten die Android-Implementation des Mikrofon-Aufnahme-Services, sowie das Berechtigungs-Management.

PiBell.iOS beinhaltet wie PiBell.Android den plattformspezifischen Code für die iOS-Plattform. Aufgrund mangelnder Entwicklungswerkzeuge wurde dieser Teil nicht großartig behandelt.

Der immer wieder auftretende Name PiBell ist dabei eine freigeistliche Erfindung. Der Begriff setzt sich zusammen aus RaspberryPi und dem englischen Wort für Glocke oder Klingel (Bell).

5.2 Portable Project

5.3 PiBell.Android

5.4 PiBell.iOS

Abbildung 5.1: Aufbau der Solution

--

6 RTSP

--

7 LibVLC Sharp

8 Push-Benachrichtigung

8.1 Push Notification Service

8.2 Azure Notification Hubs

--

9 Entwicklung

--

10 Testen

--

Teil III

Server

--

11 Grundlagen

11.1 GStreamer

--

12 Live555 Proxy

--

Teil IV

Appendix

--

A Verwendete Entwicklungswerkzeuge

--

B Lastenheft

--

C Zeitplanung

--

D Kostenübersicht

--

E Fertigungsdokumentation

Abkürzungen

RPI RaspberryPi