Team: DEC1716

Client: UAVX LLC

Adviser: Swamy Ponpandi

Team Members:

Marco Restaino (Webmaster)

Jackie Larin (Communications Leader)

William Tangney (Team Leader)

Team Email: dec1716@iastate.edu

Team Website: http://dec1716.sd.ece.iastate.edu/

Interactive TV Dashboard

Project Plan

Contents

1 Introduction

1.1 Project statement

This project is to create a dynamic and interactive TV dashboard on an HDMI dongle running an Android operating system. We will incorporate the CEC functionality of HDMI to turn your tv on and alert you for the purposes of an alarm clock. It will integrate a large list of API's to present items to the dashboard such as calendar events, weather, commute time, etc.

1.2 purpose

The purpose of this project is to create a commercial quality interactive assistant that runs on your TV. The assistant takes a number of commands from the users and executes them. We want to create a tool that can be useful and easy for everyone to use. Our client has the ultimate goal of selling this application to a commercial electronics provider to be included in some of their products. As a team our ultimate goal is to create an innovative application that someone would want to use everyday.

1.3 Goals

For this project we as a group have two main goals, to create a quality and useful application and to gain knowledge for the future. When making this interactive TV dashboard we will certainly be aiming towards making something really cool and useful but another big reward of the project will be the knowledge gained. We are excited to expand our knowledge of programming for android and learn about the plethora of technologies we will be incorporating into our project such as: voice control, hand tracking, 3rd party APIs, etc.

2 Deliverables

- Fast loading intuitive application
- Integration of hand tracking for control of the dashboard
- Integration of voice commands for control of the dashboard
- Application running on an HDMI dongle using the Android operating system
- Integration of 3rd party API's to deliver information from people's services like Google Maps
- Integration of HDMI CEC to have a "Wake Up" functionality for alarms
- Make secure system that is impossible to exploit

3 Design

3.1 Previous work/literature

1)

This article lists multiple algorithms and some source code using openCV a popular computer vision library. This should be useful as it provides a strong starting point for hand tracking and other blob tracking in an image received from the camera.

2)

This article provides a number of different configuration details and API for the HDMI-CEC interface. It provides a couple of examples on how to use the code as well as different capabilities of this interface. It also provides sample configuration XML files so that we can configure this feature to work on our device properly.

3 and 4

This link provides the API details of the gmail API. It shows how to access the emails on google's mail servers. It shows how to parse the data received from the queries to google's database. The second link shows a basic email retrieval sample application. It should not be too hard to create a bounded or started service based on this code.

5

This is an example facade design to the YAHOO weather services. This shows how to design a simple android wrapper for an API and shows how to use it safely in an android application.

6 and 7

These links show how to set up google maps API. The first API shows a basic map. The second link shows how to get directions between two points and the current travel time via the traffic_model() function.

3.2 Proposed System Block diagram

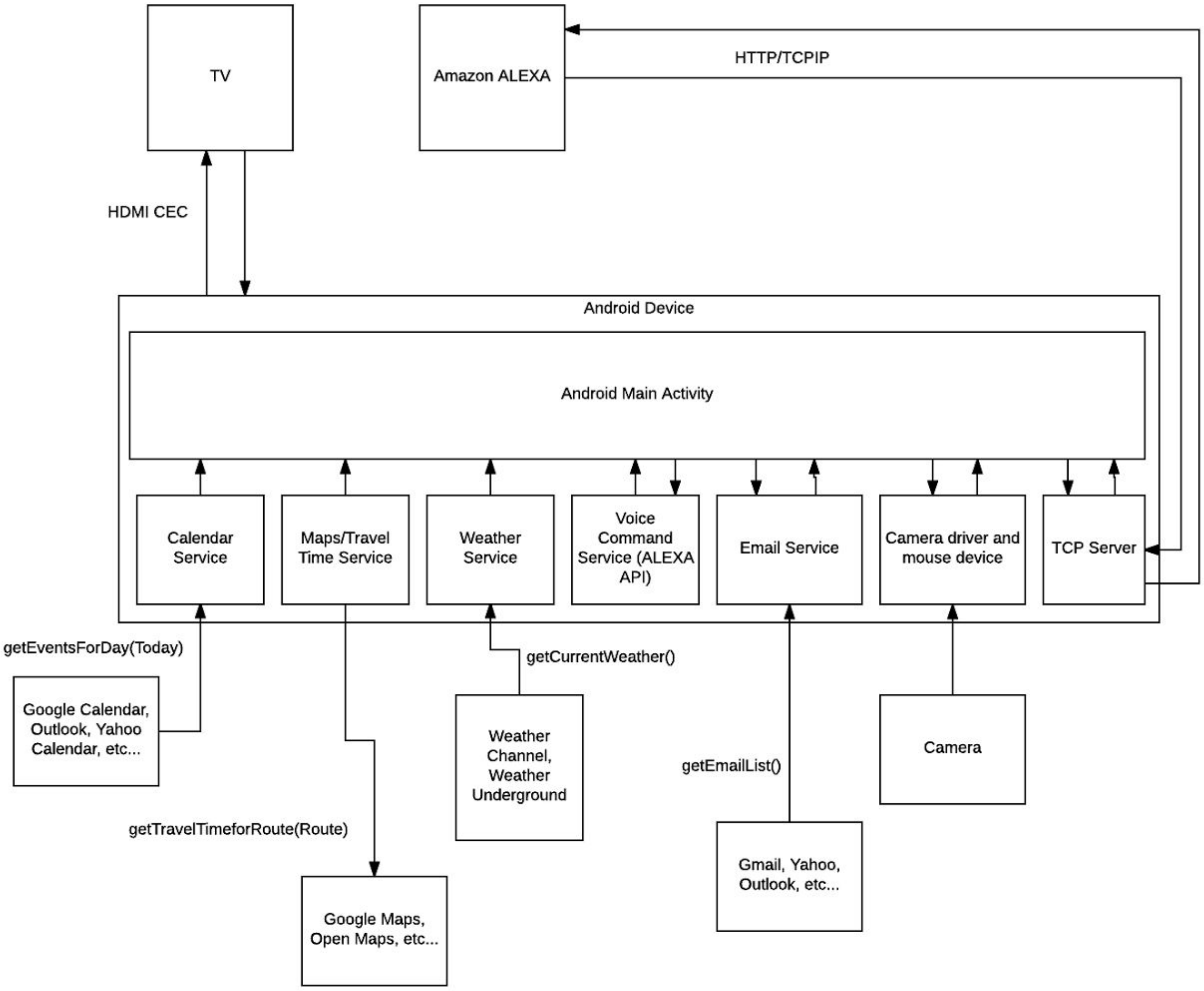
TV
Amazon ALEXA
HTTP/TCPIP
HDMI CEC
Android Device
Android Main Activity
Calendar Service
Maps/Travel Time Service
Weather Service
Voice Command Service (ALEXA API)
Email Service
Camera driver and mouse device
TCP Server
getEventsForDay(Today)
Google Calendar, Outlook, Yahoo Calendar, etc...
getCurrentWeather()
Weather Channel, Weather Underground
getEmailList()
Camera
getTravelTimeforRoute(Route)
Google Maps, Open Maps, etc...
Gmail, Yahoo, Outlook, etc...

Figure 1

3.3 Assessment of Proposed methods From Figure 1

getEventsForDay(date) this function will be a facade design to a number of calendar APIs and it will return a list of all of the events for the day for the end user.

getTravelTimeForRoute(Start, Finish) another facade design that will allow us to access a number of APIs from one service and calculate the travel time and fastest route for the chosen start and finish points.

getCurrentWeather() this method will take the current location and get the weather for that location

getEmailList() this will be a facade design for many APIs that will collect a list of emails for the end user's account.

HDMI/CEC this is a protocol for sending events from a device to a TV device. These will be things like power on/off and volume up/down.

3.4 Validation

getEventsForDay - Sample events will be prepared in the calendar service and the list of events will be compared against the list from the calendar API.

getTravelTimeforRoute(start, finish) - The calculated route will have to be checked against that of the API that is used to call or other APIs to validate its accuracy within a certain tolerance. The shortest route heuristics can sometimes produce different answers so it's best to compare within a tolerance.

getCurrentWeather() - Check the current weather with calls from other weather reports. Compare within a tolerance. It can also be checked against a digital thermometer for temperature.

getEmailList() - Check if the retrieved list of emails matches that of the user's email client.

4 Project Requirements/Specifications

4.1 FUNCTIONAL

Functional Requirements

- Integrate multiple API components into single application.
- Hand tracking control of the dashboard.
- Integrate voice commands to dashboard interface.
- Integrate 3rd party APIs to deliver information to the user.

4.2 Non-functional

- Clean and simple dashboard user interface.
- Secure application, impossible to exploit.
- Applications scales multiple types of Android devices and screen sizes.
- Avoid legal and licensing issues.

4.3 Standards

In our code we will be integrating multiple APIs. These APIs are all licensed such that they can be integrated into an application that is released under either a proprietary or open source license. Piracy of code is strictly prohibited in applications developed under proprietary licenses and is a violation of all legal standards as well as standards set forth by IEEE and ABET.

5 Challenges

Include any concerns or details that may slow or hinder your plan as it is now. These may include anything to do with costs, materials, equipment, knowledge of area, accuracy issues, etc.

We have identified a few areas which may cause challenges.

- Presently one member has full knowledge of Android development while the others have some or little knowledge
- Combining all the different APIs may cause an issue
- Limited knowledge of the HDMI dongle will cause issues in the beginning but with research and experimenting we will overcome this challenge

6 Timeline

We will be creating an Agile software development environment where we will meet weekly with our client to go over our updates as well as display our demos. Our goal is to make sure our client is involved in the design of the project.

6.1 Spring Semester

The part of our Spring work time will be spent on research. This will allow the developers to gain a complete understanding of Android, CEC functionality of HDMI, and other properties needed to construct our project.

Feb - March

- Finish first draft of the Project Plan and turn in on Feb 24th
- Finish our research
- Finish sketch of how the dashboard will look and get approved by client

March

- Finish second draft of the Project Plan and turn in on March 28th
- Finish first draft of the Design Document and turn in on March 7th
- Finalize solutions
- Develop GUI
- Prepare for Group Presentations

April

- Prepare for Group Presentations
- Finish final draft of the Project Plan and turn in on April 18th
- Finish final draft of the Design Document and turn in on April 18th
- Add the major features to our application

6.2 Fall Semester

Fall semester will be making changes to the existing application and providing the client with a final application.

August - September

- Have an application that interfaces with the tv
- Work on adding voice control and motion control
- Fix any issues the application might be having
- Work on user interface

October - November

- Continue to work and improve user interface
- Get different user to test the application
- Fix any issues the application might be having

December

- Display the project to the client
- Fix any issues the application might be having
- Turn over our project to the client

7 Conclusions

Our goal is to create an interactive assistant that runs on Android operating system. The assistant will wake up the user and will provide the user with different features. The user will have the ability to ask the assistant for the weather, travel time, and if he/she has anything in his/her calendar. Our plan for our team is to implement the basic features and if time is available implement other features. We want to make sure the basic features are working properly and that the interface is clean and smooth. Once the interface and features are working correctly we will add a feature that provides the user the ability to build their own dashboard based on the features they need. Our primary goal is facilitate sharing personal information to the user. The application will provide the user with everyday information making life a little easier for the user. Using the Agile methodology we have broken up each month into tasks. Our client will know what is going on in the project. He will have the chance to provide his feedback.

8 References

List all the sources you used in understanding your project statement, defining your goals and your system design. This report will help you collect all the useful sources together so you can go back and use them when you need them.

1) https://www.intorobotics.com/9-opencv-tutorials-hand-gesture-detection-recognition/

2) https://source.android.com/devices/tv/HDMI-CEC.html

3) https://developers.google.com/gmail/api/guides/

4) https://developers.google.com/gmail/api/quickstart/android

5) https://github.com/zh-wang/YWeatherGetter4a

6) https://developers.google.com/maps/documentation/android-api/

7) https://developers.google.com/maps/documentation/directions/