Universal Device Communication Standard

Welcome to a new project that has been started, known as the Universal Device Communication Standard. In this document we will give a brief overview of our idea for this standard, what it defines, who will refer to it and why we want to create it.

1. Introduction

Today's technology is rapidly evolving and we have noticed a certain restriction that is slowing down this progression: that is the inability for devices to understand one another. We currently have many protocols in place that allows devices to communicate and it is protocols like this that we want to take advantage of.

The Universal Device Communication Standard (UDCS) is a standard that will define information in a way that electronic devices can understand. It will rely on previous protocols and technologies to work and in this manner it can be implemented easily. Over time, as more companies and products begin to adapt the standard, technology will become more advanced and we believe that engineers and product designers will no longer be restricted in what they can create.

Like all ideas, we have to start from nothing and this is where we are currently: we need help from software developers, engineers and innovators alike to create, implement and manage this standard that we hope can encourage technology to become more advanced.

The UDCS is quite a complicated concept to explain and in this document we hope to outline several case studies and scenarios that will give a clear understanding of what we hope to achieve.

2. Communication

Today's technology already has several incredibly powerful communication protocols and it would be idiotic to attempt to merge all of these protocols to create yet another. Instead, what we want to do is to utilize these communication protocols as a method to transmit the information that adheres to the UDCS.

2.1 Example 1

For our first example, let's imagine you have just entered a train station. In the future we would be able to pull a device (a phone, a tablet, a laptop) out of our pocket and instantly see the current trains at the station, when they are going to leave and what trains are about to arrive. For now we will keep it simple and avoid adding features that one day we would hope to achieve (eg: a map of the train station and where the user is currently standing etc).

Today this would be perfectly achievable. However, an application would have to be developed for each device/platform, this app would have to be managed and there would have to be a way for the train station to keep this information updated.

Let's now make this situation a bit more complicated. Let's say you have a specific train to catch and you booked tickets for this online. You have an email somewhere in your device with all the information you need. It would be ideal if this information would be stored in some form for easy access. No doubt there is currently websites/apps that contain ticket information, booking etc but in the future we expect these two concepts to link together. The user wants to know when their specific train is arriving when they enter the station. The more complicated the users' requests, the less like that there is an application out there to achieve these things, and this is where the UDCS steps in.

As with we mentioned with current communication protocols, it would be idiotic to attempt to build a set of software and applications that handles everything. This concept will always fail due to competition and people's preferences.

We now have our scenario but let's re-imagine it in a world where the UDCS had been implemented. The user enters the train station and user's device (phone, tablet etc) links to the train station network. Currently we don't know how to achieve this step but this is an example of the problem that we wish to solve. For argument's sake we are going assume that the device has automatically connected to the station's wifi network. Once the device has connected, the train station network will let the device know what information is available. This information will be sent in a format that the device will understand (UDCS has this sorted). On the user's device is an app called 'train station' this app was designed to receive incoming information regarding train stations. It's an arbitrary app and we expect apps like this to be developed that can deal with a single thing or many things – a personal organiser app for example will have a lot more functions than 'train station'.

The first thing to note here is that the user does not have to a specific device; it could be any device that has wifi built in. The user doesn't even have to have a specific app since the information has arrived via wifi and is in a certain format. The format (UDCS) is universal and so absolutely anyone can create an app to understand it.

'Train station' has been programmed to wait for incoming wifi connections that have adopted the UDCS and carry information about train stations. Now that it has received this information it could notify the user in whatever way the manufacturer has designed the device and the user can now receive any information that the train station wants to give.

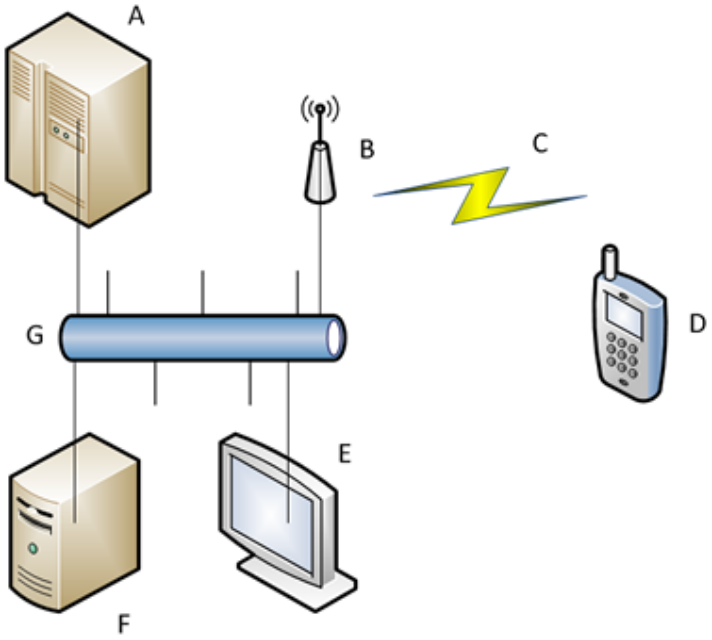
On the user's device is another application that deals with train bookings. The information from this app is stored on the user's device in a local area that all apps can access. It has been formatted using the UDCS and contains all the user's train bookings. 'train station' can access this file and read it since it knows what it's looking for, it can then parse this information however it decides. Since it's our arbitrary app we are going to make it tell the user which platform to look for and when their train is going to leave.

Hopefully this example has given a clear outline of what the UDCS hopes to achieve. There are several factors that we must take into account in order for all of this to work:

- The device needs to have an area that all applications can access, allowing applications to communicate with one another
- The device has to connect to networks automatically
- The device has to be able to accept information from these networks

It is issues like this that need to be addressed to allow the whole system to work.

We are now going to break this whole scenario down into a diagram that will explain the concept of the UDCS.



A – Train Station Servers: this is where all the train information is held. Without a doubt there are servers like this already

B – This is our proposed addition to the train station network, in this example this communication is done by wifi

C – The wireless communication signal

D – This is the user's device that is receiving the information

E – This represents the displays around the station that show the train times. These will already have a way of communicating with the server and we don't intend UDCS (at this stage) to alter this sort of communication. This is local communication that has already been defined and to change it would create hassle that would not benefit anyone.

F – This is a computer on the network that would alter train times etc and control the station and server. It would be proposed that this computer have a program installed on it that formatted data in to UDCS and sent it to **B**.

G – The train station network

This arrangement would allow a UDCS infrastructure to be implemented without drastically altering the current network. The software that takes the data on the network, formats it to the UDCS and transmits it could be designed by anyone. One day perhaps all of this functionality will be included in the entire network and not just be an add-on. The concept of having a separate bit of software to do the formatting also allows only certain information to be sent.

2.2 Example 2

Let's consider another example. We'll make this example shorter since the concept of UDCS has already been outlined.

In this scenario a user walks into their doctors surgery; on the surgeries network is a list of appointment and patients. Each patient has a unique ID that allows the UDCS to identify them. This ID is unique to both the user and the system, ie a user will have a different user ID for each system that they are attached

to. Each system will have its own unique location ID to allow the user to transmit the correct information later on.

Before we continue let's skip back a week when the user made their appointment. Let's say they called up the surgery and made the appointment. We are going to assume that phones have been developed to transmit UDCS data. Once the appointment has been booked the systems returns all the information about the appointment including the user's new unique ID along with the system's unique ID. It would be at this point that data would also be sent formatted as calendar data and other systems would be able to use this data.

Back to the surgery. The user walks in and the surgery identifies itself to the user's device (this could be via wifi, Bluetooth, RF etc) and the device responds with the user's ID for that system. The surgery's system would know that the user has arrived and could check them in and then send a notification to the user to let them know. The device could then deal with this however it wants.

We have made some large assumptions here that we will now address:

- The surgery phone would have to be linked to the local system
- The phone that received the initial booking data has synched with all the other devices that the user owns
- Once again, the user's device has connected to the system automatically
- That there are wireless systems in place that would allow this sort of communication to happen

Other than these issues, this scenario outlines both the different types of communication, the different formats available and the security of the system.

2.3 Types of Data

There are two types of data that the UDCS will deal with:

- Static Data: this data is one-way. In example 1, the train station is transmitting static data, devices will receive this data if they want and parse it how they want.

- Dynamic Data: this is data that will require a conversation between two parties. In example 2, the doctors' surgery identifies its own unique location ID to the device, the device will then return its ID assigned to that location.

2.4 Types of Communication

There are 3 types of communication that the UDCS would take advantage of:

- Broadcast: Information formatted to the UDCS would be broadcast over a network; in example 1 this would be train station broadcasting the train information. Broadcasted data is not secured as it is designed to be received and read by anyone in the local area.

- Point-to-Point: this would be one device communicated directly with another device. This communication would require encryption to ensure that other parties were not listening in.

- Point-to-Multipoint: this is information that is being communicated with multiple devices from a single point. In example 2 it would be the surgery transmitted its location ID until another device responded. The two would then have a conversation. The surgery network however could be having multiple conversations at once with several users in the surgery. This communication would also require encryption to ensure that other parties were not listening in.

2.5 Secure Communication

The main issue that users will be concerned about is security. This will be address in three ways.

Firstly, although the UDCS does not dictate how a system should behave, merely how the information in a system should be sent, designers are encourage not to transmit any personal data but instead transmit unique IDs (as in example 2). The user did not transmit their name, age etc, instead they transmitted an ID that the surgery appointed them a week before. This ID is only unique to the surgery, so the user would be identified with another ID in another location.

Secondly, current communication protocols already have security built into them and it is the designers' job to ensure that they are using correct security measures on whichever communication protocol they are using.

Finally, if a user were to attempt to communicate with a network acting as someone else, then this would have to be addressed. We don't know the answer to this solution but that is up for discussion with other contributors. Presumably the same security measures would be used as those in Bluetooth chip and pin devices. Factors like this aren't for the UDCS to decide.

3. Format

Before we continue, we must stress that the Universal Device Communication Standard does not decide how information should be transmitted between devices, only how that data should be formatted. In both examples 1 and 2 from section 2, we have assumed that the network has been setup in this way, simply because these are example of what people want the 'future' to be like.

3.1 XML-like

Data that has been formatted to the UDCS will be designed to look like XML. The number of top level categories will increase as the standard begins to become more universal and from there each category will be broken down into subcategories until the data can be transmitted in the format defined for that subcategory.

3.2 Categories

In example 1 when the train station want to transmit the timetable it will start at a top level category such as *timetables*, it will then be refined further to say, *train timetable*, and finally we'd have each train.

We have now (very vaguely) defined a method of transmitting train timetables. Of course the timetable category will have lots of different sub categories: *school timetable*, *bus timetable*, etc. The main thing to think about here is that a smart phone app designed to display timetables will be designed around this format. Similarly if there is a calendar application that a user has a meeting booked in to, the calendar app could retrieve timetable information from bus/train companies' websites and then inform the user when they need to leave the office/house.

It is concepts such as this that allow applications and device to utilise the UDCS to give users the best information possible by allowing software and devices to communicate in ways they understand.