BACHELOR APPLIED COMPUTER SCIENCE

Final Work

Combining the new Android platforms:

Wear, TV & Auto

Yoshi Degent

Promoter: Dhr. R. Dejonckheere
Co-promoter: Dhr. B. Van Rillaer
Academic year 2014-2015

Final Work submitted to obtain the degree of Bachelor in Applied Computer Science

Table of Contents

Preface

As I got my first info-session about the Final Work in spring 2014, I was immediately caught up about finding a good subject. Ideas went by and I even debated and brainstormed with my fellow students about the different ideas we had. In the process I was absolutely not short of any ideas, there were plenty. Though, they were not quiet spiking my interest that heavily. The thing about myself is, I really need to fully believe in an idea to fully give myself and be motivated to successfully accomplish the goals. Besides, I also thought about the future, which in my case means I'd preferably choose some subject related to one of the fields I could see myself working in.

So it wasn't until summer 2014, that I had an idea that really took my interest. It all began with watching some videos and reading some articles about the Google I/O conference of that summer. Me, a fanatic user of the Google products and a huge fan of their innovation, really looked forward to what they were going to announce this time.

While many other companies struggle to bring new and innovative concepts that take it to the market, Google presented a lot of new products on its I/O conference. Though I could wright pages and pages about the new innovation I liked from Google and its partner companies, let's cut to the case and get to what I personally, was most interested in for my Final Work.

Announced on the Google IO, they expanded Android with no less than three new technology environments! To be more specific in the near future, the Android so familiar from your phone or tablet, will also be on your TV, in your car, and on your wrist. This basically means the release of three new Android branches named in the same order Android TV, - Auto and -Wear.

This announcement was the fuel to get my idea rolling. As a developer I wanted to help these new and interesting technologies to take off and research what new possibilities these devises add to the current system. Because what is great hardware without developers to create usable and handy software to improve the way we interact with technology today.

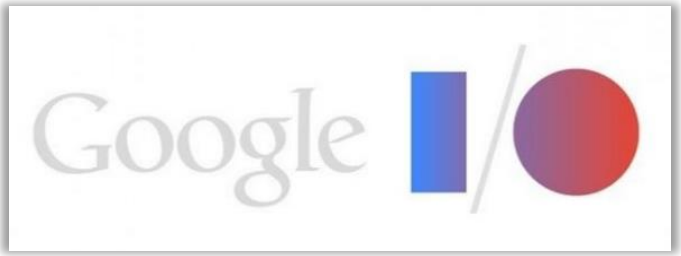
There it was, the basics of the idea, to build an application that supports all the new Android environments. What I did next was contacting my lector Bert Van Rillaer, about my idea. He also was really interested and encouraged me to dive in deeper and continue to work on the idea. Furthermore he answered all the questions I still had. For the further developing of the idea I really have to thank Gunther Sinove with whom I held late-night brainstorming sessions about all our application ideas. The question was: "What can I develop to make optimal use of the new environments and leave a new user-experience?" So after lots of research, I finally found an application idea I wanted to develop.

To enclose this preface I still want to make some notable mentions to a few people who helped me in the process of this project and without whom this would've never been possible. I want to thank Ruben Dejonckheere, who monitored my work during the whole project. I want to thank Vincent De Caster who helped me with feedback and design. And last but not least I want to thank the friends, family and colleagues who where a huge constructive and emotional support.

Introduction

Android was bought by Google in 2005 and has grown to be the most used open source operating system for mobile devices today (2015). One of the reasons Android is so big, is because it runs on thousands of distinct devices in price categories ranging from a few bucks to touching thousand Euro's and in some special cases even more. This can be seen as both a strength and a weakness of the platform, but in my opinion is still a huge advantage over its competition.

Of course this advantage is debatable but I think we can all agree that Google doesn't suffer from a lack of innovation and ideas in the world of technology. They prove this again by the announcements on the Google IO of 2014. This year they came up with no less than 3 Android versions namely Android –Wear, -TV and –Auto. This basically means in the near future we will be able to interact with technology in ways like never before due to these new platforms. These new devices in combination with the cloud will ensure us our data is accessible and sharable anytime on every device. A prediction anyone could make because the trend of people sharing their data is clearly growing and new ways of doing this submerge every few years in this fast growing business. People like Mark Zuckerberg, founder of Facebook, have also be known to acknowledge this trend and basically his business runs on this whole concept.



1 - Google IO 2014 logo
image-source: http://cdn.androidbeat.com/wp-content/uploads/2014/06/google-io.jpg

But to get back on the different announced platforms, it is truly an interesting subject for me to base my final work on. More specific I will research the interaction between these new platforms together with some side-questions. I want to research how data or media can be shared between these different Android platforms and how they interact with each other. This while all gaining a global knowledge of the possibilities of the new technologies and their real life purposes. Further in this section I will go more in detail on what I am trying to accomplish, which questions I am going to answer and how I plan to do this together with a global view of the application idea I had.

RESEARCH GOALS

The basic goals of my Final Work are to make myself familiar with the new Android platforms and try to see how they can be combined in a functional partnership. I want to know what the real life applications are of all of the new devices guided by Google's vision and principles, but with my own views and experiences as well as development during the Final Work taken into account. With all this information combined in the end I want to be able to come up with my own conclusion on the applications of these new technologies and how they can beneficially interact with each other.

Previously mentioned goals I want to achieve by taking different approaches. At first I will look into the different platforms by performing research as well as take information from my own experiences to gain a global knowledge. After I have gained a global understanding of the technologies I will deepen myself into the interactions between them and see how data is shared between the devices.

Secondly I would like to get more familiar with the development side of the platforms by developing an application which supports these new kind of devices. By developing such an application I will learn how the new Android environments work and how to share data between them. This will not necessarily be an application with a real life purpose, but it will at least consist of implementations to test some of the functionalities of the different platforms. So it is not my goal to deliver a full-fledged application for this would increase the scope of the project in a significant way where I would not be able anymore to accomplish them on my own. Rather I will try to develop as much functionality as I can and more important get the right information out of my development.

Both of the mentioned approaches will be analyzed and documented in this Final Work report. With all the gathered information of the different sources I strive to be able to make a conclusion of the previously mentioned subjects.

Hereby I will make myself more familiar with the Android ecosystem and more specifically the new parts introduced on the Google IO to eventually increase my analyzing and development skills as well as my experience in the Android field.

RESEARCH QUESTIONS

To accomplish the goals mentioned in previous section I need to ask myself a few research questions to define the problems I want to solve with this Final Work. Some of them I already teased in with mentioning the goals but here I will make it more clear what questions I am going to answer specifically. The most important of them and biggest questions are here listed below:

- *What are the real-life applications of every of the new platforms announced, namely Android Wear, -TV and –Auto?*
- *How is data shared between the devices or how do they interact with each other? Do they possess a storage capacity of their own or do they depend on another way of data storage?*

- *Are the development platforms possible to be used independent of the standard smartphone version or are they only an addition in functionality on top of the normal experience?*

- *How does development look for each of these devices more specifically on the subject of data sharing? What possibilities does Android offer?*

These are the main questions I would like to answer with my Final Work and by answering them I will try to improve my Android development skills as well as to broaden my knowledge with the new information that will be gathered.

I chose these questions because they seem to deliver the goals I set for the Final Work and also regarding my own interests. I also wanted to make sure I chose questions which would enable me to research the subject with possibilities of thinking 'out of the box' and coming up with my own ideas in the process. It is important for me to gather analysis and form a personal conclusion out of it.

For me new technologies have always been very interesting because they can possibly deliver such new experiences and the way we use technology in various ways. Therefore I surely wanted to go and find out what these possibilities are on the moment of working on this Final Work. As a tech geek the motivation was therefore very high.

I just discussed why I chose my research questions but in the following section I will explain what my plan was to give an answer on the questions. I will explain how I tried to get from questions to answers in an incrementally improving way.

WORKING METHOD

Like already mentioned in the research goals, I planned to work via 2 approaches. One being the analysis of various sources and experiencing the platforms as much as I can both as a personal user and a developer, the other being trying to develop my own application with data sharing and technology interacting functionality.

Analysis

For analysing I basically wanted to look upon the platforms from two different views.

At first I would try to get a global understanding of which features the platforms possess, their real life usages, the vision and design principals of the creators as well as the current state and evolution of the technologies.

Secondly I would like to take a look on a more detailed perspective, specifically on the subject I am going to research which is data sharing and technology interaction between these new devices.

Both of these views I will approach from the standpoint of a developer as well as a user. By doing this I can get a better understanding of the user-experience and the development for the different platforms.

In the process of analysing in all those perspectives I was able to come up with some sub questions besides the main research questions I already talked about. They help me to perform my research goal-orientated, rather than randomly searching, without knowing what I'm even looking for. Following are the most important ones I came up with:

- *Is there a beneficial user experience to the new devices in comparison with older alternatives?*
- *Do they leave enough value for customers to buy into them and for whom are they, in other words what is their target audience?*
- *How seamless is the data sharing experience and why would you want to share your data between those different devices?*

These are just a few interesting sub questions which will probably also be answered in the analysis of the different platforms, besides the main research questions I already mentioned.

With all of these questions in the back of my head I began the voyage of analysing the technologies. I started by reading up on different documentation, the official ones of Google as well as third party information. The documentation of Google would provide me with the vision and design principals they have for their products, where other articles would function more as a review or user experience to the platforms. I also watched plenty of YouTube reviews from respectable reviewers. Besides I own an Android Wear smartwatch myself which was ideal for testing out this platform and thus developing my own opinions on the user experience.

So basically I used various sources to come up with an analysis and furthermore develop the answers to the questions I ask. Hereby I could develop my own conclusions to the problems and form my own opinions and ideas on the matter. In the whole process I will always try to make a distinction on the objective view and my personal opinions in this report, but mentioning them both for sure.

Development

On the development part I planned to develop an application in which I would try to make use of data sharing between these different devices. I would need my analysis and research and apply the gained knowledge in my development. My plan was to at least develop some functionality to test the data sharing capabilities of as much platforms as I can. But in case there would be time left I would like to add some real life usages to the application. Also on the development part I asked myself some sub-questions beside the main ones, to work goal-orientated. Following are the most important ones:

- *How easy is it to upgrade your current smartphone application with a version for one of the other platforms?*
- What are the requirements for using each of these new platforms?
- How is data sharing managed between devices?

By solving these questions during the development, it will not only be a chance to research development for the platform but also a chance to apply the knowledge of the previous analysis I did.

THE APPLICATION IDEA

To research the interaction between these platforms I planned to combine them into one application. Therefore I needed an application idea that would leave a benefit for all of these devices. The implementation of these new platforms should leave a positive addition to the functionality and usability of the application.

This was a pretty hard task because it is not easy to find a good application to develop which could benefit from the support of all of the new platforms. But after a long brainstorming session I finally got an idea. This idea would basically function around the principal of sharing your media and the different ways of displaying this media to the user. That was a great idea in a way that these devices primarily leave new ways of interaction with our media. For example a TV leaves the benefit of a big screen to scale up the media we want to display, where a wearable gives us a more glanceable, quick experience. The media we share with each other could definitely benefit from these new ways of presentation to their users.

There is where my application idea would provide an improvement on the current way we share and more importantly display our media. The initial idea was to even include cloud functionality into this application so that the media would be easily shared across all your devices and easily shared with other users. This idea can be explained a little bit better with a possible use case:

Peter wants to share a video with his best friend John. For this he uses the new application in the Google Play Store. When John receives this video, he gets notified on his Android Wear smartwatch without using any other device yet. On his smartwatch he can choose to do some quick basic actions like ignoring the shared video, save it for a later time or choose to display the video on his phone, when he is traveling, or on his TV when he is at home. In this use case Android Auto wouldn't really be beneficial, but imagine the same scenario for sharing a location and the whole functionality changes.

This is basically how the application would function and it would enable the user to share audio, pictures, video, locations and maybe even more in the future. Sadly I had to see that the scope of this project was way too big for a one man project in the specified timespan and thus I would have to replace my focus.

The new more simplified idea would keep the same goals of researching the new Android technology environments but now using a more simplified version of the previously mentioned application idea.

I will still implement as much sharing and displaying functionality as possible but this will be more in the realm of a test application where I test the different sharing possibilities between platforms. I will one by one look at each of the platforms try to implement some of the sharing functionality to ultimately being able to answer the research questions.

To give an order to the application development I will first look at Android Wear, followed by Android TV and last but not least look into the development for Auto. This all at the same time of developing a smartphone application to accompany the other platforms.

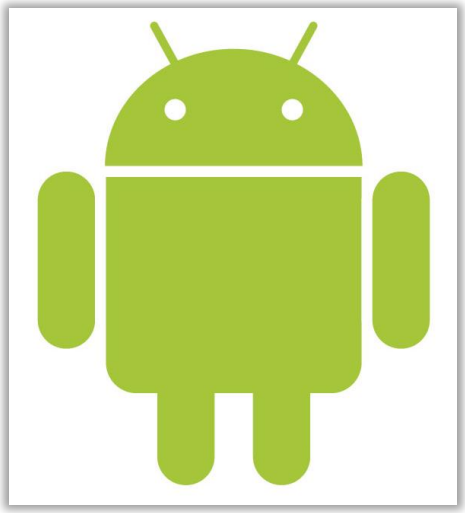
Platform analysis

To begin the research of the platforms I am going to work with, I first needed a good background knowledge of them, knowing the features, use cases, strengths and weaknesses as well as Google's vision mixed with my own opinions on its future usages. This is basically what I will discuss in following section, I will give a more detailed explanation of the platforms I researched and thus provide a summarized explanation of the android ecosystem. Furthermore I will talk about the main version I focussed myself on named Android Lollipop together with its new platforms mainly Android Wear, -TV and –Auto.

Also, I will try to give an answer to some of the previously mentioned research or sub questions by noting down my analysis. I will try to mention the most important of my findings, like the problems the platform maybe wrestling with as well as some of the future possibilities. By thinking 'out of the box' I want to mention my own opinions on the platform also and the conclusions I made.

THE ANDROID ECOSYSTEM

Because Android is open source it basically means every hardware company can edit the operating system and use it for their own devices, but Google still tries to get some influence in the design principles and guidelines they should follow. I personally love a more stock experience of Android or with other words I like the vanilla unedited version. Other companies like Samsung, HTC, Sony and others often add their personal layer, which in my and many others opinions only adds a layer of unnecessary complexity. Even worse, these devices often have to wait really long for updates, because when an update is released by Google, the companies still have to develop their own layer on top of it.



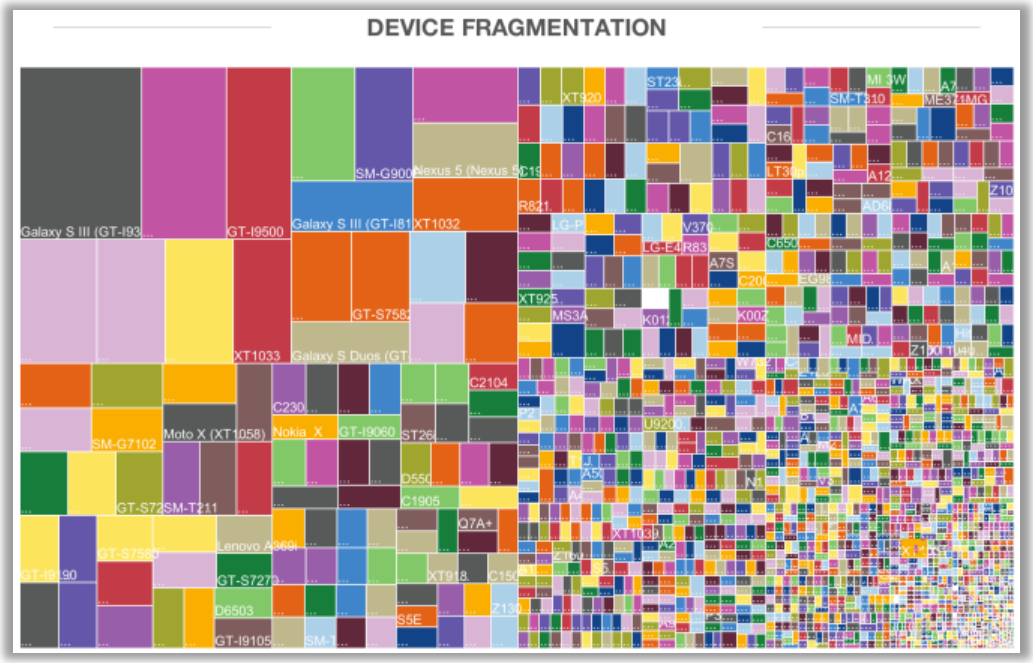
2 - Android logo,
Image-source:
http://www.android.com/media/android_vector.jpg

This leads to a large amount of device fragmentation, which doesn't make the life of an Android developer easier. You often have to provide support for hundreds if not thousands of devices, while on the Apple platform this is just a handful. A hefty counter argument for becoming an Android developer, but then again who doesn't like some challenge? Even more because it is open source you often have much less restrictions and both as a developer and a user you can benefit from a lot more features because of this open-source character.

Device fragmentation

Today (2015) Android counts more than 1 billion active users and over 19,000 distinct devices. So the challenge I talked about certainly isn't small because a lot of devices basically means a lot of different device configurations, screen sizes and Android versions.

Therefore it is very important for Android developers that they always keep their target audience in the back of their mind while developing applications. It is nearly impossible for a developer to support every single device out there and therefore choices have to be made. To illustrate how big this device fragmentation really is you can take a look at below figure which shows the market share of every device according to an article written by Phil Nickinson titled *Another Android 'fragmentation' report misses the point*:



3 - Picture dates from August 2014.
Source: http://www.androidcentral.com/another-android-fragmentation-report-misses-point

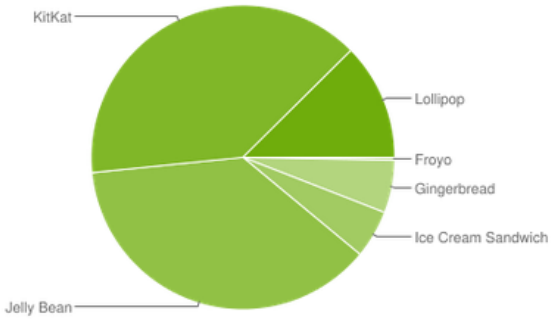
This picture only empowers the importance of specifying the target audience really well. That is also what I am going to do further in this report for the other Android platforms. Of course these new devices will not be for everyone, at least not yet. It is therefore important to take our conclusions with the target audience in mind, rather than giving them a valuation for everyone.

Besides specific devices it is also important to take a look at the different device configurations. This basically means you should test your application for devices with and without resources like Bluetooth, NFC, accelerometer, mobile data, etc. Of course this decided with the target audience in the back of your head. Let's say your application goal solely is to transfer data with Bluetooth, it would not be very efficient to test your application for devices which don't have a Bluetooth radio present. Therefore it is recommended to always decide what device configurations you want to support first.

Android versions

In Android because of the fragmentation not every device always gets the latest update, nor does every Android user upgrade their phone after a year or two when it is out-dated like some other people do. Therefore it is important to look at what market segment, each of the Android versions, possesses to make your decision on what you want to support. Following picture provides data about the relative number of devices running a given version of the Android platform fetched from the Android developer documentation website:

Version	Codename	API	Distribution
2.2	Froyo	8	0.3%
2.3.3 - 2.3.7	Gingerbread	10	5.6%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	5.1%
4.1.x	Jelly Bean	16	14.7%
4.2.x		17	17.5%
4.3		18	5.2%
4.4	KitKat	19	39.2%
5.0	Lollipop	21	11.6%
5.1		22	0.8%



4 - *Data collected during a 7-day period ending on June 1, 2015. Any versions with less than 0.1% distribution are not shown. Source: https://developer.android.com/about/dashboards/index.html*

In my research I will mainly focus myself on the latest officially released version at this time, Android Lollipop. This because of various reasons later discussed in the report.

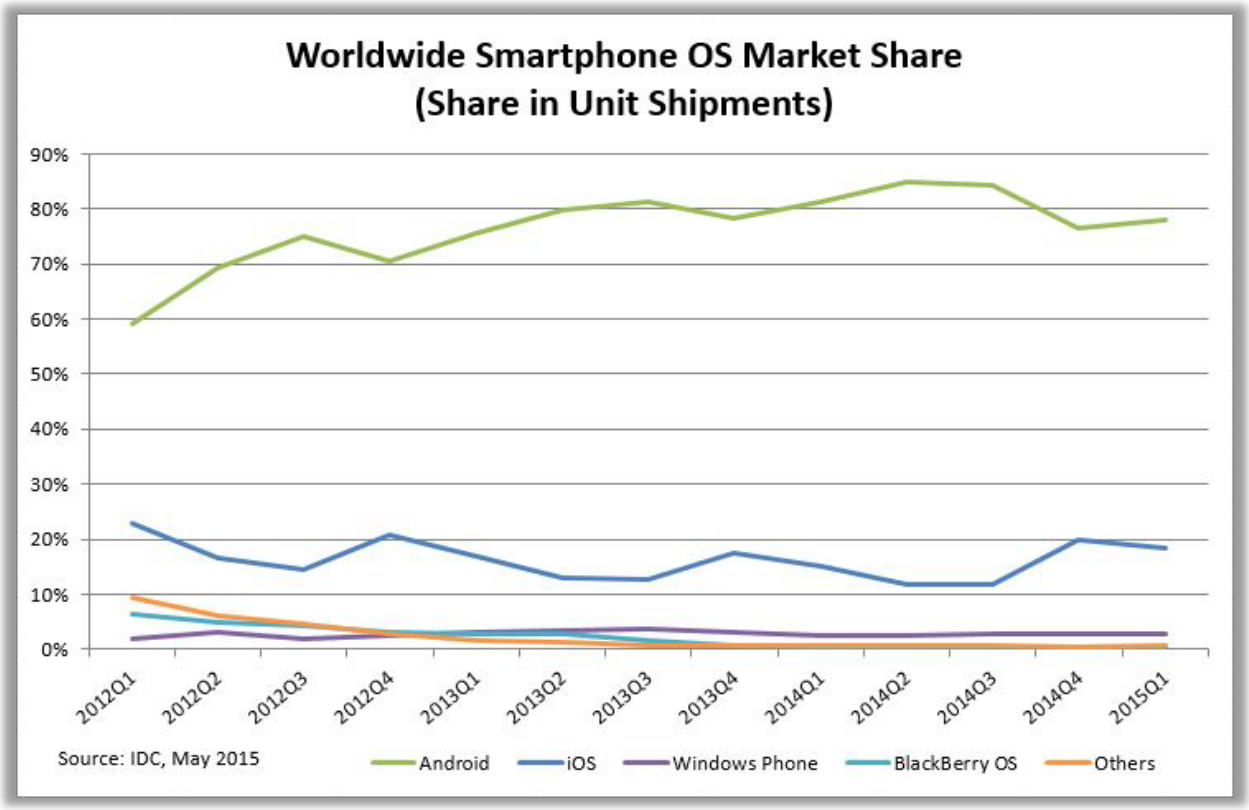
Market share

Like already mentioned before Android has the largest market share worldwide of all smartphone operating systems. And how? They possess more than 3 quarters of the worldwide audience since 2013 and has only been growing since. Sure sometimes we see small decreases now and then but like seen in following graph the trend is more or less in an inclining way.

They seem to battle it out with the iOS market most of the time, an increase for them mostly results in a decrease for Android and vice versa. So it is clear to us that if you want for your application to be just out there the better choice is certainly Android. But don't let these numbers fool you in thinking this is also the operating system where the most money is made, that part still goes to iOS. It often seems the Google Play store generates less revenue than the Apple Store. So Android users don't seem to buy applications as frequently as Apple users do.

Hereby we can conclude that if you want your application to just be out there and used as much as possible Android is the best choice. But if you are planning to generate the most revenue it is often wiser to first develop your application for iOS and Android comes secondly.

Below graph gives the market shares over the last years according to an article from IDC.com called *Smartphone OS Market Share, Q1 2015*:



Period	Android	iOS	Windows Phone	BlackBerry OS	Others
Q1 2015	78.0%	18.3%	2.7%	0.3%	0.7%
Q1 2014	81.2%	15.2%	2.5%	0.5%	0.7%
Q1 2013	75.5%	16.9%	3.2%	2.9%	1.5%
Q1 2012	59.2%	22.9%	2.0%	6.3%	9.5%

Source: IDC, May 2015

5 – Image-source: http://www.idc.com/prodserv/smartphone-os-market-share.jsp

LOLLIPOP (5.X)

Android Lollipop is the version of Android for handhelds announced on Google IO 2014 together with the new platforms discussed further in this section. Android Lollipop houses some new features together with some new compatibilities. Lately there has been a new Android version every year where they include a lot of new features and improvements to the platforms. From now on they will not only need to update their smartphone version of the OS, but also the versions for Wear, TV and Auto.

Compatibility

Compatibility can be a big challenge in general for Android. In every of the new versions new features are released and for these to be backwards compatible, extra code is needed. I needed to restrict my application to this version because of this and simplification purposes. So does for example Android Auto only work with Lollipop whereas casting to your Android TV or Chromecast is only available on devices which run version 4.4.2 (KitKat) or higher. On top of that Android Wear only supports version 4.3 (Jelly Bean) or higher, plus the new design principles from Google with their material design, which they want their developers to follow are natively only supported on Lollipop or higher.

You see that these new devices ask from you that you run a pretty up-to-date version of their smartphone OS, which basically means these devices are primarily build for the future and want people to upgrade to one of the later versions. Then again people who are interested in such devices are at the moment probably tech-savvy or early adopters in general.

So for these compatibility problems it was the best choice to just target Lollipop in this research environment also because we could possibly give up new features by targeting lower versions of Android. With this latest version I am able to go all out in the timespan I have and potentially utilise the new features present. Now that we are talking about these new features lets explain some of the most important newly added functionality in the next section.

New features

At first we can notice that the Android Lollipop from previous section not only runs on handheld devices, there is also a version for the other platforms but tailored for their specific usages and environments. This leads us to the first new feature and the most important one for my research. Now Android will run on all of the new devices or to say it with the words of the Android website: “The smarts of Android on screens big and small”. That is basically what this whole report is about the new feature Lollipop has to support all of these new platforms. This mainly for Wear and Auto, because they are still, at the time of writing, not independent operating devices without your smartphone. Of course there is often some basic functionality when not connected to your phone, but it is meant to be connected to gain the full experience. But later on that matter in the respective sections for each of these platforms. But now let’s talk about some other of the really nice features Android houses. I will only mention the ones important for my research, because a full list would be unnecessary and depart too much from the scope[1].

[1] For a full set of features please visit following web page of the official Lollipop website: https://www.android.com/versions/lollipop-5-0/#features

Material Design.

Probably the most obvious and clear change Google has made with their new version of the OS is the now famous Material Design, known for their smooth animations and often colorful layer pattern. According to Google's official Lollipop website it is (Other listed features in this section will always be quoted from this source):

- *A bold, colorful, and responsive UI design for consistent, intuitive experiences across all your devices.*
- *Responsive, natural motion, realistic lighting and shadows, and familiar visual elements make it easier to navigate your device*
- *Vivid new colors, typography, and edge-to-edge imagery help to focus your attention*

6 – Android Lollipop logo
Image-source: http://infoahead.com/wp-content/uploads/2015/02/android-lollipop.jpg

I can personally really dig this new design and it is a big improvement in my opinion just for the looks but also for offering a greater user experience with their intuitive UI. Even in this report I tried to use some small aspects of material design, look at the images most of them are designed with a Google Now card design.

Notifications.

Notifications are a big part for Android as well as for Android Wear and in the future according to the Android documentation it will also be part of Android TV. Following are features have been added or improved with Lollipop:

- *New ways to control when and how you receive messages - only get interrupted when you want to be*
- *View and respond to messages directly from your lock screen. Includes the ability to hide sensitive content for these notifications*
- *For fewer disruptions, turn on Priority mode via your device's volume button so only certain people and notifications get through. Or schedule recurring downtime like 10pm to 8am when only Priority notifications can get through*
- *With Lollipop, incoming phone calls won't interrupt what you're watching or playing. You can choose to answer the call or just keep doing what you're doing*
- *Control the notifications triggered by your apps; hide sensitive content and prioritize or turn off the app's notifications entirely*
- *More intelligent ranking of notifications based on who they're from and the type of communication. See all your notifications in one place by tapping the top of the screen*

The improvements on the interrupting of users with notifications is certainly a nice improvement over previous editions. It will not bother the user if not needed to and will therefore provide a better user experience. With this new iteration the user also gains some control on what notifications he or she receives by the ability to block the notifications of certain applications. This feature will later also be mentioned in the Android Wear section.

Connectivity.

Improvements on connectivity are also important because this will influence how data can be shared over various devices. To again let Google talk which improvements specifically were made look at the following:

- *A better internet connection everywhere and more powerful Bluetooth low energy capabilities*
- *Improved network selection logic so that your device connects only if there is a verified internet connection on Wi-Fi*
- *Power-efficient scanning for nearby Bluetooth low energy ("BLE") devices like wearables or beacons*
- *New BLE peripheral mode*

The new Bluetooth technology is like Google already mentioned ideal for devices as wearables. Also the Wi-Fi improvements can be beneficial for the Casting functionality Android possesses to for example an Android TV.

Media.
Because in my application media will be frequently shared between devices it is important to take a look at what improvements are made on this subject. Following are some of the most important ones:

- *Bolder graphics and improved audio, video, and camera capabilities*
- *Lower latency audio input ensuring that music and communication applications that have strict delay requirements provide an amazing real-time experience*
- *A range of new professional photography features for Android Lollipop that let you capture full resolution frames around 30 fps*
- *Control capture settings for the sensor, lens, and flash per individual frame*
- *Capture metadata like noise models and optical information*
- *State of the art video technology with support for HEVC to allow for UHD 4K video playback, tunneled video for high quality video playback on Android TV and improved HLS support for streaming*

The support for high video playback on Android TV is certainly interesting as well as new photography improvements which will enable users to take higher quality pictures, which will certainly be visible on the big screen.

Android TV.
Of course this feature is important because it is basically part of the subject of this report but let's still take a look on what support is added with the new Android platform:

- *Support for living room devices*
- *User interface adapted for the living room*
- *Less browsing, more watching with personalized recommendations for content like movies and TV shows*
- *Voice search for Google Play, YouTube and supported apps so you can just say what you want to see*
- *Console-style Android gaming on your TV with a gamepad*
- *Cast your favorite entertainment apps to your big screen with Google Cast support for Android TV devices*

Certainly the Google Cast functionality is interesting to us, because it will enable us to quickly share media from our phone to the big screen.

WEAR

Android Wear is a fairly new platform but today already has quite a bit of following, including myself. Therefore I am not only able to speak from the perspective of a developer/analyser but also from the standpoint of a personal user. I myself am in the possession of a Moto 360[2], but the operating system already runs on 7 distinct watches in total at the time of writing (2015) and this is even growing[3].

The amount of watches and compatible devices gives them a huge advantage over other smartwatches like the Apple Watch or the Samsung smartwatches. When you are in the possession of an iPhone for example your only choice is to buy the Apple Watch or one of the Pebble smartwatches. On the other hand if you want to buy an Apple Watch or one of the non-Android Samsung smartwatches, you need a compatible smartphone, in this case being respectively an iPhone or one of the flagship Samsung devices. So basically if you dislike their features, design or something else, you are pretty much out of luck. Instead Android Wear is compatible with every Android phone running 4.3 (Jelly Bean) or higher and it is even rumoured Google is working to expand Android Wear to also work with Apple's iPhone, but nothing is sure yet on that matter.

For more information on the current compatibility and different available smartwatch operating systems take a look at below table[4] fetched from the slides of the Android Wear subject in Capita Selecta II but edited by myself to only show relevant information to my Final Work:

	ANDROID WEAR	LG WEARABLE PLATFORM	WATCH OS	PEBBLE OS	TIZEN OS
Android compatibel	Yes	Yes	No	Yes	Only some Samsung flagships
iOS compatibel	No (not yet)	Unknown	Yes	Yes	No

Now what is Android Wear mainly used for and what is its focus? Well, in the early ages of the smartwatch even the manufacturers didn't seem to really know what a smartwatch could be used for and what benefit it would deliver over just picking your phone out of your pocket. Nowadays the world of smartwatches is still figuring this out a bit, but we have gone a long way since those early stages. Google for example has defined a clear vision on their point of view on what a smartwatch should be and what it should be used for. So in the next section I am going to summarize their vision and later on I will also talk about the most important design principles.

---

[2] More info on the Moto 360 on the official website: https://moto360.motorola.com/

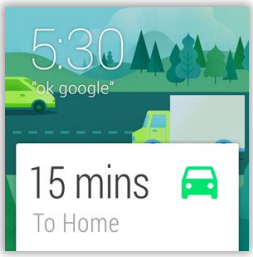
[3] For more info on all Wear devices visit the official Android Wear website: https://www.android.com/wear/

[4] Source: B. Van Rillaer, *Android Wear (Slides course Capita Selecta II)*

Vision

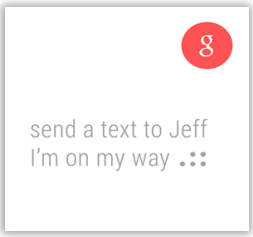
Following text is quoted from the official Android Wear website because how is Google's vision better explained then by themselves[5]:

Launched automatically.
Most people are used to launching apps by clicking an icon. Android Wear is different. Wearable apps are aware of the user's context - time, location, physical activity, and so on. The apps use this information to insert cards into the stream when they become relevant. This makes Android Wear timely, relevant and very specific.

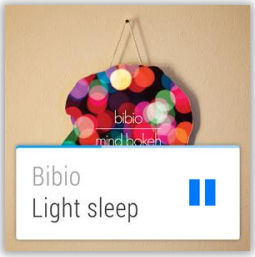


Glanceable.
A classic wrist watch is designed to let you see the time in a split second and get on with what you were doing. Designing for Android Wear is no different. The less time it takes to use your software, the more time the user can be present in whatever they are doing. Android wear is fast, sharp, and immediate.

All about suggest and demand.
Android Wear is like a great personal assistant: it knows you and your preferences, it only interrupts you when absolutely necessary, and it's always on hand to provide a ready answer. Android Wear is helpful, respectful, and responsive.



Zero or low interaction.
Staying true to the strengths afforded by a smaller form factor, Android Wear focuses on simple interactions, only requiring input by the user when absolutely necessary. Most inputs are based around touch swipes or voice, and inputs requiring fine-grained finger movements are avoided. Android Wear is gestural, simple, and fast. By providing a smart connection to the rest of the world while respecting the user's attention, Android Wear feels personal and global, simple and smart, unobtrusive and ever-ready. Applications that represent these principles will feel most at home in the overall Android Wear experience. Third party apps extend Android Wear to be more specialized and helpful throughout the day. Installing apps are a way for the user to tell the Android Wear how to do that.

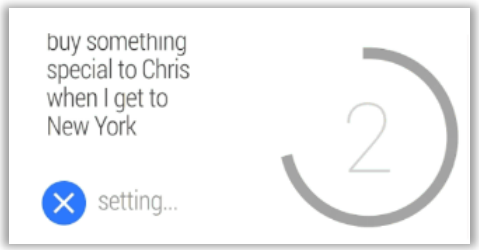


[5] Source: *Creative vision for Android Wear*, (https://developer.android.com/design/wear/creative-vision.html)

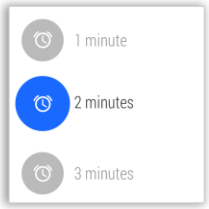
Design Principles

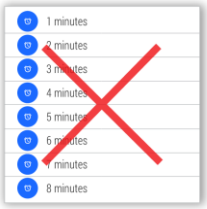
For the design principles I will also let Google talk rather than to summarise them myself because I feel like they already gave a very good short explanation themselves on what is important to take into account when developing for Android Wear. So following part is quoted from the official Android Wear website[6]:

Focus on not stopping the user and all else will follow.
A watch is a per fect form factor for a device that you can use while doing something else, such as cooking, eating, walking, running, or even having a conversation. If using your wearable app causes the user to stop whatever they're doing, it's a good occasion to consider how to improve it using the principles in this section.

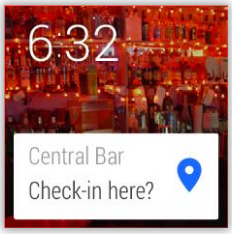


Design for big gestures.
When you swipe through photos on your phone, you're using a large area of the display and precision isn't required. That's the best kind of interaction for a wearable device. Your users are going to use your app in all sorts of situations, the least frequent one might actually be sitting down at their desk.





Think about stream cards first.
An app that offers to check in users could appear in the stream suggesting the most likely place nearby, after a certain amount of time. The best experience on a wearable device is when the right content is there just when the user needs it. You can figure out when to show your cards with sensors, or events happening in the cloud. For the cases where it's impossible to know when the user needs your app, you can rely on a voice action or touch.



Do one thing, really fast.
While users will engage with your app for only a few seconds at time, they'll use it many times throughout the day. A well-designed stream card carries one bit of information and potentially offers a few action buttons when the user swipes over.

Design for the corner of the eye.
The longer the user is looking at your app, the more you are pulling them out of the real world. Thinking about how to design your app for glanceability can vastly help the user get full value from your app and quickly go back to what they were doing.

Don't be a constant shoulder tapper.
A watch constantly touches the user's skin. Being this intimate, you want to vibrate the watch fewer times than you might on a phone.

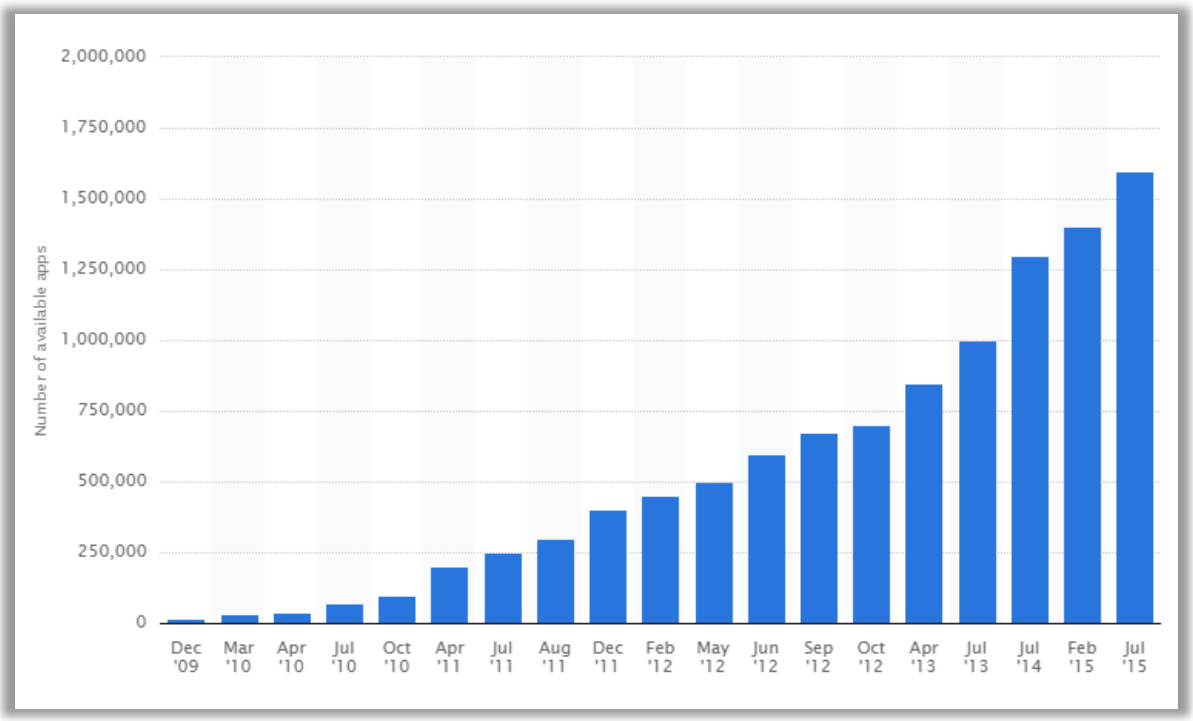
---

[6] Source: *Design principles for Android Wear,* (https://developer.android.com/design/wear/principles.html)

Features

Out of the box an Android Wear smartwatch houses the most functions the Google Now application on your phone does. This includes contextual notifications, like for example the weather at your location, a meeting coming up, the birthday of a friend, results of matches of your favourite sports team and much more. Besides this contextual information Google Now also functions as a personal assistant for information on-demand. Here you can ask basic questions or demands from the system. You could for example ask for the weather in a specific location at a specific time, ask for the time in another time region, ask for basic calculations, text or call somebody by just using your voice and so on. This is often the most important feature for an Android Wear smartwatch. It gives you quick access to helpful information often with a contextual background by using for example the time and your location.

Though this being the core of what Android Wear stands for, it is not the only thing it can do. We have to take into regard that the Android so familiar from our phones already has a Google Play store with more than 1.35 Million available applications dating from February 2015 according to an article from statista.com[7]:



This however doesn't mean all of these applications are going to support Android Wear, but it means there is a really large potential on the future features Android Wear could have. Often this is just limited by the sheer imagination of the developers and is thus surely expected to be growing.

[7] Source: *Number of available applications in the Google Play Store from December 2009 to July 2015*, http://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/

Personal experience & findings

Last but not least I also want to mention a few of my personal experiences with Android Wear and how it evolved in the past year of owning a smartwatch now. Yes that's right, at the moment of writing I already own a Moto 360 for almost a year now. I got it a few months after its release in September 2014. So regarding this I had plenty of time testing the software and hardware and since I'm an IT student I also couldn't help myself to take a peek at the development on the platform. But let me first talk about the normal user experience I had and what I personally envy and dislike about the platform in general.

The first aspect of the smartwatch I want to mention is the gathering of your personal data. Though this is something smartphones have been doing for years already if not configured otherwise. The only thing is, it only becomes a little more obvious that your devices are collecting a lot of data from you. For example if you visit a place pretty often, the smartwatch will begin to notify you of travel times to the specific location. Or if you've got a pending order, Google Now is also going to scan your e-mails to fetch order information whereas you then get notified on your smartwatch when the item has dispatched. I myself am not that much concerned about the whole privacy subject, because you get a lot of handy features in return. But this is certainly something you should take into account when shopping for a smartwatch. Also a smartwatch becomes pretty much useless without access to some of your data in my opinion.

Secondly with a smartwatch it is easy to feel what I like to call over-notified. If you don't manage the notifications you get it is easy to get a lot of uninteresting notifications which are not worth your time. But there is a solution for this in Android Wear and that is that you can block every application you want really easily from displaying notifications on your watch. This is especially handy with notification spamming applications like games often do to entice you in playing again. But this problem is not only the case with annoying notifications or applications. It sometimes also occurs in the basic Google Now features or other handy applications which sometimes seem to fail to provide you with environmental aware useful data.

To explain with an example I have an activity application which monitors your heart rate to calculate how many minutes of exercise you had that day. Though this application shows every day sometimes multiple times how many minutes of exercise I had, even if this was none at all. I would mostly be interested in notification like this when I am actually exercising and not when I have done almost none that day. A similar problem is to be found with the wheatear forecast. I seem to get a wheatear notification multiple times a day for both my work location and my home location. This is cool and all, but could be more location and time aware. In weekends for example why would I want to know what wheatear my work location has for that day? Same goes for times of vacation where it could, based on my location, see that I don't travel to my work anymore. These are only two examples of how over-notified you can get with Android Wear, often I get notifications which I just dismiss directly because I have no interest in them at that time, but I don't want to block them either because the notifications can be handy, just not in that specific place or time. Because of this it is also important to keep this in mind while developing apps compatible with Android Wear. You surely don't want to over-notify your users and thus force them to block your application because they get annoyed.

Okay previous two subjects I mentioned were kind of negative, but these are the only 2 negatives I could think of in a large sea of advantages. When it works like it's supposed to, you really feel that superior experience. Also Android Wear is updating pretty frequently and thus adding new features and maturity to the operating system.

Some feature I personally really like, nothing much really, but gives in my opinion such a huge advantage over the competition, is the watch faces. There are already hundreds, no thousands of watch faces available both paid and free ones. And these aren't just for aesthetics no, they can also implement certain functionality. For example there are watch faces which change based on the wheatear or watch faces that include certain information like temperatures, other time zones, step counting, other application info and many more. For me they also give me a possibility to change the watch face according to my mood and do I get bored of a watch face I just replace it with a different one. While this may not really be a huge deal and no reason to go right to the store and buy your own device, this certainly gives a nice touch to the experience of already owning one.

I also want to mention the countless applications already beginning to implement Wear functionality. While the available applications with really nice adding features are still slim, the trend is going up the right way and the store expands with Android Wear applications every day. But there are already some pretty handy apps and therefore I want to make some mentions applications I often use on my personal smartwatch.

For example for sure the most used application on my smartwatch is just "Find my phone". It is kind of funny but it is true, I often leave my smartphone be somewhere around the house, which is just about enough range to still get your notifications over Bluetooth, and thus I often forget where I left it. So then I open the application on my watch which makes my phone ring and emit a bright light. Another application that I often use is "Wear Mini Launcher". This basically adds an Android style application launcher to your watch when swiping from a configured side. But also enables some quick settings when swiped with 2 fingers or twice, which can also really come in handy.

There are much more examples I could mention but this would result in an array out of bounds exception (the programmers way of saying, I would get out of my scope). So In conclusion I can say it is a really nice device and can often make your life much easier, but because of its infancy still needs some work. Though I am confident Google will certainly pull this off, because they seem to listen to their users. So are smartwatches for everyone? Certainly not! At least not yet. While they can be surely beneficial to the productivity by enabling quick scanning of received mails, texts, etc. It is a device that at first needs getting used to and secondly is not something you need but is a nice gadget for the tech geeks between us. I am confident this will change over the years and smartwatches will grow to be a greater addition to the smartphone experience, but for this we still have to wait. For now it is rather a nice-to-have than a must-have.

TV

Often it is the case that when you are at home you would just like to show some pictures, video or even an audio track from those small phone speakers and screen to those big subwoofers and larger screen you have in your living room. Or maybe you have even thought that it would be nice to play a phone game like some of your favourite racing games on your TV with for example a controller.

7 - Android TV interface
Image-source: http://cdn.vrworld.com/wp-content/uploads/2015/01/Sony-Android-TV.jpg

Well with various devices now running Android TV or even with devices like Chromecast, getting your media on the big screen is easier than ever. The small screen of your phone can now be expanded with the large screen of your TV. It is even possible to use every Google Cast enabled device to stream your media to. For example say you'd have Bluetooth speakers which are also cast enabled, then you can cast your favorite audio tracks from for example a music app like Google Music to those speakers by the push of a button.

Though this report is mainly focused on Android TV, but I will mention some of the most well-known casting devices like the Chromecast further in the report. But let's first go deeper into the main part of this section Android TV and give an overview of the platform and what it can be used for. Learning from the official Android TV website[8] as well as third party articles and reviews[9] I was able to get a good view of what the device is and what is isn't. Sadly I haven't had the privilege to get some hands-on time myself with the operating system besides the Android emulators in Android studio.

---

[8] Source: https://www.android.com/tv/

[9] Sources :
- *Sony 49X830C - A 4K TV running Android!*, https://www.youtube.com/watch?v=im6a6wk80Ts
- *Android TV as Fast As Possible*, https://www.youtube.com/watch?v=m6YLiQXQRVg
- *NVIDIA SHIELD Android TV Review*, https://www.youtube.com/watch?v=wFZfCBXw6K4
- *Nexus Player Android TV Review*, https://www.youtube.com/watch?v=E5irc7QDwJ4

Compatibility

With a name as Android TV people often might expect that to benefit from such a service they have to again make large expenses to buy a TV and get all of the new functionality and app support from Android TV. This is a common misconception many make, but with a quick Google search it becomes clear this is not the case. You basically have the choice yourself! Let's say you need a new TV then you can choose to buy a new one running on Android TV from some TV manufacturer. On the other hand if you don't need a new TV or just don't want to spend the extra money, one of the available streaming boxes just might do that for you.

One of the problems that is still persisting to this day with Android TV are the available applications. Not enough applications have been ported yet at the time of writing. So is there for example no good browser available for the platforms which is really a pity. But this will probably just be a temporary problem and is just the cause of its infancy. To get the trend going developers like me will need to take action and develop handy applications for the platform. But I personally am confident this will happen, it's just a matter of time. There is a good chance the next TV you buy will be one running the Android operating system.

There are already plenty of TV manufactures going onboard on the Android TV train like Sony and Phillips with more still on the way. Next to them there is also a Nexus device called the Nexus layer which functions as a streaming box. But Android TV also catches the eye from some gaming related companies like Razer and NVidia who see it as a nice opportunity for an alternative gaming console. If you want more information on the available devices, you can take a look on the official website or one of the YouTube videos I mentioned for my research.

Anyways it quickly becomes clear Android TV is destoned to be many things and that is what the next section is going to talk about, the features.

Features

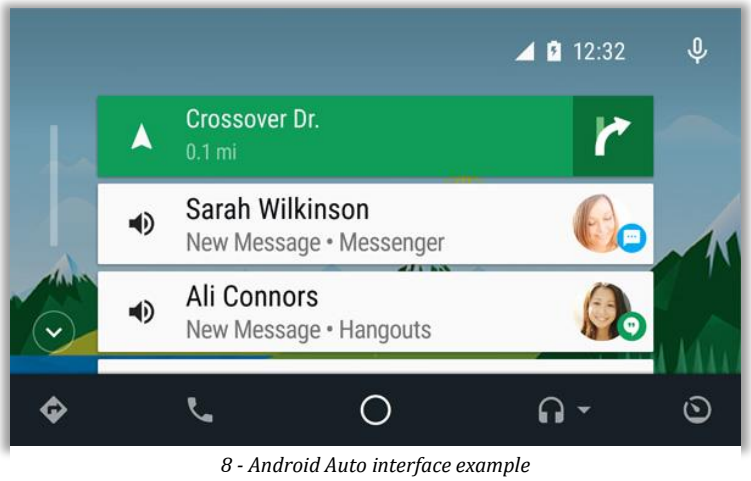
Android TV brings new functionality to the table now all available on the so familiar large screen. The OS will enable you to watch Blockbuster movies and live sports, streaming apps and multiplayer games. In short Android TV brings a world of content, apps and games to your living room.

Of course when you think about Google you think about personalization. This is done by giving you recommendations by your activity history throughout your Google or other application accounts. So do for example applications like YouTube, Hulu or Netflix all store your watch history and can therefore recommend similar videos to the ones you watched. This leaves a nice user experience where the user is directly confronted with most likely useful information on the home page. On top of this Android TV, like many of the other new technologies from Google highly depends on voice activation which is also improving really rapidly from my own experience. This also enables basic Google Now functionality like asking for the weather, but doesn't possess the full package yet, like your smartphone does. For example making or viewing appointments at the time of writing was still not possible.

Besides the many things you can watch on your TV with Android TV, this time they also largely anticipate on the possibility of Android TV devices functioning as a gaming console. Where you are able to stream your games to the big screen with superior quality and experience. Some devices like the NVidia Shield even offer special services where you can game from games running in the cloud or run and stream games from your gaming PC with all its power.

AUTO

Android Auto is sadly not too available for me to test out or to emulate from Android Studio, but what I could do was look at some reviews[10] and read up on the documentation[11] to get a global view on what its purpose is. Android Auto is currently only available in following countries according to Android Auto's website: *Australia, Canada, France, Germany, Ireland, Italy, Mexico, New Zealand, Spain, the United Kingdom and the United States.*



8 - Android Auto interface example
Image-source: official Android Auto page mentioned in the foot note.

Android Auto is therefore something I'm not going to be able to get very deeply into because not very much information is available yet. But though I got as much information as possible on the platform and tried my best to still give a global view on the new technology.

The Operating system basically wants to solve that Car operation systems have often been way to slow, offer a bad user experience and include lots of bugs. That is basically what you get when you set out a car company, with no knowledge of software and usability, out to make an operating system to configure almost everything the car can do. From the music and navigation to the air-conditioning and ventilation. For the most part these experiences have been really bad, but that is where Android Auto comes in as a savior.

Android Auto is built to be a clean, intuitive and an easy to use interface with basic functionality specifically for your car. Its purpose is basically to only provide you with information that is beneficial during driving and not cluttering you with unwanted, unnecessary notifications which would distract you from the road. Basically the message Google wants to bring across with Android Auto is to keep your eyes on the road and get informed from strictly necessary stuff via voice actions and sound feedback. Therefore it should be much safer and keep you from using your phone in the car.

There is also Google Now functionality but tailored to the usages in your car, to give only one example, locations you recently searched for will come up on Android Auto and you have the possibility to navigate to the location by the push of a button. Let it be clear that Android Auto needs to be connected to your phone for the optimal user interface like Android Wear. Besides Android Auto is only compatible with Android Lollipop and higher. So it is clear this OS is also build for the future and will probably be an option in many cars in the near future.

---

[10] Sources:
- *Android Auto Review! (Hyundai Sonata 2015)*, https://www.youtube.com/watch?v=xlsV8ModOgY
- *Android Auto review*, https://www.youtube.com/watch?v=-FL9kKqOQxI

[11] Source: https://www.android.com/auto/

Data-Sharing Analysis

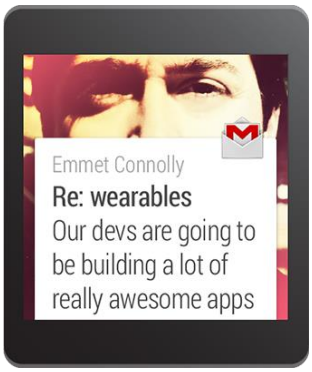
In my application the main data that would be shared would be different types of media probably mixed with actions to perform on this media and other messages. In following section I will discuss the different ways of synchronisation of this media between the previously discussed platforms.

While I already gave a broad view of the platforms and their respective usages and features, I will now give a more detailed perspective on the data sharing capabilities of each of them. I will also already give some more information about the development I planned to make myself familiar with the most important of the mentioned techniques.

ANDROID WEAR

Android Wear is all about the quick presentation of media to the user, with very little distraction. It is meant to function as a quick and often hands-free and voice activated experience to the often more feature rich smartphone counterpart. So first of all let it be clear that a smartwatch in the Android Wear environment and debatable on every environment should not try to be a replacement for smartphone functionality, but rather leave an additional user experience.

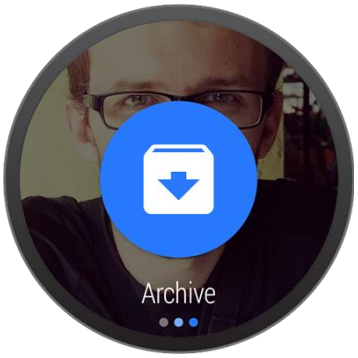
Therefore the main functionality of a smartwatch is to notify the user of important data taking into account the



Notifications

Notifications are the most important part of the Android Wear operating system and that is therefore also an important way of displaying and sharing data from your phone to your users. They can give the quick and glanceable experience Android Wear is built for and therefore be an ideal way to quickly show and let the user interact with the data you want to share.

Standard notifications are immediately and automatically displayed on the Android Wear device without even adding a single line of code to your existing application. Android automatically uses the already existing notification code and then displays it accordingly on the wearable.



Of course there are also some wearable specific notification possibilities. Or let's say you have some functionality that only should be accessible from your wearable and not from your phone then this is also possible. You just have to change a little bit of your notification code to implement this.

Let me now go a little deeper on what is specifically possible with these notifications on wearables[12]:

First you have the standard notification. This notification should be provided with at least a notification title and body as well as a content intent which provides information on what should be done when the notification would be opened.

Secondly you can add actions to the notification which will show up on the wearable like shown in one of the wearable pictures on the previous page. These actions will both show up in the phone notification and on the wearable unless you specify otherwise.

Then for some notifications it is also wise to specify a big expanded view to the notification. On a wearable device this is visible by default, but on a phone this must be specified for the notification to be expandable. Also if you want to edit the content of the notification in comparison to the standard notification this should also be done here.

Last but not least you can specify wearable specific notification functionality like adding pages or added voice input functionality. You could for example use extra pages to provide the user with more contextual glanceable information about the notification and to better decide which quick action to choose.

Android Wear will with the easy implementation of its notifications primarily be used as a messages sharing device. Weather this are actions your phone should perform after receiving a notification or contextual information on the specific notification or maybe even voice input to perform your actions.

Wearable Data Layer

For following section I will let the Google documentation[13] explain some important concepts of data sharing capabilities in Android Wear using the Wearable Data Layer API. In the development part I will then be able to explain which methods I tried to use and why. I chose to let these important API parts be explained by the Android documentation to be certain the meaning of them is clear and rightly defined:

The Wearable Data Layer API, which is part of Google Play services, provides a communication channel for your handheld and wearable apps. The API consists of a set of data objects that the system can send and synchronize over the wire and listeners that notify your apps of important events with the data layer:

Data Items.
A DataItem provides data storage with automatic syncing between the handheld and wearable.

[12] Source: *Creating a Notification for Wearables*, https://developer.android.com/training/wearables/notifications/creating.html
[13] Source: *Sending and Syncing Data*, https://developer.android.com/training/wearables/data-layer/index.html

Messages.
The MessageApi class can send messages and is good for remote procedure calls (RPC), such as controlling a handheld's media player from the wearable or starting an intent on the wearable from the handheld. Messages are also great for one-way requests or for a request/response communication model. If the handheld and wearable are connected, the system queues the message for delivery and returns a successful result code. If the devices are not connected, an error is returned. A successful result code does not indicate that the message was delivered successfully as the devices may disconnect after receiving the result code.

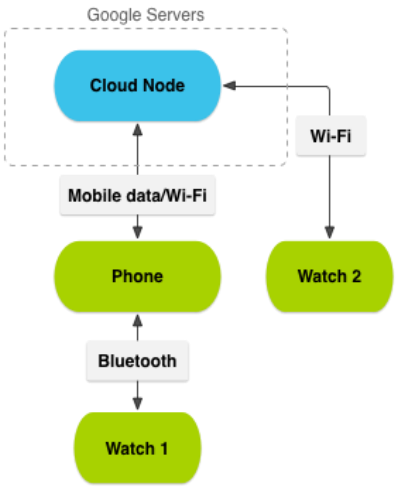
Asset.
Asset objects are for sending binary blobs of data, such as images. You attach assets to data items and the system automatically takes care of the transfer for you, conserving Bluetooth bandwidth by caching large assets to avoid re-transmission.

WearableListenerService (for services).
Extending WearableListenerService lets you listen for important data layer events in a service. The system manages the lifecycle of the WearableListenerService, binding to the service when it needs to send data items or messages and unbinding the service when no work is needed.

DataListener (for foreground activities).
Implementing DataListener in an activity lets you listen for important data layer events when an activity is in the foreground. Using this instead of the WearableListenerService lets you listen for changes only when the user is actively using your app.

Channel.
You can use the ChannelApi class to transfer large data items, such as music and movie files, from a handheld to a wearable device. The Channel API for data transfer has the following benefits:

- Transfer large data files between two or more connected devices, without the automatic synchronization provided when using Asset objects attached to DataItem objects. The Channel API saves disk space unlike the DataApi class, which creates a copy of the assets on the local device before synchronizing with connected devices.
- Reliably send a file that is too large in size to send using the MessageApi class.
- Transfer streamed data, such as music pulled from a network server or voice data from the microphone.



Android Wear supports multiple wearables connected to a handheld device. For example, when the user saves a note on a handheld, it automatically appears on both of the user's Wear devices. To synchronize data between devices, Google's servers host a cloud node in the network of devices. The system synchronizes data to directly connected devices, the cloud node, and to wearable devices connected to the cloud node via Wi-Fi.

ANDROID TV

For Android TV there is mainly one way of data sharing available namely Google Cast. In following section I will explain more on how Google Cast works and also a bit about its implementation.

Google Cast

Google Cast basically enables users to cast media from your smartphone to a casting enabled device like in this case Android TV, but is also possible with devices like Chromecast. The concept uses the principle of a sender and receiver application. Quoted from the official developer documentation[14] a quick view of what these applications can be:

- *A sender application, written for the Android, iOS, or Chrome platform which uses the following Cast APIs: Android API, iOS API, Chrome API*

- *A receiver application that handles communication between the sender app and the receiver device. You have the following options:*
    - *The Default Media Receiver presented with the Google Cast branding and styling.*
    - *The Styled Media Receiver for which you can develop the styles and branding. See Styled Media Receiver.*
    - *A custom receiver, as described in Custom receiver that implements the Receiver API and handles custom messages from your sender app; it may also interface with the media player types provided through the Media Player Library.*

So basically this means in an Android TV context that the sender app on your phone will operate in following way: First the phone app (sender) will start the MediaRouter's device discovery. Then the Media Router informs the sender app of the route the user selected and the app retrieves a CastDevice instance. Now the sender app creates a GoogleApiClient and tries to connect. The SDK confirms that GoogleApiClient is connected and if successful the sender app launches the receiver app. This time the SDK confirms that the receiver app is connected. On which the sender creates a communication channel and when established he is able to send messages over this channel. The connection is then established and the user can operate the cast enabled Android TV with his/her smartphone. So can the user now for example pause a video or music track or just change the volume.

This all gets done by just sending actions to the TV. Don't get mistaken in that the full media file gets distributed. Let's take for example YouTube. Here when you want to cast a video the sender will of course not send the full video but instead it will just send the link to the video and the action to play this video. After this is done and a connection was established the user can perform all sorts of according actions.

---

[14] Sources:
http://www.google.com/cast/
https://developers.google.com/cast/
Get Started, https://developers.google.com/cast/docs/developers
Android Sender App deployment, https://developers.google.com/cast/docs/android_sender

Application development

As an application I basically wanted to implement different types of media sharing for the different platforms I researched. For one I must say I couldn't look into development for Android Auto because no SDK was released yet during the development of my application. So I had to do it with Android Wear and TV for my functionality testing application.

In the following subject I will explain how I went to developing the application for the Final Work and the methods and technologies I used. I will also explain some problems I had and what I did or tried to do to find a solution for them. In the end I will give an overview of the results I achieved into the application.

PROGRESS DESCRIPTION

At first in the early stages of the development in the autumn of 2014, I started trying to develop an application for which the scope later would seem way to big. It was my plan to develop an application for every of the mentioned platforms, with all types of media I could think off and even more so I wanted to make an application that would provide a real life usage with design, login and everything included. Of course back then I didn't yet understand Android development and therefore had no idea what I was getting myself into. Basically what this idea would ask me to do is. To develop the design as well as a complex server backend and the normal frontend functionality of Android all in one application on my own. This just isn't possible in the specified time combined with other school projects and therefore I had to take another approach on the whole Final Work.

Another mistake I made was to not begin with the core functionality of the application. I began with development of design and login instead of immediately trying to develop data sharing functionality. This I did because these are the parts I was most familiar with from developing other applications. I made the mistake to not immediately push myself out of my comfort zone and therefore the consequences were not good.

So after realising my mistakes I had to take a whole new approach and I started the development at least from the ground up with the parts of the analysis I already did. Now I was determined to start from the core functionality and that would be media sharing with Android Wear to later expand to the other platforms.

For sharing media on an Android Wear device I first wanted to take a look at the development of notifications inside my app. I thought to start the application with the concept of taking a picture followed by sharing the picture with for example a preview to my watch in a notification and then also couple actions to that picture like for example opening the picture on the phone, casting it to an Android TV or delete it.

So the first step in developing this functionality was to capture media on my phone and just displaying the Image on my phone without the use of a wearable. This I could successfully develop but not without any problems, but these will be mentioned in the next section.

The next step was then to develop the notification functionality for which I had to learn lots of different new stuff. Like two of the most important ones: the PendingIntent and the BroadcastReceiver.

The PendingIntent is an object which encapsulates a normal Intent but is especially useful for notifications, because they keep alive even if the application is terminated. It is handled by the Android OS and not by your application itself.

The BroadcastReceiver often is used in conjunction with a PendingIntent because the broadcast basically does what the name says and that is receiving a broadcast. So what you can basically do is configure the PendingIntent to send a broadcast when a specific notification action was fired and then the receiver will catch this broadcast even if the application isn't running. After the broadcast was received an event handler will be called and you can perform matching functionality like for example performing a specific action or starting a certain activity.

These two concepts I had to use to for example delete an image. This should be available even without reopening the app again so therefore a PendingIntent in combination with my BroadcastReceiver would be an ideal implementation for this functionality.

I succeeded in developing this and next I wanted to be able to show a preview of the taken picture on my watch, so a user would be able to show a quick preview of the image on his watch to then decide if he wants to keep it or not, or choose to open or cast it to get a better view of it first.

The next step was to implement functionality to cast this picture to a google Cast enabled device, basically Android TV. But due to time shortage I wasn't able to complete this code and seemed to be more difficult than imagined at first.

The road of the development I just described was bumpy, but I certainly learned a lot. In the next section I will take about the problems I encountered and my attempts to solve them.

PROBLEMS

One of the first problems I encountered was the one of limited available memory on handheld devices as well as on wearables. I had to load my Bitmaps efficiently to counter this problem. Often your camera takes much larger pictures then the resolution of your screen so it is unnecessary to load the full quality picture inside the memory and use functionality to scale the images according to the required size.

Another problem I encountered were the Activity Lifecycle events in combination with the use of fragments. I needed to make sure the data of my picture was not lost when the application transitions between different lifecycle states. To solve this problem I had to implement different checks and Intent extras together with the SavedInstance functionality and PersistingContext to make sure the needed data was saved whenever it was needed, but not needlessly persist data when not needed.

Another big problem I encountered was that Android Wear doesn't support custom layouts for its notification pages. I had to come up with a solution on my own to display a review of the image. I basically ended up displaying the image as a background-image of the notification and then hiding the text overlay every default page has.

Conclusions

In following section I will make conclusions to the previously described research and development about the subject. I will give an answer to the research questions as well as give a summary on each of the researched platforms. And last but not least I will evaluate myself and look if I have accomplished the goals I set for myself.

ANDROID WEAR

Android Wear is a nice addition as a gadget to the current smartphone experience and is growing pretty fast in its functionality and usages. But at the moment of writing the device certainly isn't supposed for everyone. It is more a gadget for the early-adopter or a productivity tool for developers and other tech geeks.

The device doesn't really do anything your phone can't do and in my opinion that is the biggest reason most people don't jump on the band waggon of smartwatches. The only real benefit it leaves is the glanceability and quickness of the interactions over the phone. You can discover in a very quick way if the notification is worth your time or just dismissible. It often takes just as much time as checking the time on an ordinary old fashioned watch.

Another big advantage is the contextual information the watch provides according to the location and time. But this certainly needs improvement and often leaves users 'over-notified'. Another problem that is still there today is that Android Wear needs more nice apps, but the support for this is certainly growing. For example only recently Google Hangouts has also a full-fledged Android Wear version where I can scroll through my conversations without touching my phone. With improvements like this we are certainly going the right way.

ANDROID TV

Android TV is the ideal operating system for people who want to bring the so familiar Android experience from their other devices to the big screen and sound system in the living room. They are able to watch movies, series, YouTube all via this platform and it can even function as a somewhat less powerful gaming console. But then again the power doesn't matter if you know that some devices even provide game streaming from your own desktop PC or as a service from the cloud.

But though Android TV has these nice features they also suffer from a slight shortage of apps but then again this is something that is probably going to change in the future certainly if you hear that some large TV manufactures and gaming companies are on-board with the platform.

ANDROID AUTO

Android Auto is all about the striped down, to the core version of Android, which certainly doesn't try to distract you with unnecessary notifications and only shows the most important interactions from your phone together with some Google Now functionality.

It's about not having to use your phone in the care which is very unsafe and rather relies primarily on voice input and sound feedback while also enabling an intuitive, to the point, touch interface.

The platform is still not very common, but is already an option on many new cars in some countries and leaves a significant improvement on previous car operating systems which offered really bad user experiences.

RESEARCH QUESTIONS

What are the real-life applications of every of the new platforms announced, namely Android Wear, -TV and –Auto?
Android Wear is basically a notification device which lets you it do the actions you normally do on your phone quicker plus it provides you with contextual information on the right place and time though this needs some work.

Android TV is an improvement on the currently known streaming box with its intuitive UI, easy user experience, voice integrated search, and in the future the Google Play store to back the platform up with a lot of apps.

Android Auto is such an improvement over older by the car company developed operating systems. It generally leaves a much better user experience and tries to be less of a distraction and keep your eyes on the road.

How is data shared between the devices or how do they interact with each other? Do they possess a storage capacity of their own or do they depend on another way of data storage?
Are the development platforms possible to be used independent of the standard smartphone version or are they only an addition in functionality on top of the normal experience?
All of the devices will possess a storage of their own but will not necessarily depend on it. So Android Wear does always sync its data with the smartphone where the smartphone functions as host and the smartwatch as a client. This basically means that when you'd delete a watch app from your phone it's also deleted from the wearable. Same goes for Android Auto which fully operates on your phone. Only Android TV is meant to be used fully independent from the smartphone, but f course still enables functionality between phone and TV.

How does development look for each of these devices more specifically on the subject of data sharing? What possibilities does Android offer?
Upgrading from a phone app to an Android Wear app is really easy if the phone app already exists. But I had the problem I first needed to create a phone app which costed me a lot of time.

RESEARCH GOALS

At last I want to talk about the research goals I set and if I succeeded in completing them. I am of the opinion I partially did. I delivered a good analysis of the platforms and their properties and I think I was able to give a good answer on the research questions.

I think readers of the report will have a better understanding of the platforms like I achieved during the work on this project. So that goal I did achieve. I was able to broaden my knowledge with these platforms and write down my analysis accordingly.

Though I feel that on the development part my progress could have been better. I wanted to have more functionality finished but due to time shortage and persisting problems this wasn't possible. Also because of the early mistakes I lost a lot of valuable development time and that is something I learned from. I learned to always start developing from the core of the application to quickly gain results and later nice-to-haves can be added if there is time left.

I want to conclude this Final Work by mentioning it was certainly an interesting subject and I have learned a lot which was the biggest goal of them all.

Bibliography

- Official Android Lollipop website (feature list), https://www.android.com/versions/lollipop-5-0/#features
- Official Android Wear website, https://www.android.com/wear/
- B. Van Rillaer, Android Wear (Slides course Capita Selecta II)
- Design principles for Android Wear, https://developer.android.com/design/wear/principles.html
- Number of available applications in the Google Play Store from December 2009 to July 2015, http://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/
- Official Android TV website, https://www.android.com/tv/
- Sony 49X830C - A 4K TV running Android! https://www.youtube.com/watch?v=im6a6wk80Ts
- Android TV as Fast As Possible, https://www.youtube.com/watch?v=m6YLiQXQRVg
- NVIDIA SHIELD Android TV Review, https://www.youtube.com/watch?v=wFZfCBXw6K4
- Nexus Player Android TV Review, https://www.youtube.com/watch?v=E5irc7QDwJ4
- Android Auto Review! (Hyundai Sonata 2015), https://www.youtube.com/watch?v=xlsV8ModOgY
- Android Auto review, https://www.youtube.com/watch?v=-FL9kKqOQxI
- Official Android Auto website, https://www.android.com/auto/
- Creating a Notification for Wearables, https://developer.android.com/training/wearables/notifications/creating.html
- Sending and Syncing Data, https://developer.android.com/training/wearables/data-layer/index.html
  Official Google Cast website, http://www.google.com/cast/
  Official Google Cast developer page, https://developers.google.com/cast/
  Get Started, https://developers.google.com/cast/docs/developers
  Android Sender App deployment, https://developers.google.com/cast/docs/android_sender