BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS
AND MULTIMEDIA

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

CONCEPT FOR REAL-WORLD PERSON AVATAR USAGE IN VIRTUAL CITY

KONCEPT VYUŽITÍ AVATARA SKUTEČNÉ OSOBY VE VIRTUÁLNÍM MĚSTĚ

BACHELOR'S THESIS
BAKALÁŘSKÁ PRÁCE

AUTHOR FREDERIK MÜLLER
AUTOR PRÁCE

SUPERVISOR Ing. VÍTĚZSLAV BERAN, Ph.D.
VEDOUCÍ PRÁCE

BRNO 2016

Zadání bakalářské práce/19005/2015/xmulle20

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2015/2016

Zadání bakalářské práce

Řešitel: **Müller Frederik**

Obor: Informační technologie

Téma: **Koncept využití avatara skutečné osoby ve virtuálním městě**
Concept for Real-World Person Avatar Usage in Virtual City

Kategorie: Uživatelská rozhraní

Pokyny:

1. Seznamte se s tvorbou moderních mobilních aplikací se zaměřením na zobrazování 3D grafiky a geolokační funkce. Dále nastudujte metody testování použitelnosti a seznamte se s konceptem Virtual Oulu.
2. Navrhněte mobilní aplikaci na geolokaci a interakci ve virtuálním městě (Virtual Oulu). Proveďte testování na uživatelích pomocí mockup verzí GUI a aktualizujte návrh.
3. Implementujte navrženou aplikaci s využitím existujících nástrojů a knihoven.
4. Aplikaci a její rozhraní otestujte na relevantním vzorku uživatelů a výsledky vyhodnoťte a diskutujte.
5. Vytvořte plakát a krátké demonstrační video reprezentující Vaše řešení.

Literatura:

- B. Shneiderman, C. Plaisant. *Designing the user interface: strategies for effective human-computer interaction*, Addison-Wesley Computing, ISBN-10: 0-321-53735-1, 2009
- D. Stone, C. Jarrett, M. Woodroffe, S. Minocha. *User Interface Design and Evaluation*, Morgan Kaufmann, ISBN-10: 0120884364, 2005

Pro udělení zápočtu za první semestr je požadováno:

- Body 1, 2 a částečně bod 3.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Beran Vítězslav, Ing., Ph.D.**, UPGM FIT VUT

Datum zadání: 1. listopadu 2015

Datum odevzdání: 18. května 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
L.S.
612 66 Brno, Božetěchova 2

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstract

This bachelor's thesis is focused on creation of mobile and web application using geolocation and 3D graphics. Theory about application development, methods of testing and existing solutions are described in the beginning. Next, designed applications which correspond with user needs are introduced and algorithms on which applications run are also described. Finally, last part evaluate accomplished results and propose possible improvements.

Abstrakt

Tato bakalářská práce se zabývá tvorbou mobilních i webových aplikací s využitím geolokační funkce a 3D grafiky. Nejprve je popsána teorie – technologie pro tvorbu aplikací, metody testování, existující řešení a jejich využití. Následně jsou představeny navrhnuté aplikace s ohledem na požadavky uživatelů a popsány algoritmy, které zabezpečují funčnost aplikace. Závěr obsahuje zhodnocení výsledků, které byly dosaženy a návrh možných zlepšení.

Keywords

mobile application, geolocation, virtual world, user interface, iOS, Swift, Virtual Oulu, 3D web application

Klíčová slova

mobilní aplikace, geolokace, virtualní svet, uživatelské rozhraní, iOS, Swift, Virtual Oulu, 3D web aplikace

Reference

MÜLLER, Frederik. *Concept for Real-World Person Avatar Usage in Virtual City.* Brno, 2016. Bachelor's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Beran Vítězslav.

Concept for Real-World Person Avatar Usage in Virtual City

Declaration

Hereby I declare that this bachelor's thesis was prepared as an original author's work under the supervision of Prof. Vítězslav Beran. Supervision at Oulu was provided by Dr. Matti Pouke. Supplementary information was provided by Prof. Timo Ojala. All the relevant information sources, which were used during preparation of this thesis, are properly cited and included in the list of references.

....
Frederik Müller
July 31, 2016

Acknowledgements

I would like to thank Ing. Vítězslav Beran, Ph.D. for leading and a lot of advice, which he was giving me while I was working on this thesis. Also, special thanks to Prof. Ojala and Pouke Ph.D. because without them this work couldn't be created. Last but not least, one huge thanks to all supporters.

Contents

Chapter 1

Introduction

This thesis deals with usage of geolocation function and 3D graphics in mobile phones and personal computers. The goal of thesis is to design and create application which uses gelocation and where users can interact between them in virtual city. Also, final work will be used to expand collaborative project Virtual Oulu at University of Oulu.

In order to create high quality application, we need to do research in existing solutions which is similar to ours so in the beginning these applications are introduced and explained how they work and what they are used for. There is also introduction about how to develop application and then how to test it. Then, there are few words about concept of Virtual Oulu.

Later, I introduce designed native mobile application and application for web browser, then modify it according users needs. Next, there is shown realization of both application, where I explain functionality and implementation of interesting part, mainly working with database, using geolocation in mobile application and 3D graphic in web application.

At the end, application is tested on users and then there is a summarization what was successfully done and what wasn't.

Chapter 2

Augmented Virtuality in smartphones

This sections contains introduction to trend in usage of geolocation function and 3D graphics in mobile and computer industry. I show existing applications and what they are used for. Then I introduce development technologies for mobile devices and explain how to do usability testing to evaluate an application. Concept of Virtual Oulu is presented in the end of this chapter and different types of virtual continuum are described.

2.1 Mobiles phones

Nowadays, mobile phones are part of our daily lives. Almost every EU citizen has own at least one smartphone. By 2018, over a third of the world's population is predicted to own a smartphone, which mean approximately 2.5 billion smartphone users in the world [5]. Mobile phones are not only about telephony. Is is also a miniature computer. Smartphones are mobile phones featuring messaging, internet access, camera, GPS, etc.

Of course, nowadays smartphones have got high performance and some of them are as powerful as personal computers. The development of smartphones is very similar to the development of computers. It means that smartphones have got processor, memory, RAM and graphic accelerator. The biggest difference is in the energy consumption and size of used parts. The energy consumption is one the main topic in mobile industry. Manufacturers are trying to offer higher performance every year but it is more and more difficult to maintain or increase the time you use your phone on a single charge. Contrary to the personal computers we do not have this problem at all, because computers are plugged right into power socket all the time. Mainly thanks to gamers, manufacturers in PC are trying to improve performance of each element in order to satisfy high requirements from the gaming community. But why for gamers ? Because they are willing to pay huge amounts of money in order to play newest games. On the other hand, situation in mobile business is quite different. Mobile phone manufacturers are trying to make life easier in general.

Smartphones have a lot of sensors and modules such as accelometer, gyroscope, GPS, camera and many others. It gives us opportunity to create application which runs f.e. 3D graphic and use GPS at the same time. It caused that in App Store were available 1,5 millions applications in 2015 [2] and the number of applications is still increasing because there are countless possible applications. It is not only about games but there are application for health and fitness, education, finance, kids, navigation, weather and many others categories. Social media are becoming still more and more popular so we can interact with other people almost immediately thanks to internet connection and application running on smartphones.

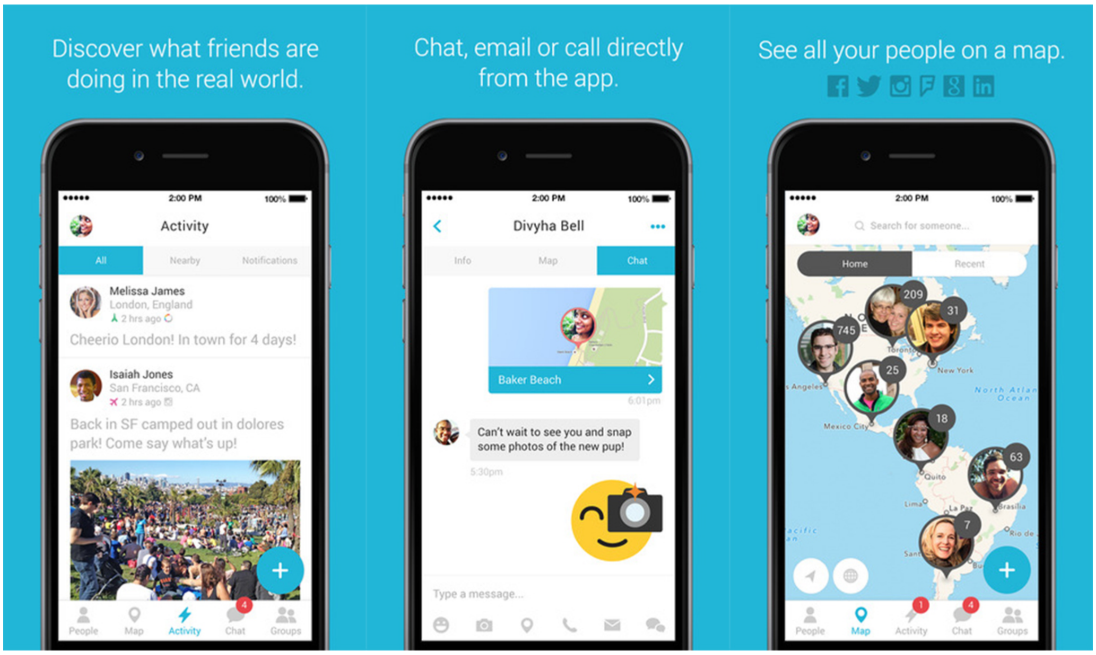
2.2 Existing applications

It is really important to get know about similar solutions. We can use useful features and avoid things that we do not like. We can find many applications which use geolocation or 3D graphics but it is quite difficult to find application which uses both at the same time. Due to definition, augmented reality applications use 3D graphics. Augmented reality refers to any case in which an otherwise real environment is „augmented“ by means of virtual (computer graphics) objects [6]. Some of these augmented reality applications use geolocation. There is also described one interesting application which uses geolocation and includes social aspect.

Layar is first augmented reality application for iOS. It was created in 2009. This application uses camera and GPS. It shows place around and acquired picture overlap with information about banks, restaurants and rent an apartments.

TwitARound is application which uses GPS, camera, accelerometer and compass. Application shows place around and shows where people were when sending the tweets. Unfortunately, this app has not been accepted by Apple to App Store because their rules.

Connect is a location-powered messaging app. The application has got many features. You can see all your friends on a map. There is possibility to discover what cities friends live in and where they've checked in recently. It notifies you when friends post nearby – on Instagram, Facebook, Foursquare, Twitter, or Connect. You can also share your real-time location with friends or groups. Chat, call, and email directly through the app is possible. And it is not all features. User interface is modern and intuitive but so many features can be considered as a disadvantage. Application requires too much information in the beginning. This long registration process can discourage new users. Maybe it is one of the reason why this application is not so well-known, although some parts of this application is quite similar to Facebook Messenger and Facebook application. For that reason, applications should be very simply to use for users and should be used in one particular activity. Despite all above mentioned disadvantages, application is quite well designed and functional.



Cheerio London! In town for 4 days!

Back in SF camped out in dolores park! Come say what's up!

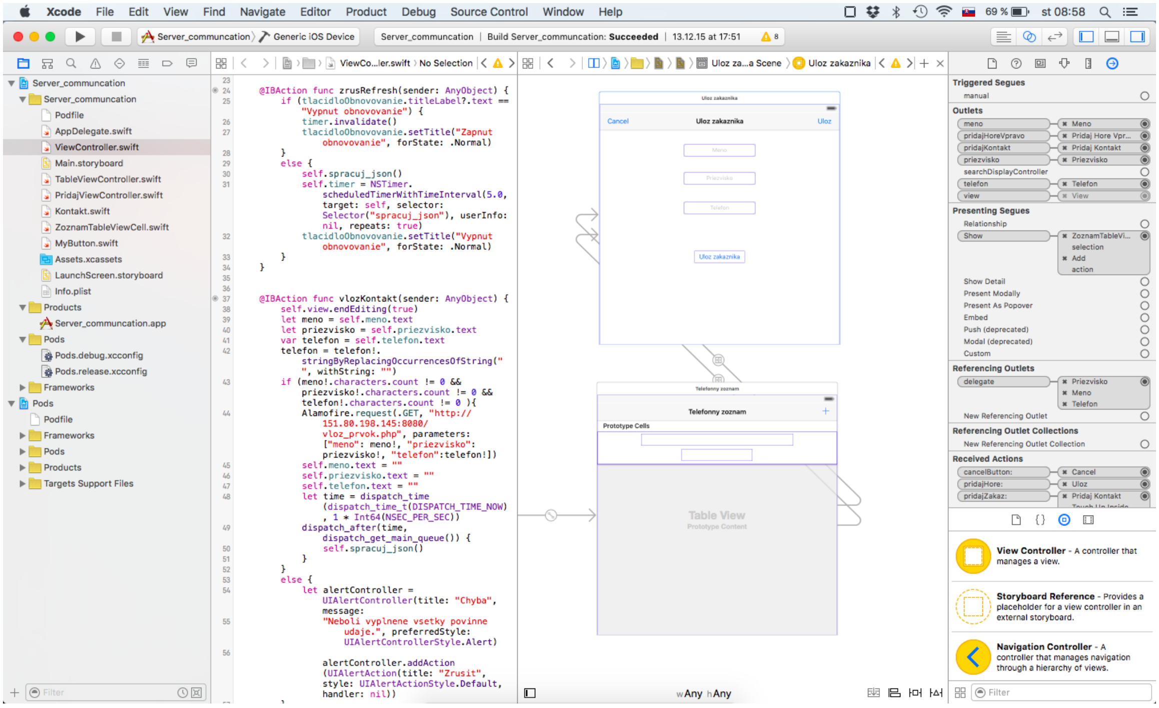
Can't wait to see you and snap some photos of the new pup!

Figure 2.1: Application Connect – Map, chat and group your friends

2.3 Development tools

In order to create application, we need to make decision what tools we want to use and consider all advantages and disadvantages which brings selected solution. It is more and more popular to develop crossplatform application. Crossplatform application means application which is able to run on different operating systems without need code new application. It is no need to know and learn programming language for every operating system – Objective-C (Apple), Java (Google), and .NET (Windows Phone/RT). The main reason for using crossplatform mobile development tools is that there is no need to code application everytime from beginning up to three times for most popular operating systems – iOS, Android and Windows 10 Mobile, but to use only one programming language. One of the most popular tools are Xamarin (.NET), Appcelerator (Javascript), Phone gap (HTML5, CSS, Javascript), Kony (Javascript), Corona (.NET) and Telerik (Javascript). Each tool of the above written has the advantages (stability, native application, easy language, built-in tools) but also disadvantage (difficult to learn, non-standard UI, UI limitations, complicated or limited native support) [3] [4].

Xcode is official application developed by Apple used to develop application for Apple devices. Newest version is Xcode 7 and it is available only for devices which run on OS X operating system. So first thing what is needed to developing application for iOS is to have a Mac. It is also necessary to register a developer account at https://developer.apple.com/ what is for free. In case that developers want to publish application in Apple Store they have to pay subscription 99 $ every year. In 2015 Apple decided that after pay one subscription it is able to develop for a variety of operating systems of their products – iOS 9, watchOS 2 (Apple Watch), OS X El Capitan (Mac), and tvOS (Apple TV). Until 2015 developers had to pay 99 $ for every operating system.



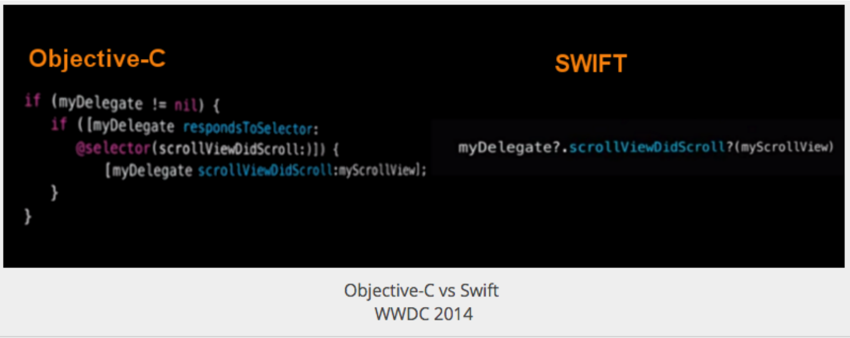
```
    @IBAction func zrusRefresh(sender: AnyObject) {
        if (tlacidloObnovovanie.titleLabel?.text ==
            "Vypnut obnovovanie") {
            timer.invalidate()
            tlacidloObnovovanie.setTitle("Zapnut
                obnovovanie", forState: .Normal)
        }
        else {
            self.spracuj_json()
            self.timer = NSTimer.
                scheduledTimerWithTimeInterval(5.0,
                target: self, selector:
                Selector("spracuj_json"), userInfo:
                nil, repeats: true)
            tlacidloObnovovanie.setTitle("Vypnut
                obnovovanie", forState: .Normal)
        }
    }


    @IBAction func vlozKontakt(sender: AnyObject) {
        self.view.endEditing(true)
        let meno = self.meno.text
        let priezvisko = self.priezvisko.text
        var telefon = self.telefon.text
        telefon = telefon!.
            stringByReplacingOccurrencesOfString("
            ", withString: "")
        if (meno!.characters.count != 0 &&
            priezvisko!.characters.count != 0 &&
            telefon!.characters.count != 0 ){
            Alamofire.request(.GET, "http://
                151.80.198.145:8080/
                vloz_prvok.php", parameters:
                ["meno": meno!, "priezvisko":
                priezvisko!, "telefon":telefon!])
            self.meno.text = ""
            self.priezvisko.text = ""
            self.telefon.text = ""
            let time = dispatch_time
                (dispatch_time_t(DISPATCH_TIME_NOW)
                , 1 * Int64(NSEC_PER_SEC))
            dispatch_after(time,
                dispatch_get_main_queue()) {
                self.spracuj_json()
            }
        }
        else {
            let alertController =
                UIAlertController(title: "Chyba",
                message:
                "Neboli vyplnene vsetky povinne
                    udaje.", preferredStyle:
                    UIAlertControllerStyle.Alert)

                alertController.addAction
                (UIAlertAction(title: "Zrusit",
                style: UIAlertActionStyle.Default,
                handler: nil))
```

Figure 2.2: Sample application in Xcode 7 in Swift programming language

Objective-C is programming language for iOS from the first version of Xcode. As you can notice from the name, it is object oriented language. Basics of Objective-C are from programming language C and system of messages was added from programming language Smaltalk. In 2014 Apple introduced programming language named Swift, which has easier syntax 2.3 compering to his predecessor Objective-C and run of application is also faster. Complex object sort is in Objective-C 2.8 times speeder then in Python and Swift even 3.9 times speeder. This comparison was shown on WWDC14 when Swift was introduced. Swift became Open Source in 2015. It is possible to run develop application in Objective-C and Swift at the same time so some parts can use old Objective-C and newer part can be written in new Swift.



Objective-C

```
if (myDelegate != nil) {
    if ([myDelegate respondsToSelector:
        @selector(scrollViewDidScroll:)]) {
            [myDelegate scrollViewDidScroll:myScrollView];
    }
}
```

SWIFT

```
myDelegate?.scrollViewDidScroll?(myScrollView)
```

Objective-C vs Swift
WWDC 2014

Figure 2.3: A comparison of Objective-C and Swift.

Model-View-Controller is a design pattern what splits application into three independent components – data model (Model), control logic (Controller) and user interface (View). This design pattern is used in Xcode. Data model represents information with which application works. Control logic responds to events and it makes changes to the user interface and data model. User interface (view) shows information from data model into intelligible form for human. The advantage of this attitude is that in order to modification of one the component does not effect other components. The application becomes easily editable, maintainable, reusable and scalable than application does not based on MVC. MVC is a core to right design Cocoa/Cocoa Touch (environment to developing applications for iOS and OS X) applications.

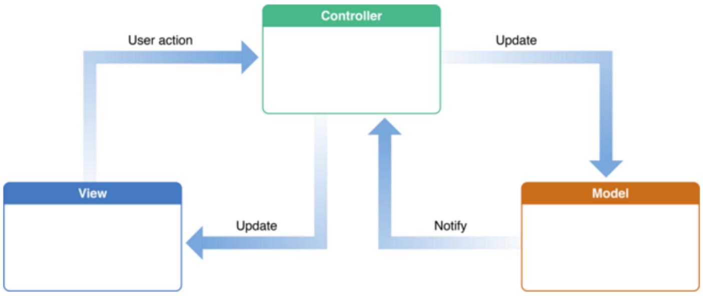


Figure 2.4: Model-View-Controller

2.4 Usability testing

Usability testing is one of the most important part of whole development process. This part is done right after application is ready to use. It is used to evaluate the application, because constructor of application can not objectively take measure application quality. We need appropriate number of real users to test application in order to provide us needed feedback.

Testing should include all aspects of application usage. Users should test all possible features which applications provides. Group of people should be as much similar to target group of application as possible. It would not have any value to test different target group. Tested group should be informed about anonymity of test that will be used for improving quality of software and not to test users. There is a lot of possible ways how to test application and I will introduce a few of them [1].

- Direct observation – writing notes or measure time. This method is easy and cheap but on the other hand, it is annoying from the user view, because he is non-stop observed and it can caused distraction. Therefore, it can lead to unnaturally behavior and we can get fake results. It is important to decide what will be observed in advance to avoid loss of some information.

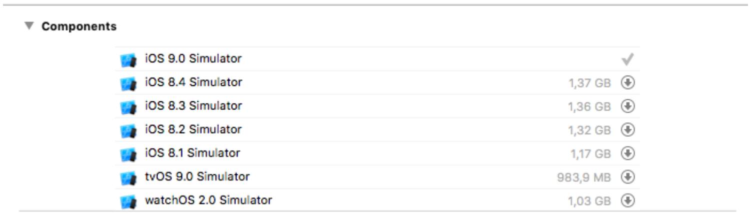
- Indirect observation – usage of device for recording. We can use audio or video recorder and we can combine it with keyboard synchronization. It is necessary to set what and what interval will be measured. This method have similar problem as previous, because user can feel observed by camera recorder. It is common practice to start recording user before test will start. It should help him to get used to it. Another problem is that data processing of this method is time-consuming.

- Think aloud protocol – user verbally describes his observation and thoughts. This method was introduced by Clayton Lewis while he worked at IBM. It is also known as talk-aloud protocol and it is not used only in development but it is used in psychology and in social sciences. Advantage of this protocol is that we can use more information such as voice tone or feelings. On the other hand, user can not keep attention for a long time because this method is more exacting. This is reason why it is common to do this test in 2 people at the same time.

- Retrospective think aloud protocol – it is the same protocol as above with the exception that protocol is done after the task is completed. It can be used in in situations where is needed height level of concentration. We should realize that protocol will differ from think aloud protocol because protocol will contain memories which he would not say in think aloud protocol. This is reason why we should prefer think aloud protocol if it is possible.

- Activity log – user activity is logged long period of time. Tools record mouse movement, pressed buttons or touches on the screen. This method is not a time-consuming process if we use automatized data processing but it is quite expensive. Biggest advantage is that this method do not disturb users.

Testing applications is possible directly in Xcode using built-in simulator. As you can see in the image 2.5, simulator supports several devices (up to four years old devices) so it is easily possible testing application on many devices without need to buy devices for testing purposes and application can be tested on different display resolutions. There is also option to test application on older version of iOS (2.6). Of course application can be tested directly on real device. This application needs to be installed by USB cable. I think this solution is the best because some features such as 3D Touch (it is function of iPhone 6s, where phone react differently according the power of touching on screen) in simulator is not able to test.



▼ Components

iOS 9.0 Simulator		✓
iOS 8.4 Simulator	1,37 GB	⬇
iOS 8.3 Simulator	1,36 GB	⬇
iOS 8.2 Simulator	1,32 GB	⬇
iOS 8.1 Simulator	1,17 GB	⬇
tvOS 9.0 Simulator	983,9 MB	⬇
watchOS 2.0 Simulator	1,03 GB	⬇

Figure 2.5: Ability to download different operating system

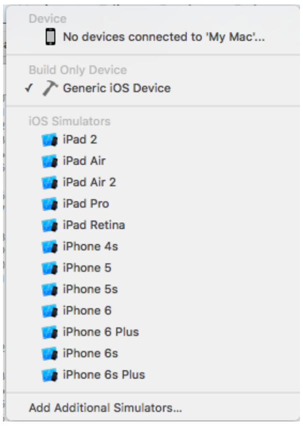


Figure 2.6: Choosing device in simulator

2.5 User interface

User interface (UI) is a communication canal between person and machine. The goal of communication is execute the task by machine which person entered. It is not communication in our natural language – human language because human language is difficult to be processed by machine. Of course, there are attempt to process human language but it is still not perfect. For example in our mobile phone there are assistants such as Siri in iOS, Cortana in Windows Phone or Google now in Android. Nevertheless, we seek the most efficient and most natural way of communication from human perspective. It was not common because in the past, UI has been adapted to the machine. We can make a conclusion that we are trying to create the UI that will be most similar to human-to-human communication.

Norman interaction model [11] created in 1988 became one of the most used general model's of interaction between human and computer. It consists of seven parts illustrating the example. Programmer gets the task to create a program for finance management. Programmer make a decision how program will look like so his vision is becoming goal (first part is establishing the goal). Intention (second part is forming the intention) is to create a windows, menu, tables, graphs and calculator. To achieve this goal is need to define the action sequence (third part is specifying the action sequence) in which will be create parts of program – e.g. first calculator then tables, graphs and at the end menu. He creates (fourth part is executing the action) components using programming language. He watches (fifth part is perceiving the system state) the results of his work, what is created and how it looks like. Then he reacts (sixth part is interpreting the system state) to changes and later this changes are evaluated (seventh part is evaluating the system state with respect to goals and intentions). In case that everything is correct then loop is completed. If it is not correct, loop is going again from the first part.

User interface failure from user perspective:

- misunderstanding of button's meaning – efforts to self-explanatory buttons using usual appearance of buttons.

- mistouch – with the arrival of touch screen phones we need to accurately capture touch (HW) and increase the distance between buttons (SW)

Application should be designed for user to have completely overview about where he is, in which part of application he is at the moment and what information is required. [1] Application user interface should be intuitive so there is no need of reading user guide which actually are not read at all. Application uses common ways of interaction, appearance of buttons and place of elements. Users should have available "rescue button" which returns one step back. This button provides user confidence and comfort while using application. One of key features of application is to provide only those selection options that do not lead to errors such as "can not field", "unknown order" and so on. Using of colors with sense – green color means success and red is a color of failure. These colors are used to highlight most important elements. In case that application does not provide immediate response to given task it is satisfactorily to inform about processing of executed action. In the other case user should get wrong opinion that application does not react or stop working. The UI should be personalized according to the target group. When the target group are seniors UI should seems more conservative than application targeted on the youth.

2.6 Virtual 3D Oulu

Virtual 3D Oulu is a collaborative 3D virtual model of downtown Oulu realized atop the realXtend open source platform for the 3D Internet. The model was constructed using large-scale laser scanning of Oulu's downtown area, augmented by digital photographs of building facades. The platform serves as a collaborative infrastructure where information from physical sensors (such as WiFi users, bus positioning, sensing data from citizens' smartphones) is visualised to enable collaborative viewing and interaction. This work begun in 2014 and is funded by numerous grants from the Academy of Finland, City of Oulu, and by strategic funding from the University of Oulu. This project is research activity at University of Oulu, Faculty of Information Technology and Electrical Engineering, Center for Ubiquitous Computing[1].

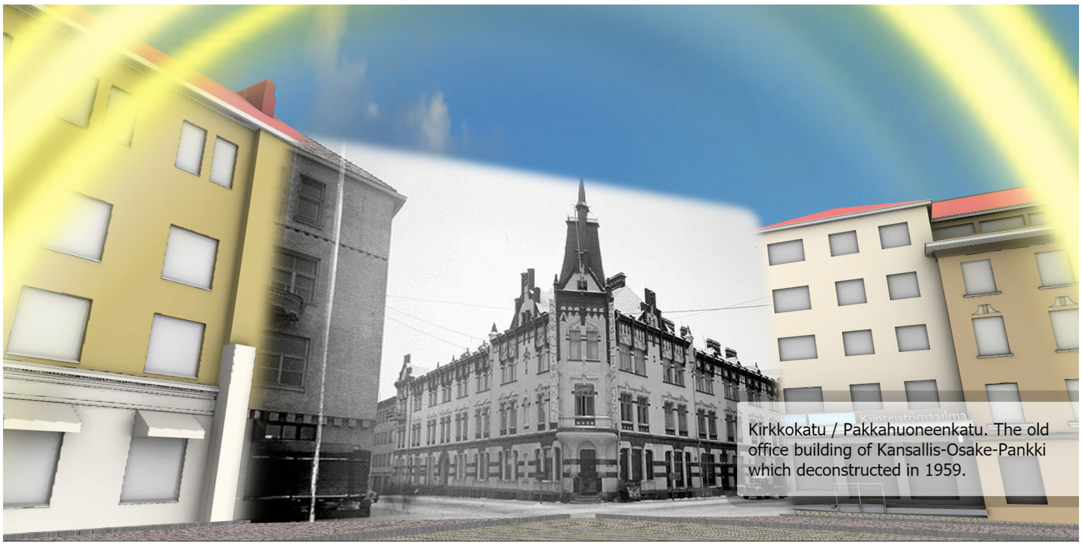
The Virtual continuum is a continuous scale ranging between the completely virtual (virtuality) and the completely real (reality). This concept was introduced by Paul Milgram in [6]. According to him, virtual continuum continuum encompasses all possible variations of real and virtual objects. We can distinguish two extremes – real environment and virtual environment. Everything between these two environments he called mixed reality (MR). Mixed reality consists of augmented reality (AR), where the virtual augments the real (Augmented Reality), and augmented virtuality (AV), where the real augments the virtual. AR refers to all cases in which the display of an otherwise real environment is augmented by means of virtual (computer graphic) objects. The converse case on the virtuality continuum is therefore AV.

Virtual reality (VR) is term which is used and is very popular nowadays, even though it has not got uniform definition. VR is environment where computer perfectly simulate real word [13] . Observer in this environment should be immersed in and able to interact with a completely synthetic world. As I said, VR is very popular topic and because of it a lot of companies are trying offer VR experience – Google Cardboard (Google), Oculus Rift (Facebook) and many others. 8 milion Gamer VR headsets will be sold according the prediction at the Game Developer Conference in San Francisco [14].

University has got 3D virtual model only of downtown Oulu but model is being expanded to other parts of city. It gives opportunity to explore this practically new field. There is

[1] http://ubicomp.oulu.fi/

countless possibilities how virtual reality can be used and it is limited to our minds. For example one of the projects is about getting to know the history of the town. User walks through downtown and he visits portal. User walks through downtown and he visits portal. After he visits portal, historical photography is shown and overlap present look of building in model. Information about photography are displayed in the lower right corner. This application is shown in picture 2.7.



Kirkkokatu / Pakkahuoneenkatu. The old office building of Kansallis-Osake-Pankki which deconstructed in 1959.

Figure 2.7: Example of application in Virtual Oulu

There are also many similar concepts in this virtual worlds area where they are trying integrate real world activities to virtual world but the question is how and what we can integrate. Mapping of real-world activities to virtual world is managed by multiple sensors. In paper [9] they described acoustic activities monitoring system that automatically detects and classifies 17 major activities usually occur at daily life. Another paper [10] uses acoustic monitoring system to life-logging. According to scenario, their motivation was to create system which sends every day email message to sons of sick people reporting their daily life-sketch.

Chapter 3

Geolocation and messaging on map

My goal is to design and create mobile application which uses geolocation and where users can interact between them in virtual city. I invented application which send some information from real world to virtual world and then get some information from virtual word back to real world. Many application only send information to virtual world, but I consider to very important to get some feedback back to reality. I decided to create web application, too. In this chapter are described steps which led me to create application. I introduce designed application and then I custom it according users needs.

3.1 Mobile application design

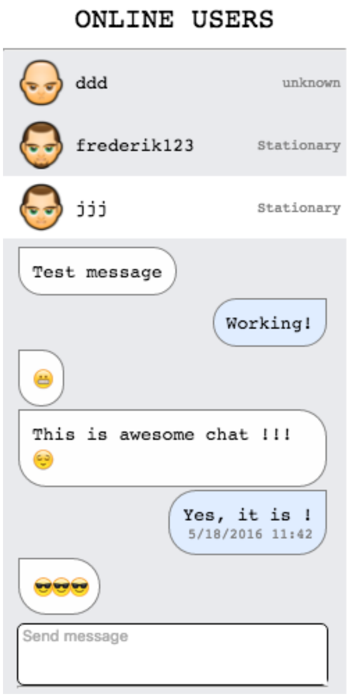
As is described in 2.1, mobile phone became our daily part of our lives. Mobile social network is becoming still more popular. Same tendency we can notice in web-based social networking. People have a need to communicate so I invented application, which sends user's location and show it on map. People see each other on map and after click on one of the icons which represents some user, they can write them message directly. User do not have to find people only on map but he is able to show all online people and these people are shown according distance from current location. After user choose one of the users, chat with previous conversation will be shown if exists. Because people prefer to contact other people when they have some common characteristics or hobby I added feature, where user can filter visibility of other users according their activity. Triaxial accelerometer is becoming a standard in smartphones and paper [12] describes that we are able to recognize activity with very high accuracy – f.e. standing with 100 % accuracy and walking with 97 % accuracy. They achieved best results combining metaclassifiers using Plurality Voting. They also found that energy is the least significant attribute and energy is huge topic in mobile phones nowadays. For example, when I am cycling, I want to see only cyclists and not somebody who is walking right now. The concept of bridging real world activity to virtual world is discussed in [7]. Authors of this article proposed this feature to most used virtual world simulator called Second Life. Users have their unique nicknames and avatars which represents their person. Avatar can be changed any time in setting of account. One of the most important features is center to current location. User can easily lost where he is on map so center to current location can be safe button for him.

3.2 Web application design

In order to offer this application to more people I decided design web application, too. The advantage of web application is that there is no need to install application and application can be run from any operating system. Designed application offers 3D graphics where I use 3D virtual model of downtown Oulu because I want to give feeling of virtual reality to users. Because this model is too small for now, application uses less detailed models from internet. User are able to see other people and contact them directly through personal messages. Also, user is able to walk through city to meet new people and explore the city. User is able to control his walking human with classic game's controllers „WSAD". There is also possible „running" mode with pressed shift button.

3.3 Chat design

One of the most important part in mobile and web application is chat because user probably spends most of his time by chatting. Therefore, I place importance on exquisite design of chat. Chat should be well arranged, attractive in design, bring something new and last but not least, chat have to easy-to-use even for users who use it for the first time. With the above-mentioned principles I designed chat for web application and also for mobile application as you can see on pictures 3.1 3.2. I used vertical chat in web application so user can easily scroll between online users. After clicking on the user, messages with user will be shown right on the chat bar. Date and time when message was sent is shown on mouseover. If user again click on username then messages disappear. When user receives message, sender starts blinking and audio notification is played. Thanks to these two features it is not possible that user will not notice new messages from other users. User is also able to open chat with somebody after clicking on user who is on map. Sending message is possible with pressing enter button. User only writes text which he wants to send into input field and then he presses enter. Sent messages are shown on the right side and received messages are shown on the left side. This way is used by all application which I know and usage of different way would be meaningless. Chat in mobile device is very similar to original build-in messaging application called Messages. It is because Apple users are familiar with this solution and other solutions could distract them. Same as in web application, sent messages are on the right side and received are on the left side. User can write message by three ways. First and most common way is clicking on user who is on map. Next way how to write message is from the navigation panel and then press „Messages" button what opens list of online users sorted by distance from actual user's location and after clicking on one of users, chat will appear. Last way ho to write a message is by receiving notification. If user get message, notification is appeared on the top of display. After clicking on notification, chat is opened.



ONLINE USERS

ddd unknown

frederik123 Stationary

jjj Stationary

Test message

Working!

😬

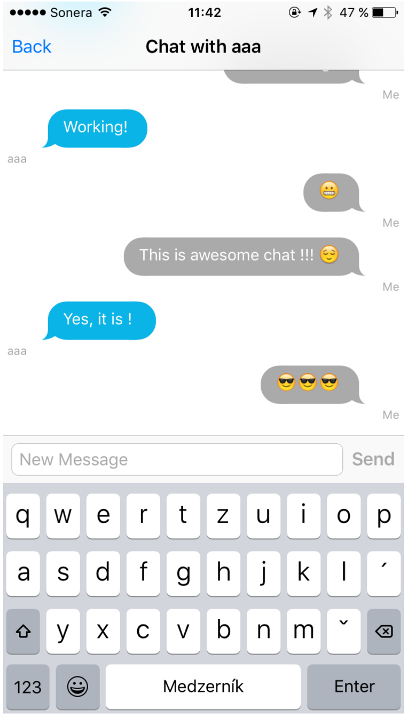
This is awesome chat !!!
😌

Yes, it is !
5/18/2016 11:42

😎😎😎

Send message

Figure 3.1: Web application chat



●●●●● Sonera 11:42 47 %

Back Chat with aaa

Me

Working!

aaa

😬

Me

This is awesome chat !!! 😌

Me

Yes, it is !

aaa

😎😎😎

Me

New Message Send

Figure 3.2: Mobile application chat

3.4 Mockups

This section contains two designed graphical user interfaces (GUI) mockups of landing page in web application and two designed graphical user interfaces mockups of landing page in mobile application. Practically, landing page for users in web application is only page they can see so it is extremely important to design as good graphical user interface as possible. According to my research which I described in 2.5 I designed two different interfaces.

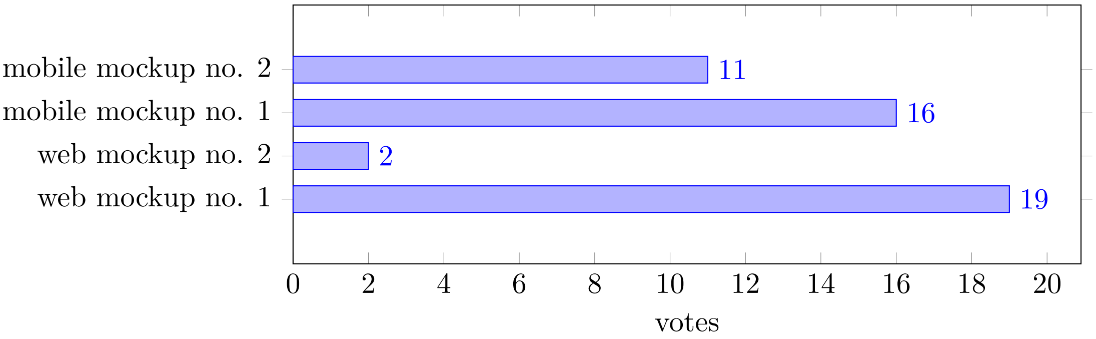


Figure 3.3: Application mockup testing

I proceeded to design a number of mockups and those mockups consulted with potential users. I made some iteration and then I chose two final mockups. First mockup of web application GUI corresponds with picture 3.6 and second with 3.7. Picture 3.3 shows that mockup number 2 prefer 90% of questioned from total 21 people. This result was big surprise for me, because design number 1 is similar to worldwide used Facebook. It can

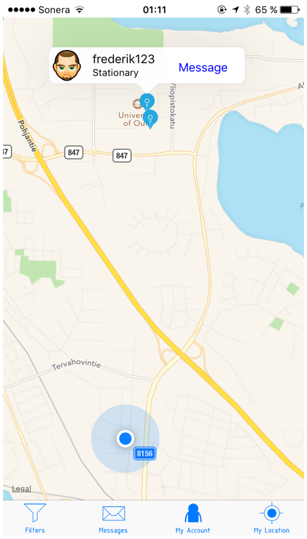


Figure 3.4: Mobile app mockup no. 1

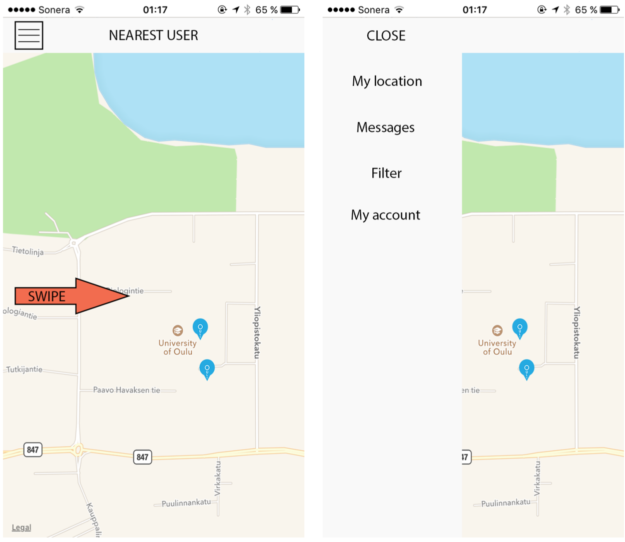


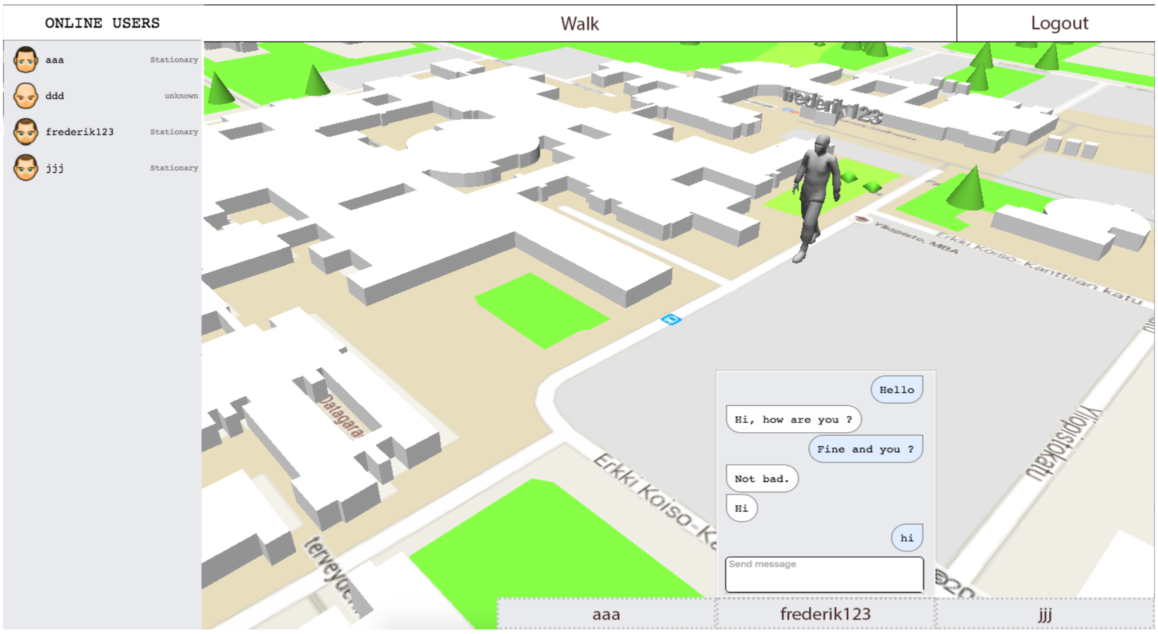
Figure 3.5: Mobile app mockup no. 2

be caused by fact, that design number 1 was shown firstly and afterwards design number 2 was shown. I think that people give preference to second solution over first solution because they psychologically think that second solution is something better and updated. Therefore, I decided to uniformly change order of prototypes in next testing to avoid this possible loss of value.

First mockup of mobile application GUI corresponds with picture 3.4 and second with 3.5. Picture 3.3 shows that mockup number 1 prefer 60% of questioned from total 27 people. This result was not convincing. In such a small number of votes it would not mind to choose second solution but I chose to implement first mockup.

Difference between mobile first mockup and second mockup is in menu layout. Position of menu in first mockup is at the bottom of screen. This menu is visible all the time in map view. It encourages users to use other functions like filter or messages. Second mockup uses different approach. User has to swipe from left side of screen to right side or click upper left button in order to show menu. Main advantage of this attitude is that menu does not disturb users while working with map and we can also show name of nearest user at the top of the screen. Disadvantage is that user has to touch display two times in order to show filter or messages.

As you can see from mockups, I chose two different approaches to map view in native mobile application and web browser application. There are several reasons why I decided to do it. In mobile application I use icon to show user on 2D map. After click on icon, application shows user avatar, activity and message button. This solution is very common and user friendly. Mobile phones have a common screen size up to 5„ so it is important to show all things as small as possible but still readable. Because personal computers have incomparable larger screen size and often we do not care about battery life, web application use 3D map visualization and 3D large avatar. It offers to user a sense of the real world. User can see activity of other users immediately according movements of their avatars. Also, these activity is shown on the chat panel. The question was, where to put chat. First mockup version has got list of online users on left side and opened chats at the bottom of screen while second mockup version has got both list of online users and opened chat on right side.



ONLINE USERS

aaa	Stationary
ddd	unknown
frederik123	Stationary
jjj	Stationary

Hello

Hi, how are you ?

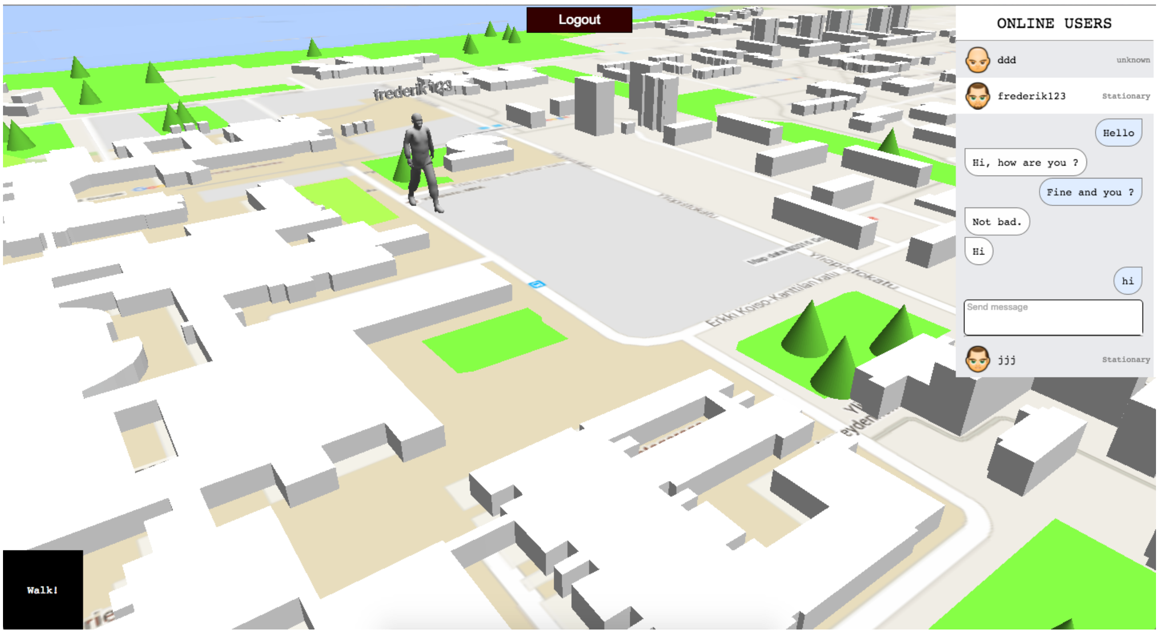
Fine and you ?

Not bad.

Hi

hi

Figure 3.6: Web application mockup no. 1



ONLINE USERS

ddd	unknown
frederik123	Stationary

Hello

Hi, how are you ?

Fine and you ?

Not bad.

Hi

hi

jjj	Stationary

Figure 3.7: Web application mockup no. 2

Chapter 4

Virtual Oulu

This chapter deals with main and interesting parts of created application called Virtual Oulu. First, I introduce getting and sending location and activity tracking. Then there are few words about what and how data are stored. Next, there is described log in and sign in process. Then I explain how is map implemented and how to communicate with database. At the end, there is explanation how map is implemented and how to get maps from internet.

4.1 Localization and activity

Applications in mobile phones use localization services for various purposes depending on the type of application – navigation, mobile phone localization, running/bicycling analyses and social networks. We need to use localization services for showing users on map. Getting localization from GPS chip uses class from Core Location framework. This framework provides services for monitoring actual localization of devices.

To use framework we need to link `CoreLocation.framework`.
Request authorization to use location services → include key `NSLocationAlwaysUsageDescription` and `NSLocationWhenInUseUsageDescription` – type `String` and value `This application requires location.` into file `Info.plist`.
There are two services which provide getting position:
1. standard location service – configurable according needs
2. significant-change location service – log only huge differences between two locations
I used standard location service. Here are steps to getting the position.

```
import CoreLocation
var manager: CLLocationManager = CLLocationManager();
// create an instance of the CLLocationManager class and store a strong
        reference to it
manager.delegate = self // set delegate
manager.desiredAccuracy = 50 // set accuracy in meters
manager.requestWhenInUseAuthorization() // requests permission to use
        location services
manager.startUpdatingLocation() // start the generation of updates that
        report the user's current location
```

In order to receive events by delegating is needed implement method `locationManager: didUpdateLocations:` according to `CLLocationManagerDelegate` protocol. Then receiving of events looks like this:

```
func locationManager(manager: CLLocationManager, didUpdateLocations
       locations: [CLLocation]) {
        let user_location: CLLocation = locations[0] // save location
        let latitude: CLLocationDegrees = user_location.coordinate.
               latitude // save latitude
        let longitude: CLLocationDegrees = user_location.coordinate.
               longitude // save longitude
    }
```

There also exists important function `distanceFromLocation(_:)` which measure distance between two locations. This function is used to order users according to current user's location.

User can disable getting location so it is good practice check ability to get location. It is checked by class method `locationServicesEnabled` of class `CLLocationManager`.

When I have a user's last location, I compare it with last location and if it differs a lot, `send_location` variable is set to true, which mean that localization will be sent maximum in 5 second. I do not send small differences between old and current location to save battery life and internet data usage. Location is sent every 5 second if `send_location` variable is true.

Application offers showing user's activity and filter according user's preferences. User's motion activity provides `CMMotionActivityManager` class. The `CMMotionActivityManager` class provides access to the motion data stored by a device. Motion data reflects whether the user is walking, running, cycling in a vehicle, or stationary for periods of time. First I need to import CoreMotion and create an instance of the `CMMotionActivityManager` class by calling method `CMMotionActivityManager()`. Then I check availability by calling `isActivityAvailable()` and then method `startActivityUpdatesToQueue(_:withHandler:)` initiates the tracking of motion data asynchronously. If received activity is different from last activity then I immediately send new activity to database.

4.2 Database

For programming cloud back-end we can use existing SDK. Practically, I had to choose between SQL and NoSQL database. I decided to use NoSQL database because application will not require complexity of SQL databases. In general, advantages of NoSQL include being able to handle [8]:

- frequent code pushes
- easy scalability
- large volumes of structured, semi-structured and unstructured data

Firebase is a cloud application platform that provides an SDK for creating back-end without any coding. It support three platforms – Android, iOS and Javascript. We will need database for storing locations of users, authentication, messages and details about users such as nickname or avatar. Listing 4.1 shows structure how the data are saved.

Database contains 3 basics nodes – locations, messages, users. Locations has children nodes – users, which contains activity (string), country (string), course (integer), image (string), information about last activity (integer), information about user’s last location – latitude (double) and longitude (double), speed (double) and if user is online, “online„ value exists and is set to “YES„. Node has children nodes – users, which contains another nodes, with information about received messages. Message contains two values. First value is time, when message was sent and text of message. F.e. in path messages->user1->user2 are all messages, which user2 sent to user1. There is also node with name “new_messages„ which symbolize that user have new message from another user, whose username is in this node so if path messages->user1->new_messages->user2 exists, it means that user1 have unread message from user2. Node users have children nodes which contains information about user. We can find here email, password and username. I use `.indexOn` rule what tells the Firebase servers to index specific keys in your data to improve the performance of queries. This rule is used for user’s country, because application shows only people in the same country what means we download only users in the same country.

```
■ locations
    □ user1
        • activity:  "Walking"
        • country: "FI"
        • course: −1
        • image: "B01"
        • lastOnline: 1463475623310
        • latitude: 65.05811301235664
        • longitude: 25.46853469697695
        • speed: 5
        • online: "YES"
    ...
■ messages
    □ user1
        • user2
            ○ messages_key_1
                – date: "1460627816426"
                – text: "This␣is␣message"
            ○ ...
        • ...
        • new_messages
            ○ user2: "1"
            ○ ...
■ users
    □ users_key_1
        • email: "user1@email.com"
        • provider: "password"
        • username: "user1"
    ...
```

Listing 4.1: Structure of database

Security is an integral part of the database. Firebase provides a flexible, expression-based rules language with JavaScript-like syntax to easily define how data should be structured and when data can be read from and written to. Combined with login service which allows for easy authentication, we can define who has access to what data and keep all of user's personal information secure. For example, how to ensure that user's location can change only authorization person ? This situation illustrates picture 4.2. We need to set `.write` permission. Variable `$name` is username which belongs to particular location of the user. If this variable is equal to actually logged in user's username, user can write or update data in this location.

```
"locations":{
  ".read": true,
  ".indexOn": "country",
  "$name":{
    ".write": "root.child('users').child(auth.uid).child('username
          ').val()␣==␣$name"
  }
},
```

Listing 4.2: Security rules for locations

4.3 Registration and login

Every user of mobile application have to be signed in because we need to know the identity of a user. Landing page for new users in mobile application is login page. User have to enter his unique username (nickname), unique email, password and avatar. Users in web application do not have to login immediately. They are asked to login or register in order to write a message to other users. For logging in and registration is used Firebase authentication system. When a user authenticates to an application, three things happen:

- Information about the user or error is returned in callbacks on the user's device.
- The user information returned contains a UID (a unique ID), which is guaranteed to be distinct across all providers, and to never change for a specific authenticated user.
- The value of the auth variable in your app's Security and Firebase Rules becomes defined. For authenticated users it is an object containing the user's unique (auth.uid). This allows me to securely control data access on a per-user basis.

Log in and registration functionality is implemented in RegisterViewController.swift and LoginViewController.swift. While mobile application is waiting for callback, I used SVProgressHUD[1] – a clean and lightweight progress HUD for iOS to inform user that request is being processed. If error occurred, application call function `signupError(title: String, message: String)` which shows alert with explanation of the error. When user is registering, first we check is if user filled all required data and check if username is unique, then application call `createUser` method and then `authUser` method which is also used in logging in process but here we also call functions `createNewAccount()` and `createFirstLocation()` to store data to database. Function `createFirstLocation()` create user's location data and set its to impossible value so we know that this user is registered but he have not already sent valid location data. I used `NSUserDefaults` class

[1] https://github.com/SVProgressHUD/SVProgressHUD

to save UID, username of user and for filter functionality. NSUserDefaults saves data on principle key and value. After successful registration (or login) application perform segue with identifier “NewUserLoggedIn„ (or “CurrentlyLoggedIn„) which shows main storyboard with map.

In web application process of login and register is similar to mobile application. First in function registerFirebase I check if username exists, then I create user by calling method createUser to create account and login by calling method authWithPassword. Finally, application sends location and user data to database by calling function set() which saves new data to the specified database reference. Then application call variable $.result which writes corresponding message. During login process in loginFirebase function is called another function authWithPassword() and then downloaded and saved username from database.

4.4 Map

In mobile application development for iOS is a lot of possibilities how to show map. Many of map APIs offer offline maps but these solutions are paid, f.e. solutions Mapbox[2] or Skobbler[3]. It is reason why I chose official Apple MapKit API. The Map Kit framework provides an interface for embedding maps directly into views. This framework also provides support for many other things. I used annotating the map and performing reverse-geocoding lookups to determine placemark information for a given map coordinate.

First, we need to import Map Kit framework. Whole Map Kit functionality is implemented in MapViewController class. This class conforms to MKMapViewDelegate to request annotation, which we will need to be able to show information about user and ability to send message from map. Then I added Map Kit View from object library to MapViewController View and created @IBOutlet map to access property from code in function viewDidLoad(). Next, I set delegate of map to self. Now we have all required things to show map, but map is not centered to any location by default. I use setRegion(_ region: MKCoordinateRegion, animated animated: Bool) function to change the currently visible region and optionally animates the change. First parameter is type of MKCoordinateRegionMake and to create variable of type MKCoordinateRegionMake I used funcion MKCoordinateRegionMake(_ centerCoordinate: CLLocationCoordinate2D, _ span: MKCoordinateSpan). First parameter is type of CLLocationCoordinate2D so we can use CLLocationCoordinate2DMake(_ latitude: CLLocationDegrees, _ longitude: CLLocationDegrees) which returns MKCoordinateRegionMake. I used aproximatelly center of city – latitude 65.013159 and longitude 25.496600. To pass second parameter of MCCoordinateRegionMake I used function MKCoordinateSpanMake(_ latitudeDelta: CLLocationDegrees, _ longitudeDelta: CLLocationDegrees).

LatitudeDelta (LongitudeDelta) is The amount of north-to-south (east-to-west) distance to use for the span and one degree of longitude spans a distance of approximately 111 kilometers at the equator but shrinks to 0 kilometers at the poles. I used value equals 0.1 to cover whole city.

Now when map is loaded, new timer is created and every 5 second is function timerAction called. This function checks internet connection and starts notification system of incoming messages. After that, function reverseGeocodeLocation() is called to get ISO country

[2] https://www.mapbox.com

[3] https://www.skobbler.com

code of user's position, because I decided that the best solution is if only users from the same country are downloaded. It is pointless to download users from all over the world, because it would slow down application and consume internet data. If user's last country differs from current country, I delete all users from previous country and sets observation to nodes where country equals current ISO country code. Its timer function is also sening user's longitude, latitude, speed, course and country.

When user location is downloaded, application call function `showLocation (snapshot: FDataSnapshot)`. First, function checks if downloaded user is user itself. Next, all data are saved to variables and I save information for chatting what is described in 4.2. Then annotation subtitle is created. I created class named `MapPin` to create annotation. This class inherits from `NSObject` and `MKAnnotation` classes. This class is implemented in `MapPin.swift` and contains only one method for initialization of variables. Subsequently, I load variable from phone memory to check if user want to show downloaded user with his activity. This variable contains associative array `[String:Bool]` what represents activity and if it should be shown (f.e. "Running„ equals false). If user should be shown, application check if user is already downloaded (only changed location) and according this information, application delete old annotation `removeAnnotation()` and show annotation by calling function `addAnnotation()` or only show annotation without deleting it. Last thing what application checks in this function is that if user already selected annotation as picture 4.1 illustrates, annotation is selected again by calling function `selectAnnotation()` what cause blink of selected annotation.

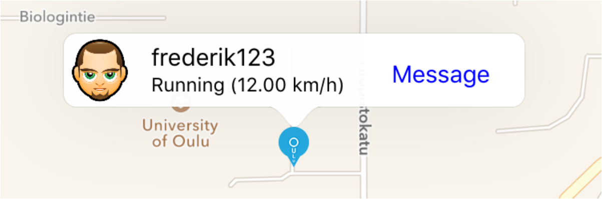


Figure 4.1: Selected annotation

In web browser application is situation quite differenct. First of all, application needs map data to show it. I decided to use Google maps for 2D map, OpenStreet Maps for 3D models and detailed 3D virtual model of downtown Oulu. Whole functionality of maps is implemented in Javascript using three.js library.

Getting data and showing map from Google maps is implemented in `loadMap.js` file. There is implemented `function mapLoader()` and another functions within this function because we do not want to access these functions outside `mapLoader` function. We have three different quality of maps. The first function is called `updateMap()` which gets the center point of the tile where camera is looking at each zoom level then load objects by calling functions `getBuildingsAndDraw()`, `getForestsAndDraw()`, `getPlantsAndDraw()` and checks whether the tiles near the camera exist by calling `checkAndRemove` and `checkAndCreate()` function at the end. Last written function contains nested loop to create visible tiles by calling `createTile()` function which create tile if does not exist where it called important function `createMapPlane()` because this function download data from Google API and add it to the scene.

Getting data from OpenStreetMap is implemented in `getData.js` file. I use OpenStreetMap API[4]. There is special query language described here. First, I needed to choose

[4] http://wiki.openstreetmap.org/wiki/Overpass_API

between four endpoints. Unfortunately, one of these endpoints is not stable but after long-term testing I chose Overpass API on openstreetmap.fr. Basically, box-query lets us download all nodes, ways or relations from the given bounding box. Overpass query language (QL) form has the implicit order (s, w, n, e). "S„ is the southern limit in decimal degrees, "w„ is the western limit in decimal degrees, "n„ is the northern limit in decimal degrees and "e„ the eastern limit in decimal degrees. A node represents a specific point on the earth's surface defined by its latitude and longitude. A way is an ordered list of between 2 and 2,000 nodes that define a polyline (connected series of line segments). A relation is a multi-purpose data structure that documents a relationship between two or more data elements (nodes, ways, and/or other relations). I used building, landuse and natural tags to show buildings, forests and some green parts (scrub, park...).[5] When query is constructed, application call `encodeURIComponent()` function to encode special characters and the following characters: , / ? : @ & = + $ #. Then application call jQuery functio `$.getJSON(url [, data ] [, success ])` what load JSON-encoded data from the server using a GET HTTP request. This function is called in loop, because endpoint is not stable so it often happens that data is not loaded on the first try. Then I call draw functions which add interprets data as models and add them to scene.

Visualization is implemented in draw.js file. First we need to set material and geometry for models. I explain functionality of each function. Function `worldObject()` can add or remove mesh to scene. It uses function `convertScenetoGPS()` to convert scene coordinates to GPS coordinates. There are also drawing functions for buildings, forests and plants which contain initialization. Last but not least this functions contain two important functions `drawWays()` and `drawRelations()` which represent data gained from Openstreet API as models. These models are pushed to `worldObject` variable to add them to scene. Function `getDistance()` computes the distance between the two GPS coordinates in kilometers. This function ensures that detailed 3D virtual model of downtown Oulu does not collide with models from OpenStreetMap, because every time before mesh is added to scene, application checks if distance from middle point of downtown model is at least 220 meters away from current mesh. Function `fillWithObjects()` is called if model is forest or plant and this function randomly fills these object. Ways in relations are not sometimes ordered so I created function `orderNodes()` which returns ordered ways.

4.5 Chat

In mobile application are three ways how to write a message. First way how to do that is from map annotation what is described in 4.4.

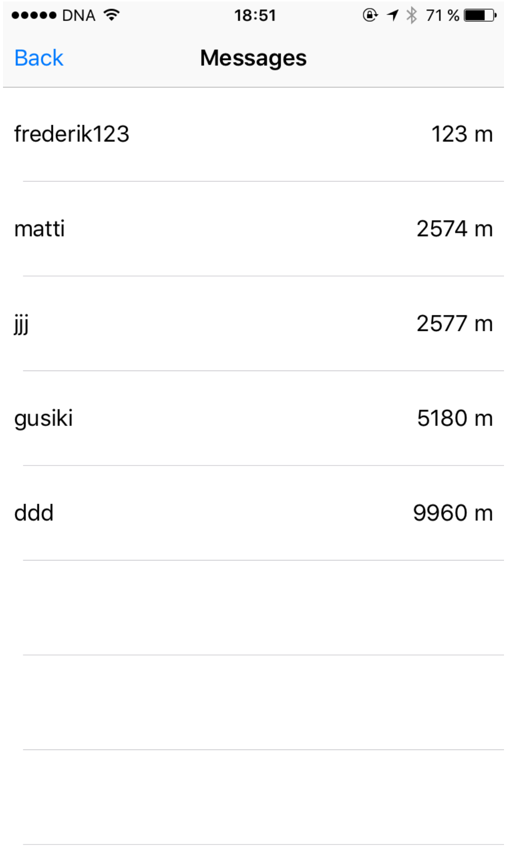
Second way how to write a message is from list of online users. This functionality is implemented in `MessageListViewController.swift`. I created `TableView` so class `MessageListViewController` has to adopt two protocols – `UITableViewDataSource` and `UITableViewDelegate`. Basically, I load data for every row from `UserDistance` array of type `Distance`. Class `Distance` has 3 variables – distance from actual position, longitude and latitude. There is also one method `updateDistance()` except initialization method. This method is called every 5 second to update distances from current location and then I use this `UserDistance` array and I sort his keys according the distance in ascending order. Then I use this sorted keys to load the data in correct order.

Last thing how to write message is clicking on incoming notification. I used LNRSim-

[5] http://wiki.openstreetmap.org/wiki/Elements

pleNotifications library for creating in-app notifications. First, I had to import LNRSimpleNotifications library. Then I created variable by calling `LNRNotificationManager()` and in `viewDiDLoad()` function I set notification panel appearance and sound. At the end, `new_message` variable from database is observed. If new message come, this variable is set, `methodThatTriggersNotification()` is called and variable is removed. Function `methodThatTriggersNotification()` set title and body of notification and perform segue if notification is clicked.

I used to create chat as much similar to original iOS Messages application as possible. Whole chat is implemented in `ChatViewController.swift` file. There is function `downloadMessages` which downloads last 20 received and 20 sent messages from database and subsequently messages are sorted by calling build-in `sort()` function according date in ascending order. It means that last 20 messages can be shown but not more if we want to be sure that messages are in right order. `Setup()` function sets title and names of users. `didPressSendButton` function sends text and time stamp to database and `new_message` variable is set in order to notify other user because he is observing this variable in order be notified when he receive new message.



frederik123	123 m
matti	2574 m
jjj	2577 m
gusiki	5180 m
ddd	9960 m

Figure 4.2: List of online users for chatting

In web application, chat funcionality is implemented in index.html file. I used Firebase SDK and jQuery library. Chat is implemented in HTML, CSS and Javascript language. All online users are shown on right panel. This panel is also visible to unregistered users but when somebody wants to send a message, he is asked to login or register.

Every user have unique nickname so I could use nickname as "id„ of `<div>`. There is `showchat()` function triggered on click. This function have one parameter – nickname. This function add new `<div>` where messages will be displayed, asynchronously download sent and received messages, add messages to array called `messagesArray`, call function `showMessageWith()` and scroll to last message. Function `showMessageWith()` deletes all shown messages with user, sort messages according time stamp and show messages again. Every message has `onmouseover` attribute which fires when the mouse pointer moves over an element. Fired `showTime()` function uses Date object to parse time stamp and shows it in a human readable string.

Place for sending message is implemented as text area with `onkeypress` attribute which fires `sendMessage()` function. This function checks if enter was pressed and then push new message to database and clear textarea.

Chapter 5

Testing

I tried to design and implement as good solution as possible which corresponds mainly with section 2.5. Firstly, I design how the testing will look like. Then, this chapter evaluate quality and user-friendliness of final solution from user perspective.

5.1 Test design

I focused on review whole system. Every functionality which applications offer should be tested, because I tried to design application where should not be any unused function. I used two testing because of two reasons. First reason was to increase number of tester. Second reason was to get different user's opinions because every type of testing is special so we get different type of results. I designed test for in-person testing and remote test. I also have to mention continuous testing with my friends mainly from University of Oulu, because in-person test is more suitable while final product is not done.

During in-person test we can get detailed reaction – emotions, feelings, mimics of tester, we can track task duration or first expression. Biggest disadvantage of this method is that user can feel spied. Questions and tasks in questionnaire are designed to be understandable without requirement of any additional information. In-person usability test is attached in appendix A.

On the other hand, remote test allows us to conduct user research with participants in their natural environment so he do not feel spied as in personal testing. User have more time to send feedback so he can think about improving software, too. User send completed questionnaire via the internet. Remote usability test is attached in appendix B This test was send by email or Facebook chat to my friends who were willing to test my application.

5.2 Test evaluation

I tried to test as many people as possible. Finally, 25 people was tested, where 15 people was tested web application remotely and 10 people participate in-person web and mobile application testing. Most people who tested applications is under 30 years old. More than half testers do not have any problem with testing application because application works as they expected. The last question in the questionnaire was: "Would you recommend this application to your friends and why ?„. The answer to this question was in 85 % cases positive and most frequently reason was that they have not seen such a concept until now. Tester wrote his opinion on each task with emphasis on simplicity and clarity of tasks execution. I created graph which represents difference between web and mobile application in one certain test. It tested ability to work with map. We can see huge difference – users in web application had much more problems with map controlling than users in mobile application. This graph is shown on picture 5.1. Other results were quite similar both in web and mobile application and mostly positive.

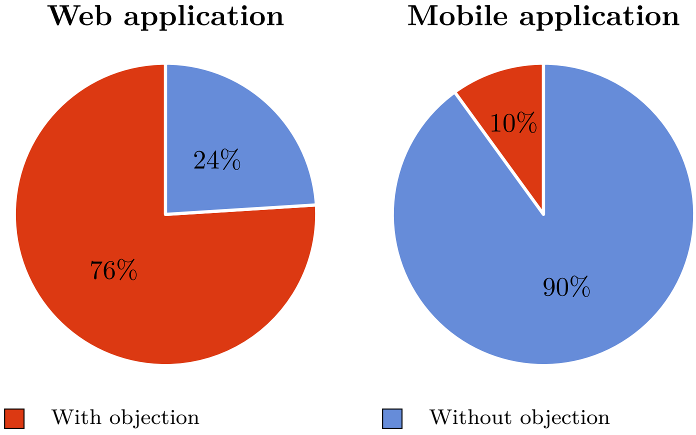


Figure 5.1: Usability testing – test number 3

Below I describe occurred objections and suggestions for improvements in specific points in the test (from test 1 to test 5).

- Log in and registration process – divide login and register page (web), possibility to create own avatar
- Writing a message – if distance to user is more than 999 meters then write distance in kilometers (mobile), ability to check time of message (mobile), ability to show previous messages, show user avatar (mobile), set automatically focus on text filed (web), sometimes it does not scroll to last message (mobile)
- Working with map – it takes time to get used to control (web), add user manual (web) because it is new type of control
- Incoming message notification – ability to disable notifications
- Filter functionality and edit avatar (mobile) / Walking in city (web) – add own avatar (mobile), signalization of activated filter (mobile), support control with arrows (web)

Some features was considered as missing functionality – f.e. unavailability to load more messages than 20. On the other hand, I had to add user manual to webpage because many people had same problem with controlling. It is probably caused by fact, that there is not so many 3D map applications and people are not used to work in that environment.

Chapter 6

Conclusion

The project deals with usage geolocation, 3D graphics and interaction in mobile and web applications. I managed to design and implement mobile and web application, where user see other users on map and he is able to contact other users directly from map. Users are represented by their avatars. I focused on intuitive user interface and usage of modern elements during the design, realization and testing process.

Firstly, I did a research in creation of modern mobile applications, usability testing and concept of Virtual Oulu. Study of these 3 parts was an inseparable part of making this thesis. Firstly, I had to design what application will do and then I created mockups which were updated according user's needs and opinions. Then I chose tools which I used to implement designed applications. I used new programming language – Swift which is used to programming for iOS operation system. To extend Virtual Oulu project I also created version for web browser, which is able to run this application almost to any operating software with Javascript and WebGL support. In the penultimate chapter I described interesting parts of my solution. In the end I explained process and evaluation of usability testing.

As was mentioned in previous chapter, 85 testers would recommend created application. It gives me motivation to continue in this interesting project and augmented virtuality at all. In the future, I would fix issues and add features which were described in evaluation testing. Then I would like to add support to other mobile platforms and offer new features such as arrow with distance to nearest user or usability with glasses for virtual reality to gain better user experience.

Bibliography

[1] Addison-Wesley: *Preece J. et al. Human-Computer Interaction*. Wokingham, UK. 1995. ISBN 0-201-62769-8.

[2] Apple: iPhone SE. http://www.apple.com/iphone-se/. [Online; cited 26.5.2016].

[3] Bluetube: CROSS PLATFORM MOBILE DEVELOPMENT TOOLS COMPARISON . http://bluetubeinc.com/blog/2014/5/cross-platform-mobile-development-tools-comparison. May 2014. [Online; cited 26th May 2016].

[4] Heitkötter, H.; Hanschke, S.; Majchrzak, T. A.: Comparing Cross-platform Development Approaches for Mobile Applications. In *WEBIST 2012 - Proceedings of the 8th International Conference on Web Information Systems and Technologies, Porto, Portugal, 18 - 21 April, 2012*. 2012. pp. 299–311.

[5] Liu, C.: Smartphone users. *Worldwide internet and mobile users: emarketer's updated estimates for 2015*. August 2015.

[6] Milgram, P.; Kishino, F.: A taxonomy of mixed reality visual displays. *IEICE TRANSACTIONS on Information and Systems*. vol. 77, no. 12. 1994: pp. 1321–1329.

[7] Mirco Musolesi, N. D. L. S. B. E. T. C., Emiliano Miluzzo; Campbell, A. T.: The Second Life of a Sensor Integrating Real-world Experience in Virtual Worlds using Mobile Phones. In *In Proc. of the 5th Workshop on Embedded Sensor Networks*. 2008.

[8] mongoDB: Advantages Of NoSQL. https://www.mongodb.com/scale/advantages-of-nosql. [Online; 26th May 2016].

[9] Mostafa Al Masum Shaikh, K. H. I. M., Helmut Prendinger: Easy living in the virtual world: A noble approach to integrate real world activities to virtual worlds. In *Proceedings - 2009 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT 2009*. 2008. pp. 466–473.

[10] Mostafa M. A. Shaikh, M. K. I. M.; Hirose, K.: Automatic Life-Logging: a novel approach to sense real-world activities by environmental sound cues and common sense. In *Ubiquitous Computing and Communication Journal*, vol. 6: Issue 2. Apr 2011. pp. 797–806.

[11] Norman, D. A.: Affordance, Conventions, and Design. *interactions*. vol. 6, no. 3. May 1999: pp. 38–43. ISSN 1072-5520.

[12] Ravi, N.; Dandekar, N.; Mysore, P.; et al.: Activity Recognition from Accelerometer Data. In *Proceedings of the 17th Conference on Innovative Applications of Artificial Intelligence - Volume 3*. IAAI'05. AAAI Press. 2005. ISBN 1-57735-236-x. pp. 1541–1546.

[13] Steve Aukstakalnis, D. B.: *Reálně o virtuální realitě - umění a věda virtuální reality*. Jota. 1994. ISBN 80-85617-41-2.

[14] World, N.: 40 virtual reality predictions. `http://www.networkworld.com/article/3043979/virtualization/40-virtual-reality-predictions.html`. March 2016. [Online; cited 26th May 2016].

Appendices

List of Appendices

Appendix A

In-person usability testing

The purpose of the questionnaire is to evaluate web application called iAvatar. This application is a result of bachelor's thesis. Topic is concept for real-world person avatar usage in virtual city. Below every question is space for writing task duration, user's reaction and everything connected to the question with emphasis on his opinion about simplicity and clarity of tasks execution. Also user can share his ideas and suggestions for improvements.

1. Try to register, logout and login.

2. Try to write a message from map and chat panel.

3. Try to move around the map – including zooming and rotation.

4. What do you think about message notification ?

5. Try “Walk„ functionality and explore new places in web application.
Try filter functionality and edit avatar in mobile application.

Would you recommend this application to your friends and why ?

Appendix B

Remote usability testing

Dear respondents, please take a few minutes of your time to fill out the following questionnaire. The purpose of the questionnaire is to evaluate web application called iAvatar. This application is a result of bachelor's thesis. Topic is concept for real-world person avatar usage in virtual city. Below every question is space for you where you can write everything connected to the question with emphasis on opinion about simplicity and clarity of tasks execution. Also you can write your ideas and suggestions for improvements.

1. Try to register, logout and login.

2. Try to write a message from map and chat panel.

3. Try to move around the map – including zooming and rotation.

4. What do you think about message notification ?

5. Try “Walk„ functionality and explore new places.

Would you recommend this application to your friends and why ?

Appendix C

CD content

- src/mobile/ – mobile application source code
- src/web/ – web application source code
- report.pdf – technical report
- poster.pdf – presentation poster
- 2016-xmulle20-virtual_oulu.mp4 – presentation video
- SECURITY.txt – security rules for Firebase
- README.txt – installation manual

Appendix D

Installation manual

How to run native mobile application

- register and download newest version of Xcode from official Apple site [1]
- delete `Virtual Oulu.xcodeproj` from Project navigator and copy again
- open `Virtual_Oulu_database.xcworkspace` in Xcode
- create your team provisioning profile [2]
- choose connected device with iOS 8.0+ or choose iOS simulator with iOS 8.0+
- press build and run button

How to run web application

- copy source code on web server and open `index.html` with web browser

or

- run localhost server, copy source code on localhost server and open `index.html` with web browser

or

- open `index.html` with web browser and ensure that run JavaScript from local is enabled

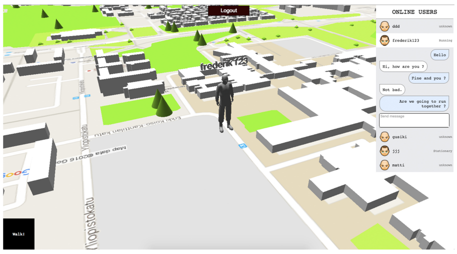
Notice: You have to use web browser which support Web-based Graphics Library (WebGL).

[1] https://developer.apple.com/download/

[2] https://developer.apple.com/library/ios/documentation/IDEs/Conceptual/AppStoreDistributionTutorial/CreatingYourTeamProvisioningProfile/CreatingYourTeamProvisioningProfile.html

Appendix E

Poster



ONLINE USERS

ddd unknown

frederik123 Running

Hello

Hi, how are you ?

Fine and you ?

Not bad.

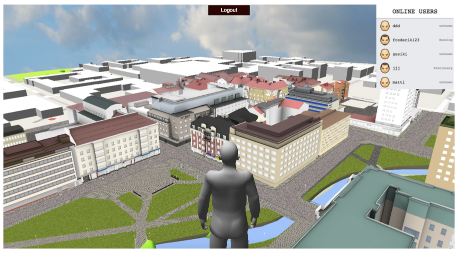
Are we going to run together ?

Send message

gusiki unknown

jjj Stationary

matti unknown



ONLINE USERS

ddd unknown

frederik123 Running

gusiki unknown

jjj Stationary

matti unknown

BRNO UNIVERSITY OF TECHNOLOGY

FACULTY OF INFORMATION TECHNOLOGY

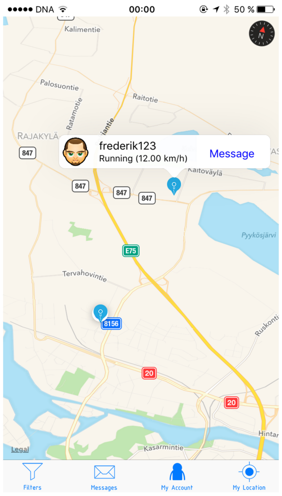
Concept for Real-World Person Avatar Usage in Virtual City

Bachelor's Thesis

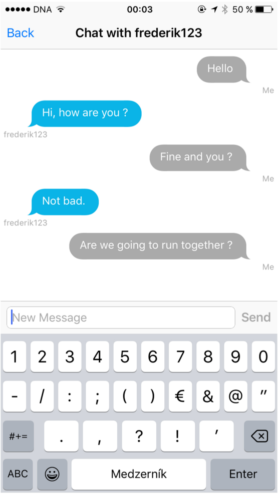
Frederik Müller
Author

Ing. Vítězslav Beran, Ph.D.
Supervisor

Matti Pouke. Ph.D.
Supervisor



frederik123
Running (12.00 km/h)
Message



Chat with frederik123

Hello
Me

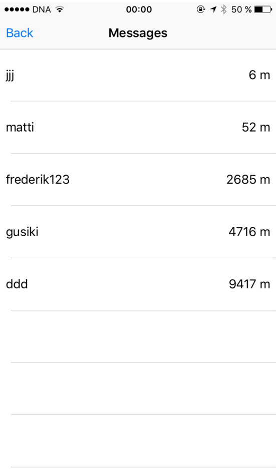
Hi, how are you ?
frederik123

Fine and you ?
Me

Not bad.
frederik123

Are we going to run together ?
Me

New Message Send



Messages

jjj	6 m
matti	52 m
frederik123	2685 m
gusiki	4716 m
ddd	9417 m

Messaging on map
See your friend's location
Explore new city
Stunning 3D graphics

iOS Swift language
OpenGL and Three.js library
Firebase-powered app