SOFTWARE DEVELOPMENT METHODOLOGIES ON ANDROID APPLICATION USING EXAMPLE

Bumbak, Ivan

POLYTECHNIC OF ŠIBENIK

DEPARTMENT OF MANAGEMENT

SPECIALIST STUDY OF MANAGEMENT

Ivan Bumbak

SOFTWARE DEVELOPMENT METHODOLOGIES ON ANDROID APPLICATION USING EXAMPLE

Graduate thesis

Šibenik, 2018.

POLYTECHNIC OF ŠIBENIK

DEPARTMENT OF MANAGEMENT

SPECIALIST STUDY OF MANAGEMENT

SOFTWARE DEVELOPMENT METHODOLOGIES ON ANDROID APPLICATION USING EXAMPLE

Graduate thesis

Course: Software engineering

Mentor: PhD Frane Urem, college professor

Student: Ivan Bumbak

Student ID number: 0023096262

Šibenik, September 2018.

TEMELJNA DOKUMENTACIJSKA KARTICA

Veleučilište u Šibeniku

Odjel Menadžmenta

Diplomski specijalistički stručni studij Menadžment

Diplomski rad

Razvojne metode programa na Android platformi koristeći primjer

Ivan Bumbak

bumbak.ivan@gmail.com

Postoji mnogo razvojnih metoda programskih rješenja koje se mogu koristiti za razvoj istih na bilo kojoj platformi. Koja metoda će se koristiti ovisi o zahtjevnosti samog projekta, koliko ljudi radi na projektu, te u kojem vremenskom roku projekt mora biti isporučen. U svrhu ovog diplomskog rada razvijena je Android aplikacija putem tradicionalne metode, iako su danas sve više i više popularne takozvane *agile* metode. *Agile*, ili agilan, znači biti brz i sposoban reagirati na vrijeme te prilagoditi se svim promjenama u bilo kojem trenutku razvoja projekta. U radu su objašnjenje najpopularnije *agile* metode te su prikazane prednosti korištenja *agile* metoda u odnosu na tradicionalnu metodu.

(37 stranica / 37 slika / 0 tablica / 14 literaturnih navoda / jezik izvornika: engleski)

Rad je pohranjen u: Knjižnici Veleučilišta u Šibeniku

Ključne riječi: metode, agile, razvoj, aplikacija

Mentor: dr.sc. Frane Urem, prof. v.š.

Rad je prihvaćen za obranu: 7. rujna, 2018

BASIC DOCUMENTATION CARD

Polytechnic of Šibenik Graduate thesis

Department of Management

Specialist study of Management

SOFTWARE DEVELOPMENT METHODOLOGIES ON ANDROID APPLICATION USING EXAMPLE

Ivan Bumbak

bumbak.ivan@gmail.com

There are many software development methodologies which can be used for developing any kind of software, on any platform. Which method will be used depends on how much project is big, how many people work on it and in what time needs to be finished and delivered. For the purpose of this thesis Android application was developed using traditional methodology. However, nowadays agile methodologies are more and more popular. Agile means, quickly and adaptable to any kind of changes at any point in development time. Thesis also describes the most famous agile methodologies and their advantages compared to traditional development methodology.

(37 pages / 37 figures / 0 tables / 14 references / original in English language)

Paper deposited in: Library of Polytechnic in Šibenik

Keywords: software, development, methodology, agile, android

Supervisor: PhD Frane Urem, college professor

Paper accepted: September 7, 2018

Table of Contents

1. Introduction

Android applications are becoming increasingly more used in our daily life and almost in every aspect of our lives. People use Android applications for navigation, shopping, tracking daily activity, daily water supply, communication etc. Except from their common use, we can connect our mobile phones on our house appliances and use them for controlling heat, lights, food supply and more. Using Android applications is very easy but making them can be very hard and tricky. It takes time to make application perfect, and programmers are not the only ones involved in making them. Users are one of the main parts of its idea, usage and existence. Without users, programmers cannot know what they want, and if programmers do not know what users seek, they cannot make any application.

This graduate thesis is going to describe what an Android application is, when it was introduced, what the main programming languages for developing Android applications are and what methodologies are in use today for application development. Also, there will be provided an example of building and testing Android applications. The application, used as an example, was developed in Google's software, Android Studio 3.0 by the author of this graduate thesis. Application "eVUŠ" calculates student scholarship based on a difference between enrolled ECTS[1] and failed ECTS points. There are also some filters like choosing your study or enrolling year because of changed price through years. After selecting graduating year and type of study, a student clicks on the ''calculate'' button and application will show how much of scholarship he or she has to pay in Croatian kuna and total of gained ECTS points. Application is made only for the students of Polytechnic of Šibenik and for the purpose of this graduate thesis.

[1] ECTS – European credit transfer system – points based on learning achievements in Bologna Process

2. Story of Android

We have been using Android for a long time now. However, it has been around 10 years since the Android OS[2] was introduced to consumers. Thanks to Google, Android is an open source OS[3] which allowed it to become highly popular with third-party phone makers, e.g. HTC, Huawei, Samsung, etc. Today, Android OS has become the most popular OS in the world. It defeated many competitors like Symbian, BlackBerry and Windows Phone, but still its greatest and most serious competitor is Apple's OS.

2.1. Founding Android

The company, Android Inc, was founded in Palo Alto, California on October 2003 by four founders: Rich Miner, Nick Sears, Chris White and Andy Rubin. One of its founders, A. Rubin, in 2013 revealed that Android OS was originally meant to improve the operating systems of digital cameras[4]. The company made pitches in 2004, and showed how Android, installed on camera, would wirelessly connect to a PC which would be connected to Android Datacenter, where camera owners can store their photos on a cloud server. However, in that time, market for stand-alone digital cameras was declining and few months later after their pitch the company shifted using the same OS towards mobile phones.

Next big step in Android's history was made when Google decided to acquire the original company in 2005. Android's founding member continued to develop the OS, but together with Google, they decided to use Linux as base software. This meant that Android can be offered to third-party mobile phone manufactures for free.

Until now, they have launched 15 different versions of Android, and are currently developing the 16th. Every version had different name, except the first two versions, and Android used names of sweets for their versions of OS. Also, as every other programming software with graphical user interface (GUI), Android uses APIs[5]. API is a set of routines, protocols, and tools for building software applications.[6]

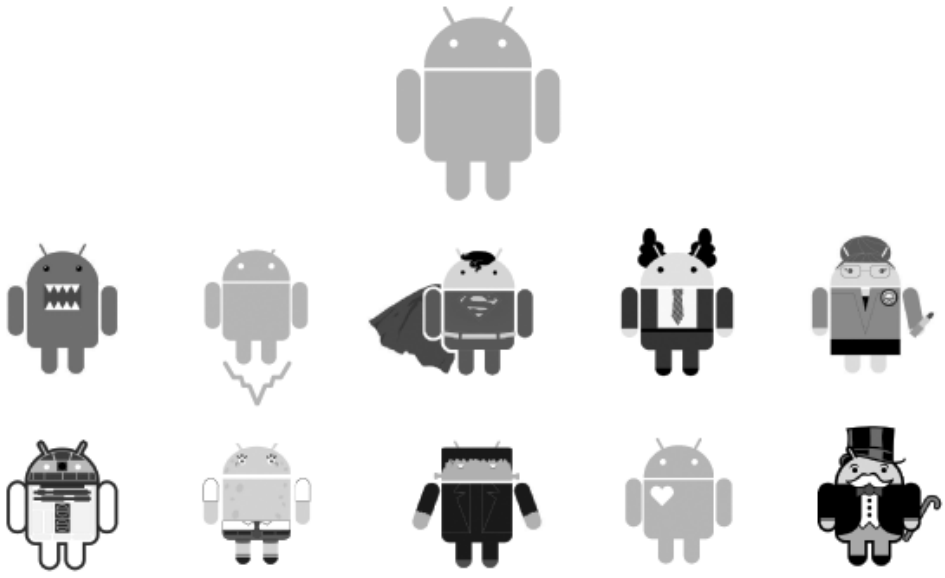
[2] Operating system – system software that manages computer hardware and software resources and provides common services for computer programs
[3] Open source OS – type of software with its source code made available with a license in which copyright holder provides the rights to study, change, and distribute the software to anyone and for any purpose
[4] https://www.pcworld.com/article/2034723/android-founder-we-aimed-to-make-a-camera-os.html
[5] API – Application program interface
[6] https://www.webopedia.com/TERM/A/API.html

2.2. Android Logo

The logo, as we know it today, looks like a combination of a robot and a green bug, was designed and created by Irina Blok while she was working for Google in 2007. The only directive she had was to make it look like a robot. The inspiration came from non-other than the signs of the universal man and woman that often appears on restroom doors in restaurants, café bars, etc.

Designing team agreed that the logo, like the software, should be open-sourced. This means that everybody can take initial logo and make its own design, without being sued for copyright. Since then, Android logo has been dressed up as many characters and given many different features like skins, skateboards, etc. Illustration 1 shows different Android logo skins through years since its original creation. Also, every version of Android OS has its own logo which associates the given name for each version of OS. Except that, in front of Google Visitor Centre building, California, there are placed statues for each version of Android OS.

Illustration 1. Android logo



Source: https://www.nytimes.com/2013/10/13/magazine/who-made-that-android-logo.html

3. Versions of Android OS

There are 8 different major versions released, and currently Google is working on its 9th version of software. Its name has not been revealed yet, and it is known as Android P, or Android 9.0. As was mentioned before, every version has its own code name. Code names were used from Android 1.5 until today. The first two versions did not have code name, but version 1.1 was internally known as ''Petit Four[7]''. There is no specific reason why Google developers choose to use candy and dessert names for Android code name. However, after Android 4.4 was released, Google offered official statement about code names, saying, '' Since these devices make our lives so sweet, each Android version is named after a dessert.''[8]

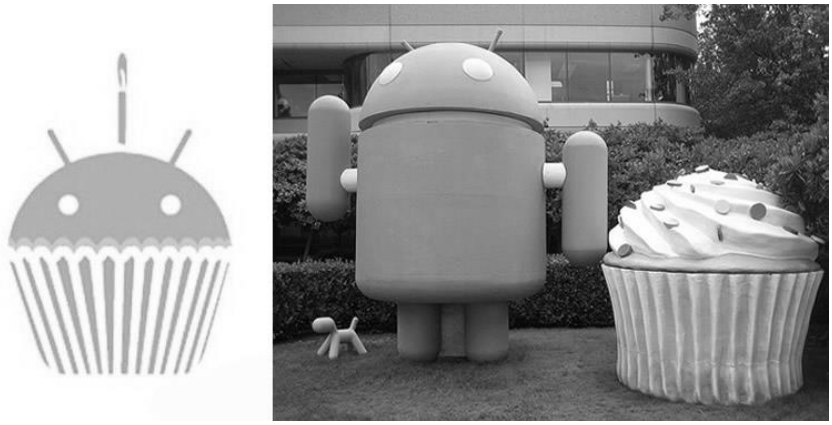
First two versions of Android, beta version of Android 1.0, were released on November 5, 2007. This version already had today's globally known Google's business plan trademarks. It integrated other company's products and services, e.g. Google Maps, YouTube and HTML browser and Android Market. In February 2009, there was a minor update to Android 1.1.

3.1. Android 1.5, Cupcake

Cupcake was the first official public code name for Android OS. Cupcake was released in April 2009. It contained a few new features and improvements compared to its previous version. Cupcake had ability to upload videos to YouTube, back than hard to imagine, support for third-party keyboards and automatic display rotation to the right position. The first versions of Samsung Galaxy phone, and HTC Hero were the first phones which had installed Cupcake out of the box. Illustration 2 shows the original logo of Android Cupcake version and its statue placed in front of the company.

[7] Petit four – small bite-sized confectionery or savoury appetizer
[8] https://www.androidauthority.com/history-android-os-name-789433/

Illustration 2. Android 1.5, Cupcake



Source: https://www.androidauthority.com/history-android-os-name-789433/

3.2. Android 1.6, Donut

Donut was launched only five months after launching its previous version, Cupcake. New features included support for carriers that used CDMA-based network, which allowed Android phones to be sold and used by all carriers around the world. Other features were introduction of Quick Search Box, quick toggling between camera and gallery, and Power Control widget for managing Wi-Fi, Bluetooth, GPS, etc. The first phone with Donut version was unpopular Dell Streak phone, which had huge 5-inch screen. Just for comparison, nowadays 5-inch display phones are considered to be average sized for a smartphone. Illustrations 3 shows original logo for Android Donut and its statue in front of the company.

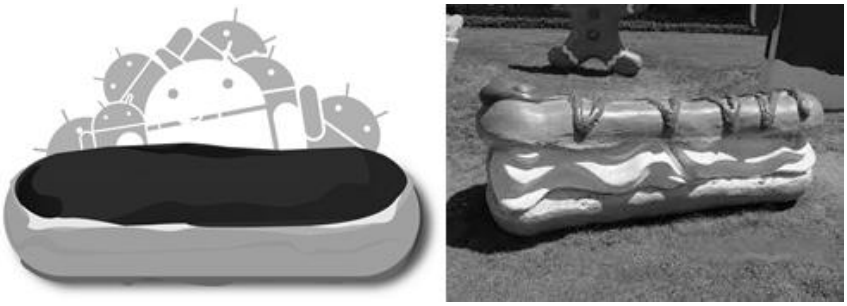
Illustration 3. Android 1.6, Donut

Source: https://www.androidauthority.com/history-android-os-name-789433/

3.3. Android 2.0-2.1, Eclair

Android 2.0, or code name, Eclair was launched in October 2009. In this version, Google introduced text-to-speech support for the first time, live wallpapers, multiple account support and Google Maps navigation. The first phone which came with preinstalled Eclair was Motorola Droid, and it was the first phone based on Android that was sold by Verizon Wireless. In that time, Lucasfilm used ‘’Droid’’ term as reference to robots in the movie franchise Star Wars, which also was trademarked. Because of that, Motorola had to get permission and pay some money to Lucasfilm to use Droid as the name for their phone. Motorola used Droid brand for many phones until 2016. Next illustration shows its original logo and its statue.

Illustration 4. Android 2.0-2.1, Eclair



Source: https://www.androidauthority.com/history-android-os-name-789433/

3.4. Android 2.2, Froyo

Android 2.2, Froyo was launched in May 2010. Froyo is short for frozen yogurt. Phones with this version of software had several new features, like Wi-Fi mobile hotspot functions, push notifications via Android Cloud to Device Messaging service, etc. The first smartphone which was able to receive Froyo update was Google’s Nexus brand phones, the Nexus One.

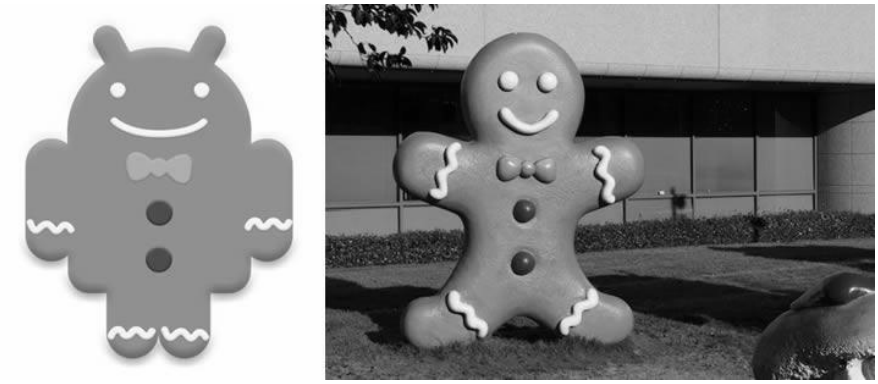
Illustration 5. Android 2.2, Froyo

Source: https://www.androidauthority.com/history-android-os-name-789433/

3.5. Android 2.3, Gingerbread

Android 2.3, with code name Gingerbread was released in September 2010 and it is currently the oldest version of the Android OS that is still listed on monthly platform version update page. Gingerbread version refreshed user interface and added support for NFC[9] functions, but only for devices which had the required hardware, NFC chip. The first phone which added both Gingerbread and NFC hardware was the Nexus S, developed by cooperation of Google and Samsung. Except NFC function, Gingerbread brought groundwork for selfie, because of support for multiple cameras and video chat support within Google Talk. Illustration 6 shows Gingerbread logo and its statue.

Illustration 6. Android 2.3, Gingerbread



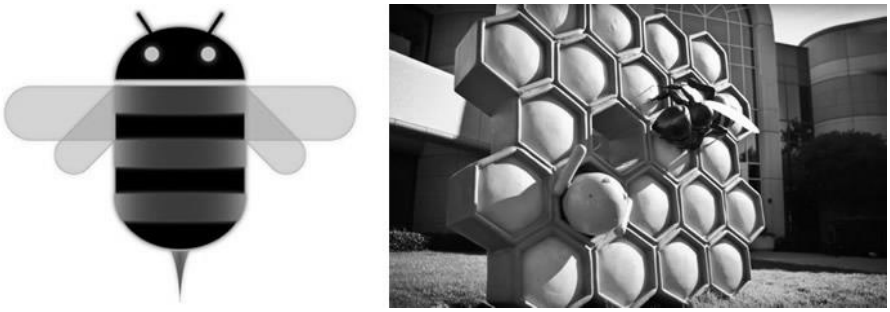
Source: https://www.androidauthority.com/history-android-os-name-789433/

3.6. Android 3.0, Honeycomb

Android 3.0, called Honeycomb, was introduced in February 2011 along with first Motorola Xoom tablet. This version of Android OS was little bit oddball compared to other versions. It was released for installation only on tablets and other mobile devices with larger displays than current smartphones. It brought redesigned user interface, specifically for large screens, and notification bar which was placed on the bottom of a tablet's display. Its idea was to offer specific features that could not be handled by the smaller displays. Honeycomb was also replayed for their competitor's Apple iPad, but Honeycomb ended up as unnecessary version. Illustration 7 shows Honeycomb original logo and its statue in front of the company.

[9] NFC – near field communication

Illustration 7. Android 3.0, Honeycomb

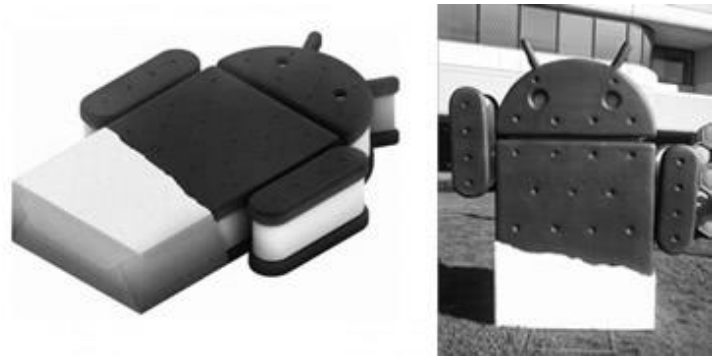


Source: https://www.androidauthority.com/history-android-os-name-789433/

3.7. Android 4.0, Ice Cream Sandwich

Android 4.0, publicly known as Ice Cream Sandwich was released in October 2011, 8 months after its previous version. It was feature combination of Gingerbread and Honeycomb versions. Ice Cream Sandwich introduced face detection for unlocking the phone. Except this feature, it contained wipe gestures for dismissing notifications and browser tabs and ability to monitor data usage of mobile network or Wi-Fi. As of today, there is 0.4%[10] of all Android devices still running on Android 4.0 version. Illustration 8 shows original Ice Cream Sandwich logo and its statue.

Illustration 8. Android 4.0, Ice Cream Sandwich



Source: https://www.androidauthority.com/history-android-os-name-789433/

3.8. Android 4.1-4.3, Jelly Bean

Era of Android Jelly Bean started in June 2012. It had two updated version with the same name, Android 4.2 released in October 2012, and year after first Jelly Bean version, Android 4.3 in

[10] February 5th, 2018 - https://www.statista.com/statistics/271774/share-of-android-platforms-on-mobile-devices-with-android-os/

July 2013. In this version Google updated notification panel adding more content and action buttons, which are today unimaginable. Its 4.2 version included full support of the Android version of Google's Chrome web browser. First device which came with pre-installed Android 4.1, Jelly Bean version was Google Nexus 7 tablet. All versions of Jelly Bean are still very much active on Android devices. As for now, there is about 5%[11] of all Android devices which use Jelly Bean versions. Illustration 9 shows original Android Jelly Bean logo and its statue.

Illustration 9. Android 4.1-4.3, Jelly Bean

Source: https://www.androidauthority.com/history-android-os-name-789433/

3.9. Android 4.4, KitKat

Android 4.4, or Android KitKat, is the first version of Android OS that uses a name of some candy trademark. It was officially launched in September 2013. Its first name was not as we know it today. Google, at their conference ''Google I/O conference'', that year said that codename for Android 4.4 would be Key Lime Pie. However, Google's director of Android global partnership J. Lagerling thought ''Key Lime Pie'' would not be familiar name enough, so he called Nestle and they agreed to use its candy bar trademark for the code name of new Android version. Nestle also released their famous candy bar shaped like Android robot mascot. This version did not have so many new features, but it was optimized to run on smartphones which had only 512 MB of RAM. Google's smartphone Nexus 5 was the first smartphone with pre-installed KitKat version. It has been 5 years from its release and there is still 12%[12] of all Android devices which runs on Android KitKat.

[11] [12] February 5th, 2018 - https://www.statista.com/statistics/271774/share-of-android-platforms-on-mobile-devices-with-android-os/

Illustration 10. Android 4.4, KitKat

Source: https://www.androidauthority.com/history-android-os-name-789433/

3.10. Android 5.0, Lollipop

Android 5.0, commonly known as Android Lollipop was launched in the fall of 2014 and it brought a major overall look difference of the OS. Lollipop was the first version that used Google's new Material Design language, which made liberal use of shadow effects, simulate a paper-like look of the Android user interface, revamped navigation bar, rich notifications for the lock screen, etc. Android 5.1 update made a few more under the hood changes: support of dual-SIM, HD Voice calls, and Device Protection to keep thieves locked out of your phone even after a factory reset. The first devices which came with pre-installed Lollipop were Google's Nexus 6 smartphone and its Nexus 9 tablet. Statistics from February 2018 show that there is 24.6%[13] of all Android devices which runs on Android Lollipop and makes it the third most used version at the moment. Illustration 11 shows original logo and Android Lollipop statue.

Illustration 11. Android 5.0, Lollipop

Source: https://www.androidauthority.com/history-android-os-name-789433/

[13] https://www.statista.com/statistics/271774/share-of-android-platforms-on-mobile-devices-with-android-os/

3.11. Android 6.0, Marshmallow

Android 6.0, code named Marshmallow is 6th major version of Android OS. It was released in the fall of 2015. It introduced features such as vertically scrolling app drawer, Google Now on Tap, support of fingerprint unlocking, USB Type-C support, introduction of Android Pay, etc. The first devices which came with pre-installed Marshmallow were Google's Nexus 6P and Nexus 5X smartphones, along with Google's Pixel C tablet.

Android Marshmallow takes 2nd place on the list of most installed Android OS version. On February 5th, 2018 there was 28.1%[14] of all Android devices which were running on this version of Android. It is most popular version together with Android 7.0 or commonly known as Android Nougat. Illustration 12 shows Android Marshmallow original logo and its statue in front of the company.

Illustration 12. Android 6.0, Marshmallow

Source: https://www.androidauthority.com/history-android-os-name-789433/

3.12. Android 7.0, Nougat

Android 7.0, known as Android Nougat is currently the most popular version of Android OS. Also, the most of today's Android devices run on Android 7.0 version, despite the Android 8.0 Oreo. There is 28.5%[15] of all Android devices which use this version. Android Nougat was introduced 2 years ago, in the fall of 2016. This version included better multi-tasking functions, split-screen mode for bigger displays, quick switching between apps and many other features. Google's Pixel, Pixel XL and LG V20 were the first phones which came with pre-installed

[14] [15] https://www.statista.com/statistics/271774/share-of-android-platforms-on-mobile-devices-with-android-os/

Android Nougat. Illustration 13 shows official logo of Android Nougat and its statue which is placed in front of the company building.

Illustration 13. Android 7.0, Nougat

Source: https://www.androidauthority.com/history-android-os-name-789433/

3.13. Android 8.0, Oreo

Android 8.0, code name, Oreo, is currently the newest version of Android OS. It was released in August 2017. Android Oreo contains a number of major features including picture-in-picture support for video, performance improvements, battery usage optimization and many others features. This version is still very new and some smartphones which were introduced around the same date will receive official update through May and June 2018. Android Oreo is the 2nd version which uses candy bar trademark. As of April 16, 2018, there are 4.65% of all Android devices which run on Oreo.[16] Interesting fact is that 5% of all Android devices run on Android 4.1-4.3 versions, Jelly Bean version.[17]

Nowadays, Google is working on its 9th version of Android OS. Still, Android 9.0 is known only as Android P, and it is still in development. Recently they published beta version, but it would be out by the fall of 2018, as its new sweet code name. Illustration 14. shows Android Oreo official logo and its statue.

Illustration 14. Android 8.0, Android Oreo

[16] https://developer.android.com/about/dashboards/
[17] This data was actual in May 2018

Source: https://www.seroundtable.com/photos/google-android-oreo-statue-24336.html

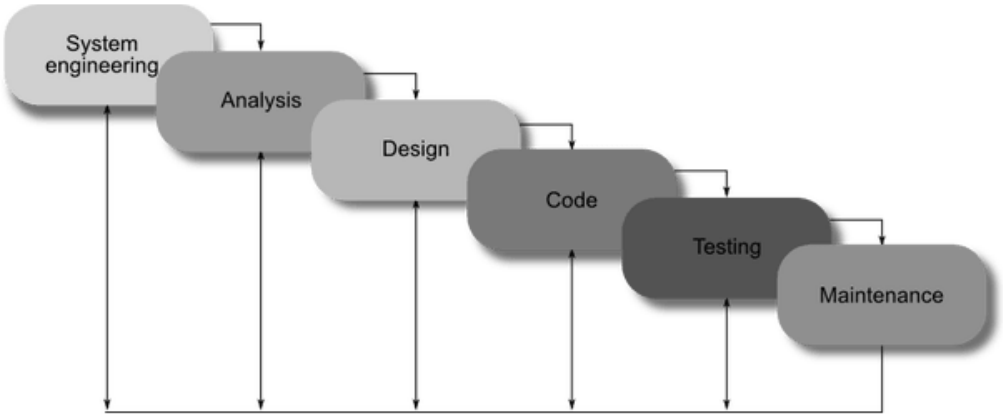
4. Development of "eVUŠ" Android Application

There are many ways how to develop an application or software. Developers can take their time and develop it as they want, or if they are working in teams or cooperating with potential users they can use some of existing software development methodologies like traditional methodologies or agile methodologies which are becoming a standard in developing any kind of projects. Software development methodology in software engineering, including application development for mobile devices, is conceptual framework which is used to plan, design, structure and test or control developing processes. Application "eVUŠ" was developed by waterfall methodology. "eVUŠ" is a very simple android application which calculates enrolment student scholarship based on collected ECTS credits. Development lasted from July 2016 to October 2016.

4.1. Waterfall Methodology

Waterfall methodology is traditional development methodology introduced by Dr. Winston W. Royce in a paper published in 1970. This model emphasizes that steps of software development life cycle are the same as cascading steps down an incremental waterfall. In short, when one phase is done, the next one can begin and not before. Nowadays, waterfall methodology has waned in a favour of agile methodologies. However, logical nature of waterfall processes cannot be denied, and it remains a common design process in the industry. It has 6 phases of development: requirements, analyses, design, coding, testing, operations.

Illustration 15. Waterfall model



Source: https://airbrake.io/blog/sdlc/waterfall-model

4.1.1. Requirements

Requirement is the first and initial phase of waterfall methodology. In this phase developers are writing down and documenting potential requirements of the application which will serve as the basis for all future development. It defines what application should do, not how it works. Requirements are documented together with someone who ordered the application. "eVUŠ" was ordered by Polytechnic of Šibenik. Together with them, developer has collected next important requirements:

a) Application should calculate scholarship based on collected ECTS. This means that a user must be able to enter how many of ECTS he or she has enrolled. Also, there must be an indicator by which total amount price will be calculated. For example, difference between enrolled credits and failed credits should give a total amount of collected credits which later will be multiplied by price of 1 ECTS credit.
b) As the price of 1 ECTS is not same for all types of study, a potential user must have option to choose proper type of study
c) There must be a constant amount of enrolment fees which needs to be calculated into total amount of scholarship
d) Application needs to show to user the total amount of collected ECTS credits and total amount of scholarship
e) There are different rules for paying scholarship: if a student has more than 55 ECTS credits he or she will pay only enrolment fee. However, if he or she has collected less than 30 ECTS credit he or she has to pay the whole amount of scholarship.

After all potential requirements are collected a developer can move on to the next phase.

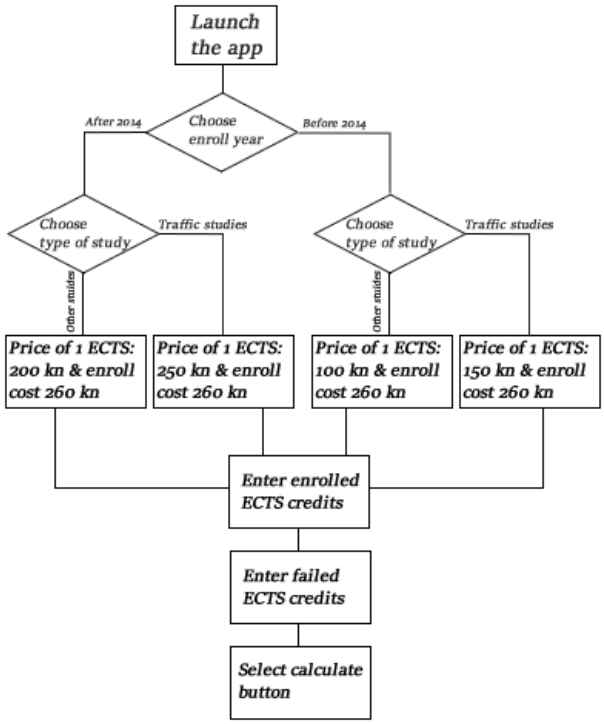
4.1.2. Analysis

Analysis is the second phase in waterfall methodology. This phase should provide analysed system in order to properly generate the models and business logic that will be used in application. It should show step by step how application will work. So, steps for "eVUŠ" are next:

1) User launches the application
2) User needs to choose between enrolment years. If users choose enrolment academic year before 2014/2015 their price will be calculated with lower price per 1 ECTS, else it will calculate with higher price.
3) After choosing enrolment year, user will choose type of study. If he or she chooses traffic type of studies, he or she will have different price per 1 ECTS.
4) User enters enrolled ECTS credits.
5) User enters failed ECTS credits.
6) User selects ''calculate'' button.

For easier understanding, next illustration shows flow-chart diagram with all steps mentioned above.

Illustration 16. Flow-chart of "eVUŠ" Android application



Source: Author's own work

4.1.3. Design

In this phase, developer covers technical design requirements, such as programming language, data layers etc, but it does not define how its UI[18] should look like. This phase outlines how exactly the business logic covered in analysis will be technically implemented.
"eVUŠ" application was developed in Google's software called "Android Studio 3.0" in Java programming language. Applications UI was built using XML[19] language. Using drawn flow-chart "eVUŠ" should contain:

a) Some kind of a list from which user can choose between 4 different types of studies. For example, list can be placed in dropdown box for easier selecting.
b) Text field which will store users enrolled ECTS credits.
c) Text field which will store users failed ECTS credits.
d) Button which will show calculated numbers
e) Places for displaying total ECTS credits, and total amount of scholarship that has to be paid for enrolment in next academic year.

4.1.4. Coding

In this phase developers are writing actual source code of an application or software. It implements all models, business logic, services and integrations that were specified in the all prior stages. Next illustrations show some parts of code of "eVUŠ" application.

Illustration 17. Source code of "eVUŠ" application

[18] UI – user interface in the industrial design field of human–computer interaction, is the space where interactions between humans and machines occur
[19] XML – extensible markup language – language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable

MainActivity.java

```
43          super.onCreate(savedInstanceState);
44          setContentView(R.layout.activity_main);
45
46          ActionBar actionBar = getSupportActionBar();
47
48          //Initialization
49          userEcts = (EditText) findViewById(R.id.ects_input);
50          enrolledEcts = (EditText) findViewById(R.id.ects_enrolled);
51          ectsPriceText = (TextView) findViewById(R.id.ects_price_text);
52          courses = (Spinner) findViewById(R.id.course_spinner);
53          laterEnrolmentRB = (RadioButton) findViewById(R.id.laterEnrolmentRadioButton);
54          earlierEnrolmentRB = (RadioButton) findViewById(R.id.earlierEnrolmentRadioButton);
55          summaryEctsText = (TextView) findViewById(R.id.text_summary_ects_number);
56          summaryPriceText = (TextView) findViewById(R.id.text_summary_price_number);
57          calculateBtn = (Button) findViewById(R.id.calculate);
58
59          //Setting clickListener on button
60          calculateBtn.setOnClickListener(new View.OnClickListener() {
61              public void onClick(View v) {
62                  //When the button is clicked, call the calculate method.
63                  calculate();
64              }
65          });
66
67          //Initialization of list of courses for Spinner
68          final String[] courseArray = new String[4];
69          {
70              courseArray[0] = getString(R.string.undergraduate_man);
71              courseArray[1] = getString(R.string.undergraduate_traf);
72              courseArray[2] = getString(R.string.undergraduate_admin);
73              courseArray[3] = getString(R.string.graduate_man);
74          }
```

Source: Author's own work

Illustration 18. shows part of code written in Java programming langue. The code defines the rules and logic of the application. Illustration 18. shows part of code which defines UI settings written using XML markup language.

Illustration 18. UI source code of "eVUŠ" application

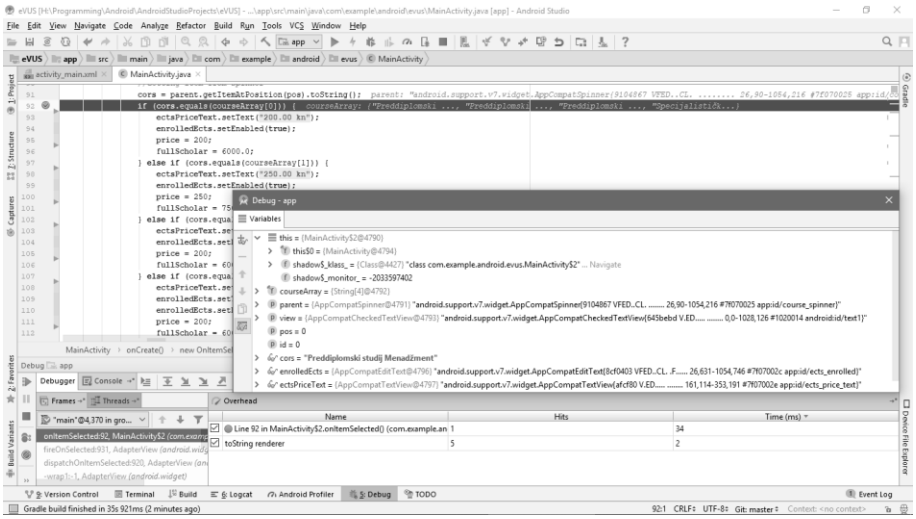
Source: Author's own work

4.1.5. Testing

Testing is fifth out of six phases in waterfall methodology. During this phase developers release beta versions[20]. Beta version is sent to a limited number of potential users. They will discover and report issues within the application that need to be resolved. This phase will probably cause necessary repeat of the previous coding phase. "eVUŠ" was sent to five different students on five different Android devices, OnePlus2, OnePlusX, Vernee Apollo, Huawei P9, Samsung Galaxy S6, including testing using Google Nexus 5 emulator as integrated part of Android Studio software and debugger tools and log statements which are also part of the same software. Illustration 19. shows part of debugger tool and log statements.

[20] Beta version - version of a piece of software that is made available for testing

Illustration 19. Debugger tool of "eVUŠ" application



Source: Author's own work

Testing result showed that choosing proper type of study does not change the price of 1 ECTS based on group of studies. Also, users notice that some of them do not have the same price per 1 ECTS as they enrolled few years earlier. That problem led to wrongly calculating the amount of scholarship. Except that users were unhappy with UI. Changes that were necessary:

a) Option for choosing enrolment year. Before academic year 2014/2015 there was different price, so for students who enrolled before that year scholarship is based on that price. Students have to be able to choose the enrolment year. This change belongs to requirement phase, which is the very first phase of waterfall method.
b) If application contains source code for selecting enrolment year and accordingly updating cost of 1 ECTS there should be the option for selecting the right year. So the best option would be radio buttons as users need to choose between two options: enrolment before academic year 2014/2015 and enrolment after academic year 2014/2015. This change belongs to third stage which covers design of the application and fourth stage in which developer writes code for all defined requirements.
c) Enrolment cost amount was wrong. Also, there is no need for displaying max enrolment ECTS credits. Fields for entering points are not so clear. Text in that input fields are not showing whole explanation.
d) Application does not have an explanation what a user should do with it.
e) UI should be a little bit modern. There is too much of blue colour. You cannot properly see dropdown box for choosing type of study.

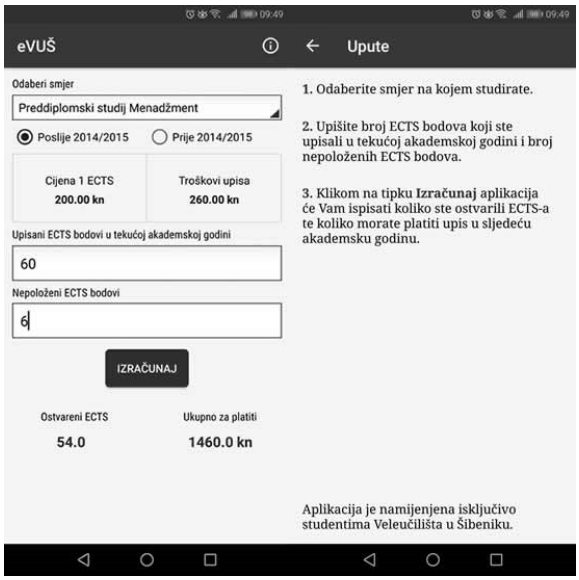
Every time users notice something a developer needs to go back to coding phase and fix the problem. Test it using debugger tools, and then is sent again to potential users to check if everything goes as it should. Next illustration shows the very first version of the application whose important changes are explained above.

Illustration 20. The very first version of "eVUŠ" application before testing

Source: Author's own work

After developer made changes what users and customer asked for the application was tested again. Now, there was everything they asked for, so the result is displayed on illustration 21.

Illustration 21. Final version of "eVUŠ" application. Main screen and instructions screen



Source: Author's own work

4.1.6. Operations

Operations are the last stage in waterfall methodology and at this point application is ready for live environment. It entails not just the deployment, but also support and maintenance that may be required. This phase is important to keep application up-to-date. Also, this phase actually never ends. There is constant need of updating an application because of updating Android software, faster phones, modern UI design, slow performance, etc. After Android application is ready there are small procedures which need to be done for publishing it on Google Play Store. After publishing there is constant feedback from users. It is very important for application to be up to date because users will not use it if developers do not follow up trends. In "eVUŠ" example, next update will be:

a) Adding new types of study as they are starting form academic year 2019/2020. This change belongs mainly to operations, the very last stage, but also as the code has to be changed a developer needs to go back to coding phase.
b) There will not be need for enrolment year choose option because there is no sense in choosing different option if there are no students who were enrolled before academic year 2014/2015. So higher price will be set as the only price of 1 ECTS. Next illustrations show adding new type of study in dropdown box.

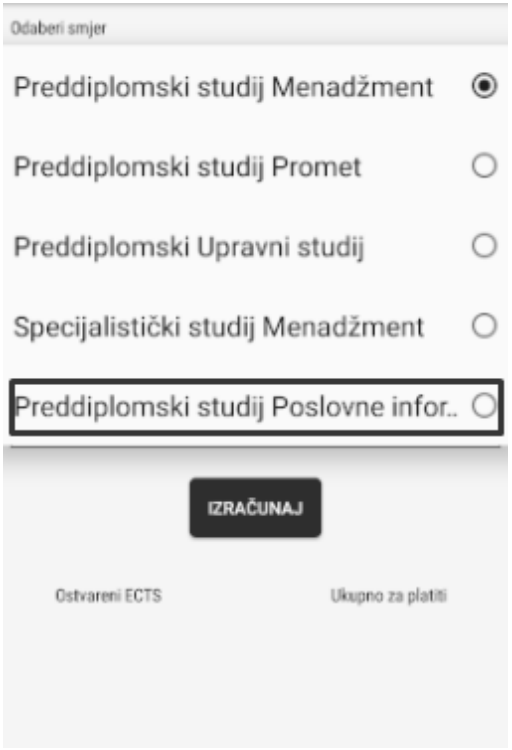
Illustration 22. Adding new type of study in dropdown box

```
//Initialization of list of courses for Spinner
final String[] courseArray = new String[5];
{
    courseArray[0] = getString(R.string.undergraduate_man);
    courseArray[1] = getString(R.string.undergraduate_traf);
    courseArray[2] = getString(R.string.undergraduate_admin);
    courseArray[3] = getString(R.string.graduate_man);
    courseArray[4] = getString(R.string.undergraduate_inf);
}
```

```
} else if (cors.equals(courseArray[2])) {
    ectsPriceText.setText("200.00 kn");
    enrolledEcts.setEnabled(true);
    price = 200;
    fullScholar = 6000.0;
} else if (cors.equals(courseArray[3])) {
    ectsPriceText.setText("200.00 kn");
    enrolledEcts.setText("60");
    enrolledEcts.setEnabled(false);
    price = 200;
    fullScholar = 6000.0;
} else if (cors.equals(courseArray[4])) {
ectsPriceText.setText("200.00 kn");
enrolledEcts.setEnabled(true);
price = 200;
fullScholar = 6000.0;
```

Source: Author's own work

Illustration 23. Adding new type of study in dropdown box



Source: Author's own work

4.2. Advantages of Waterfall Methodology

It is true that waterfall model has seen a slow phasing out in recent years when we compare it with agile methods, but it can still provide a number of benefits. Especially for larger projects and organisations that require the stringent stages and deadlines.

1. **It adapts to shifting teams** – the model allows project as whole to maintain, detailed, robust scope and design structure due to all planning and documentation stages. So it is very good for large teams that may see members come and go through life cycle of project. That way the whole concept and design is placed on the core documentation and not on any individual member.
2. **Model forces organization to be structured** – while developing some project waterfall model forces that project, even organisations to be extraordinary disciplined in its design and structure. Larger projects are disciplined and very well structured by its nature; also it includes detailed procedures to manage every aspect of the project, from design and development to testing and implementation. Developing smaller projects in this way can be very tiring and exhausting.

3. **Possible early design changes** – design changes can be very difficult, especially as projects goes further and further in its developing process. Waterfall model allows these changes because of documenting specification in the first stages together with developing team and clients. So, these alterations can be immediately done and with minimal effort, since there is no coding or implementation up to that point.
4. **Suited for milestone focused development** – as it mentioned before, waterfall model is very good for very large projects, it is also well suited for organizations or teams that work under a milestone and date focused paradigm. With clear and concrete stages, everyone on the team can understand and prepare for and because of that it is relatively simple to develop a proper timeline for the entire process.

4.3. Disadvantages of Waterfall Methodology

Some things in software development never change. When Dr. Royce introduced waterfall model for the first time it was ground-breaking. However, almost five decades later, bunch of cracks are starting to show up.

1. **Nonadaptive constrains** – the most demanding aspect of waterfall model is its inherent lack of adaptability across all stages of life cycle. Test stage, which is stage five in the process, can discover fundamental flaws in the design of the system. Except from requiring a dramatic leap backward in the stages, it can lead to devastating realization regarding the legitimacy of the entire system. For example, in "eVUŠ" application, in testing phase it can reveal that the logic, or conditions like choosing types of study and enrolment year are in some kind of conflict, and together they do not calculate nor scholarship nor gained ECTS points properly.
2. **It ignores mid-process potential user feedback –** as this model has strict step-by-step process, potential users of the software cannot give feedback until very late stages of entire life cycle. So late feedback can be very insufficient, and it is often too late to consider and make wanted changes. Project managers can enforce a process take step back and make unforeseen requirements or make changes coming for user, but both of that will be expensive and time-consuming for everyone on the project.
3. **Testing period is delayed** – most other modern methods are attempting to integrate testing as a fundamental and always-present process throughout whole development process, waterfall method avoids testing until late stages of the process. This not only

means that most bugs or other issues won't be discovered until late into the process, but it also encourages lackadaisical coding practices since testing is only an afterthought.[21]

Waterfall model has explicit software testing in its development, especially in implementation phase, but that testing is often too late and it's not enough for whole life cycle. However, this model is still known as traditional model, and maybe it is suitable for projects like "eVUŠ" Android application which is not so heavy, and it is not demanding. Still, when developing some software, users` feedback, their thoughts, and their testing should be mandatory throughout the whole life cycle of development, and whole developing team should take in consideration users` thoughts, ideas and suggestions.

[21] https://airbrake.io/blog/sdlc/waterfall-model

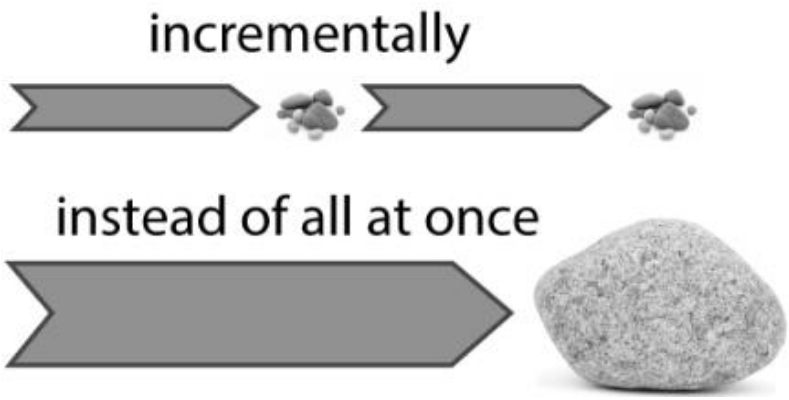
5. Agile Methodologies vs. Waterfall Methodology

There are many methodologies except Waterfall. Nowadays, developers are using more and more agile methodologies which means that developers are working more and more with users and customers through the whole life cycle of the project.

5.1. Agile methodologies

As an adjective agile means to move quickly and easily. It is the very same if we use it in software development glossary. Using this kind of method a developer approaches to problem incrementally instead developing and delivering all at once. Incrementally means that they break project on small pieces, priories them, and continuously deliver them in short week cycles which are called iterations.

Illustration 24. Agile methodologies scheme



Source: http://www.agilenutshell.com/

Developing by agile methodologies works like this:

a) Developer makes a list – together with a client makes a list of features a client wants in their software. These features are called user stories[22]
b) Sizing things up – using agile estimation techniques developers size that stories relatively to each other, and write down estimation time for developing all user stories
c) Setting priorities – there are always things to do than time for doing the same things. So at this point, developers ask a client to priories the features
d) Developing – going through the list from top to bottom. As a developer goes, it receives clients` feedback

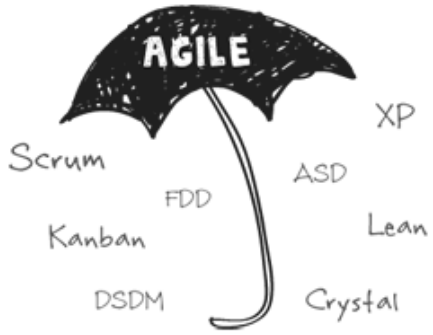
[22] http://www.agilenutshell.com/how_does_it_work

e) Updating as developing – as developer starts delivering the software these things can happen:
    a. Everything is good, and developer develops very fast and accurate. Or,
    b. There is no much time left and too much to do

    At this point developer has two choices:
    a. Cut the scope and do less than it is asked
    b. Ask for more time and more money

There are a bunch of agile methodologies, but the main are: scrum, Kanban, lean, extreme programming (XP), crystal, dynamic systems development method (DSDM), feature-driven development (FDD), etc.

Illustration 25. Agile methodologies



Source: https://www.versionone.com/agile-101/agile-methodologies/

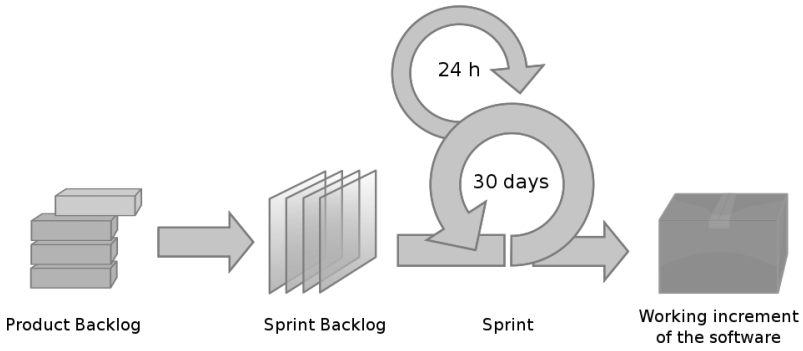
5.1.1. Scrum

Scrum is a lightweight agile project management framework with broad applicability for managing and controlling iterative and incremental projects of all types.[23] It is designed for teams of three to nine developers who break their work into actions, called sprints. These sprints can be delivered within 30 days. Before developer team take these actions, they iterate with product owner for easier identification and prioritization of system functionality stored in a from called product backlog. In product backlog client and development team document software features, bug fixes, requirements and whatever need to be done for successfully working software. Because of its simplicity and proven productivity, scrum has increasing

[23] https://www.versionone.com/agile-101/agile-methodologies/

popularity in the agile software development community. Illustration 26. shows the scrum process.

Illustration 26. Scrum process



Source: http://www.learningfacilitated.com/2016/04/bringing-scrum-to-education/

5.1.2. Lean Development and Kanban

Lean software development (LSD) is translation of lean manufacturing principles. It was adopted from Toyota. LSD focuses the team on delivering the value to a client. There are seven principles which describe LSD method:

a) Eliminate waste – everything that does not add value to a client
b) Amplify learning – while developing both development team and clients are learning more and more about development and the software by usage of short iteration cycles
c) Deciding as late as possible – in software development uncertainties are always present, so better results should be achieved with option-based approach delaying decisions as much as possible until they can be made based on facts
d) Deliver as fast as possible – technology grows very rapidly, so the sooner the complete product is delivered, the sooner users` feedback can be received
e) Empower the team – in software development managers are thought to listen developers, they do not see them just as resources, they let them do their work encouraging them, catching errors without micro-managing them
f) Build integrity in – the client, software owner, needs to have an overall experience of the system. They need to know how software is being advertised, delivered, deployed, accessed, how intuitive its use is, its price and how well it solves problems. Developing this way mistakes can be removed instantly

g) See the whole – nowadays, software is not just the sum of its part, but also the product of software interactions between developers and clients. Big tasks should be decomposed to smaller tasks because that way defects are easier found and eliminated

The Kanban methodology is used by organizations to manage the creation of products with an emphasis on continual delivery while not overburdening the development team[24]. Kanban method was design to help teams work together more effectively. It is based on three basic principles:

a) Visualise what you do now
b) Limit the amount of work in progress
c) Enhance flow

5.1.3. Extreme Programming

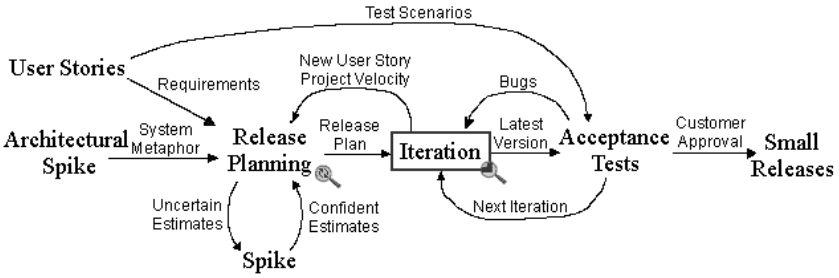
Extreme programming (XP) has emerged as one of the most popular and controversial agile methodologies.[25] XP describes disciplined approach of delivering software, both quickly and continuously. As every other agile methodology, XP promotes customer involvement, rapid customer feedback, close teamwork and continuously testing. Software delivery cycle is usually between 1 to 3 weeks. This methodology is based on four values:

a) Simplicity
b) Communication
c) Feedback
d) Courage

Developing by this method customer works very closely with development team so they can together define more prioritizing requirements and they refer them as user stories. Estimating, planning and delivering the highest priority user story is done by development team on iteration by iteration basis. Next four illustrations show XP flow-charts. Illustration 27 shows complete process for developing project, illustration 28 shows how iterations actually work within XP methodology, illustration 29 shows main development of a project and lastly illustration 30 shows planning and feedback loops.

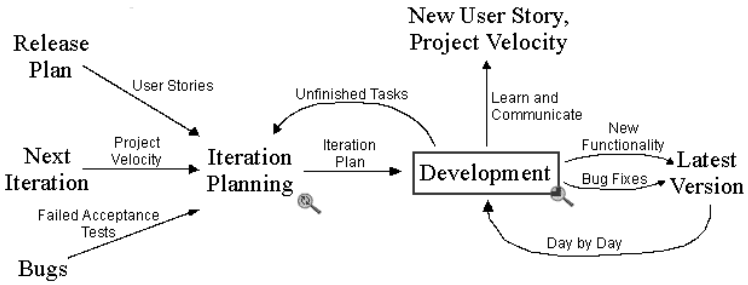
[24] [25] https://www.versionone.com/agile-101/agile-methodologies/

Illustration 27. Extreme programming flow-chart



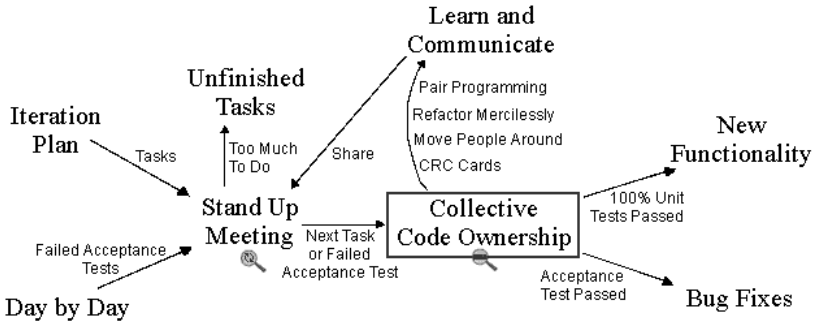
Source: http://www.extremeprogramming.org/map/project.html

Illustration 28. Iteration flow-chart



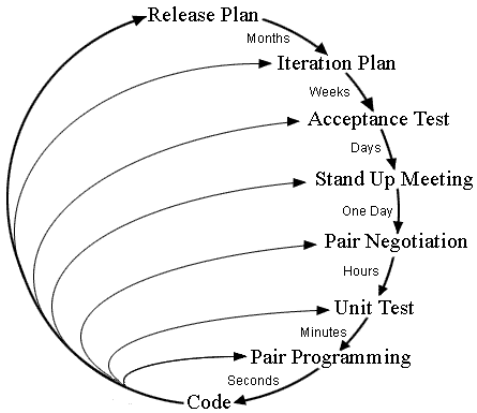
Source: http://www.extremeprogramming.org/map/iteration.html

Illustration 29. Main development process



Source: http://www.extremeprogramming.org/map/development.html

Illustration 30. Planning and feedback loops



Source: http://www.extremeprogramming.org/map/loops.html

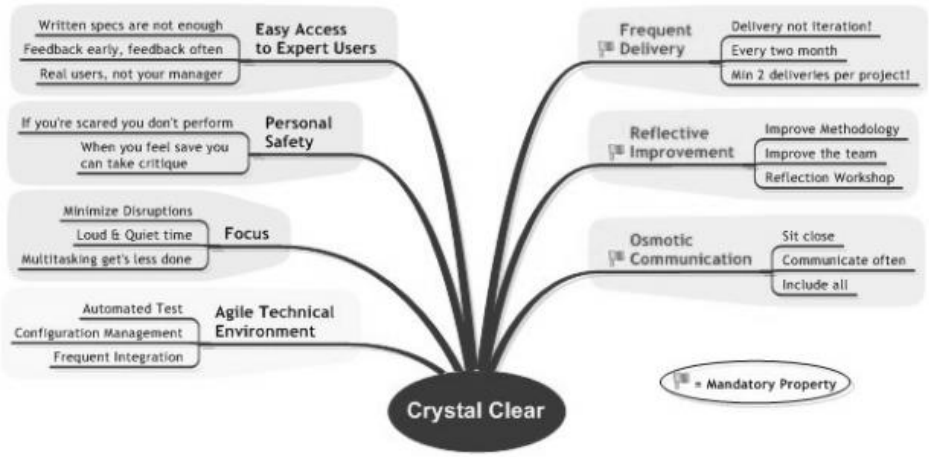
5.1.4. Crystal Methods

Crystal methods are whole family of software development methodologies as there are many of them like crystal clear, crystal sapphire, crystal yellow, etc and it represents one of the most lightweight and adaptable approaches to software development. Each crystal methodology is different, and which of them should developers use depends on the complexity of the software, team size, project priorities and similar things. For example, if a project may involve high risk to human life the project team will use crystal sapphire methodology, and if there is not with such risk the team will use crystal clear which is also the most selectable method for software development. These methodologies focus on 6 primary aspects:

a) People
b) Interaction
c) Community
d) Communication
e) Skills
f) Talents

It is also described by 7 priorities that indicate higher possibility of success and it includes frequent delivery, reflective improvement and easier access to expert users.

Illustration 31. Seven properties of crystal clear method



Source: https://project-management.com/xp-fdd-dsdm-and-crystal-methods-of-agile-development/

One of the biggest advantage of crystal methodologies is that they are adaptive. They do not have prescribed tools nor techniques, without too much documentation, management or reporting.

5.1.5. Dynamic Systems Development Method

Dynamic Systems Development Method, or short DSDM, was published in 1995 by the DSDM consortium[26] and since then evolved and matured to provide a comprehensive foundation for planning, managing, executing and scaling agile process. In this method requirements are baselined early in the project, rework placed into the process itself and all changes must be reversible. Requirements are prioritised using MoSCoW method, which stands for must have, should have, could have and will not have. ''Os'' are there for easier pronunciation[27]:

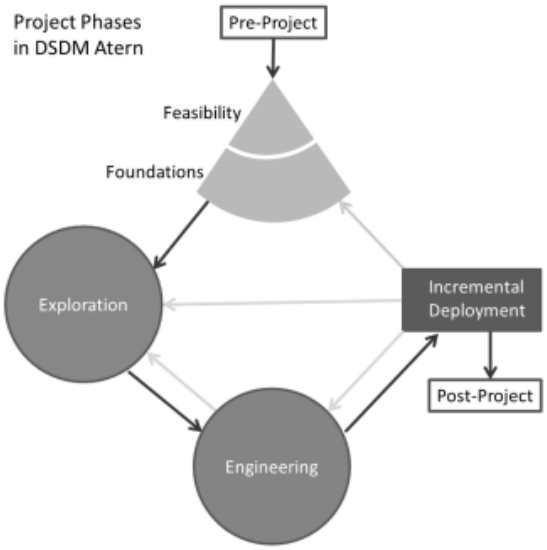
a) M – must have, represents requirements which are critical to current iterations for success
b) S – should have, represents important, but not necessary for deliver in current iterations
c) C – could have, represents desirable, but not necessary, which could improve user experience or customer satisfaction for small development cost
d) W – will not have, represents least critical, lowest payback items and not appropriate requirements at the time

DSDM consist of eight principles[28] that will directly create a mindset to deliver on time and within budget:

[26] DSDM consortium – association formed by vendors and experts in software engineering
[27] https://www.forbes.com/sites/alastairdryburgh/2015/08/19/moscow-rules/#2036ab3a2910
[28] https://agilekrc.com/resource/168/what-dsdm-and-8-principles

a) Focus on the business need
b) Deliver on time
c) Collaborate
d) Never compromise quality
e) Build incrementally from firm foundations
f) Develop iteratively
g) Communicate continuously and clearly
h) Demonstrate control

Illustration 32. DSDM Framework



Source: https://project-management.com/xp-fdd-dsdm-and-crystal-methods-of-agile-development/

5.1.6. Feature-Driven Development

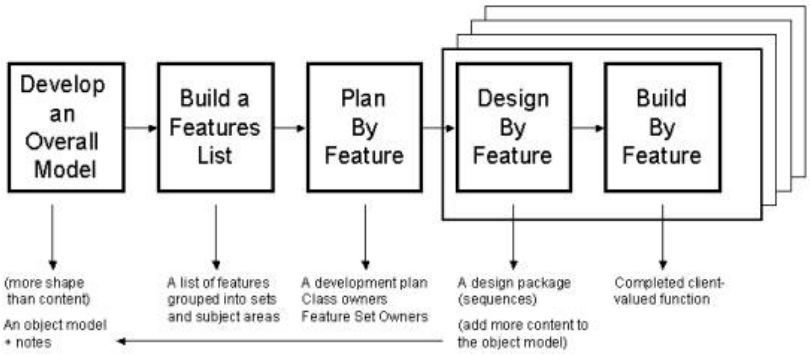
Feature-Driven Development (FDD) was built around software engineering best practice such as domain object modelling, developing by feature and code ownership. It consists of five basic activities:

a) Development of an overall model
b) Building of a feature list
c) Planning by feature

d) Designing by feature
e) Building by feature

Using this method, every project has its own unique model which will result in a feature list, and the last three activities are short iterative processes whose feature doesn't take longer than two weeks to build. However, if those features take more than two weeks, they need to be broken into smaller pieces. It begins with establishing an overall model shape and then continues with a series of small useful, useful for the client, results. Next illustration shows five basic principles of FDD software development methodology.

Illustration 33. Five principles of FDD



Source: https://project-management.com/xp-fdd-dsdm-and-crystal-methods-of-agile-development/

FDD methodology design uses following eight practices:

a) Domain object modelling
b) Developing by feature
c) Component Ownership
d) Feature teams
e) Inspections
f) Configuration management
g) Regular builds
h) Visibility of progress and results

Developers who use FDD methodology claim that this method scales more straightforward than other approaches and that is better suited to larger teams. Also, unlike other agile methodologies

it describes specific and very short phases of work, which are to be accomplished separately per feature and those includes:

a) Domain walkthrough
b) Design and design inspection
c) Code and code inspection
d) Promote to build

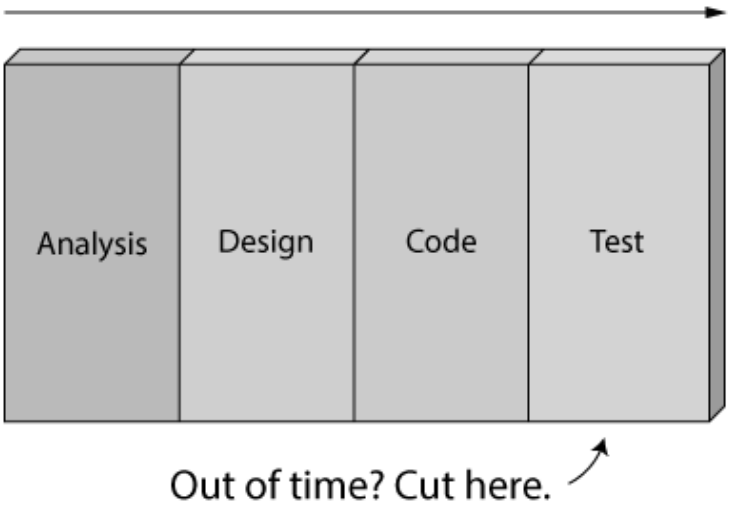
5.2. Advantages of Agile Methodologies Compared to Waterfall Model

First of all, waterfall model was a very good way of software development, especially while change costs were high. As waterfall model treats its phases very discretely and as change costs are lower on a daily basis there are couple of things which define waterfall method a poor choice.

5.2.1. Poor quality and poor visibility using waterfall model

As a project starts to run out of time and money developers are approaching to its last phase – testing. This means that any projects, good or bad, are forced to cut testing developed software, which also has impact on delivering the software. Developers cannot be sure what important things are to change, if there are all asked requirements, if customers will be satisfied. The result of getting out of time and money budget can lead from very bad software not in use to throwing away the whole product because there is no right audience who will use it. Also, the working software is not produced until the very end, so developers never really know where they are exactly on project. The last 20% of project can take up to 80% of developing time.

Illustration 34. Poor quality and poor visibility

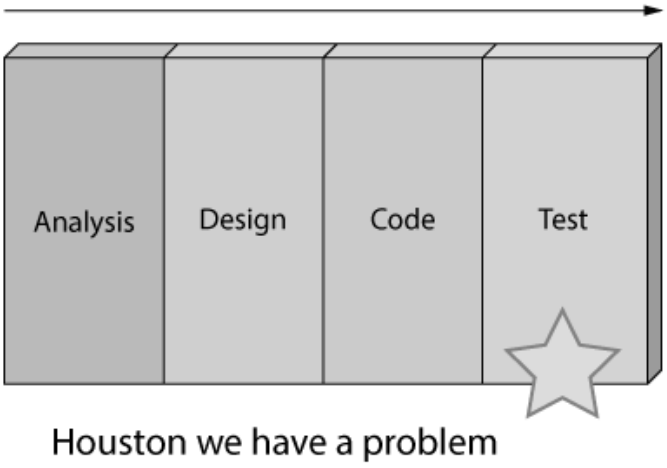


Source: http://www.agilenutshell.com/agile_vs_waterfall

5.2.2. Cannot handle change

Except poor quality and poor visibility, schedule is very risky because as a developer you never know when or if you will finish on time. Technical risks and product risks are very common just because developers cannot test their software until the last phase and they do not know if they are building the right thing until is too late to make any changes.

Illustration 35. Cannot handle change



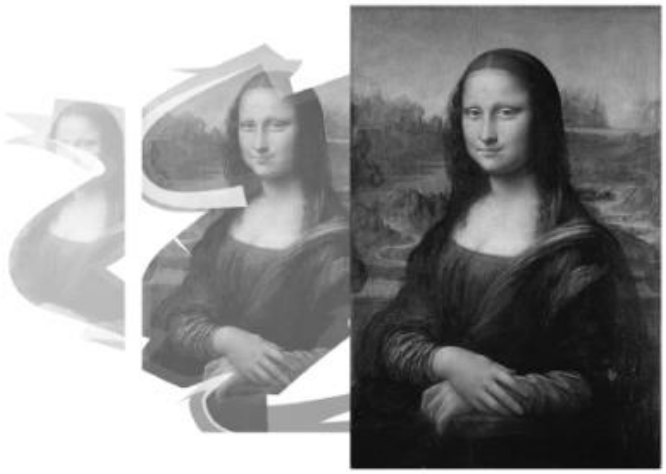
Source: http://www.agilenutshell.com/agile_vs_waterfall

5.2.3. Continuous activities

As waterfall model follows stage by stage, agile approach does not follow exactly from the first stage to the very last one. In fact, stages are continuous, which means as long as there are features to build or to deliver them, they will keep happening until project if perfectly

developed. Also, development is iterative. This means that developers are starting with something very simple, and step by step, or like drawing line by line, are making something marvellous.

Illustration 36. Iterative development



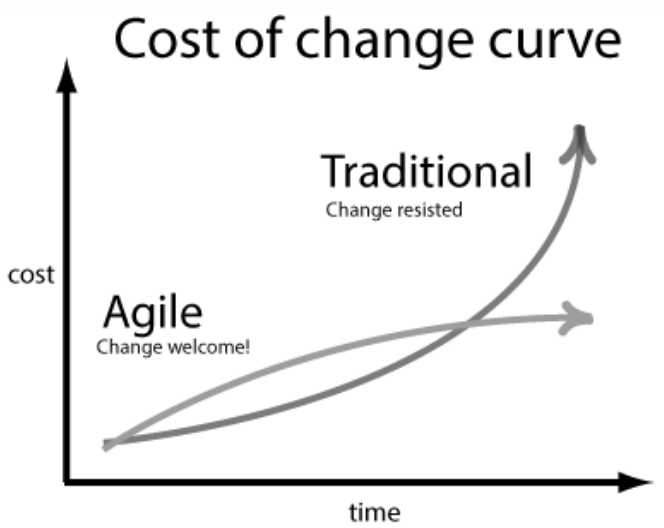
Source: http://www.agilenutshell.com/how_is_it_different

Planning is very adaptive using agile approach. In waterfall model, using one line, one plan, until very end and then coming back to early phases where changes are necessary. In agile, whenever change is required change happens thanks to all time communication with the final client.

5.2.4. Requirements can change

As above was mentioned, using traditional, waterfall model changes can be very painful timely and costly especially in late phases. Using agile methodologies, developers actually are making changes all the time. Next illustration shows who costs of change grow in time unit.

Illustration 37. Cost change growth in time unit



Source: http://www.agilenutshell.com/how_is_it_different

6. Conclusion

We can see that developing process differs from project to project. There is no right or wrong choice between these methodologies. Every methodology takes a lot of time, money and most importantly communication with developers, clients and potential users. Which methodology developers should use depends on how big the project is, and how many people will work on the project. The ideal methodology for developing ‘’eVUŠ’’ application was traditional methodology, because it was a very simple application which calculates points based on user input, it was not financially demanding, and one developer worked on the project. However, if it is not clear which method you should use, think about how big the project is, consider it with the client all the time, write down how much time each phase will take. Using one of the agile methodologies would be the right choice most of the time, because you are always changing

the plan and requirements while developing and there is always a part of delivered software for testing while using waterfall model, you cannot be sure what is happening and if final software will be the right one as it was asked for until the very end of project.

References

1. Android, https://www.android.com (visited: 11.05.2018.)
2. Android authority, The history of Android OS: its name, origin and more https://www.androidauthority.com/history-android-os-name-789433 (visited: 11.05.2018.)
3. Who Made That Android Logo, The New York Times Magazine, https://www.nytimes.com/2013/10/13/magazine/who-made-that-android-logo.html (visited: 23.05.2018.)
4. Android History and Versions, https://infograph.venngage.com/p/127159/android-versions (visited: 23.05.2018.)

5. Android version market share distribution among smartphone owners as of in February 2018, The Statistics Portal https://www.statista.com/statistics/271774/share-of-android-platforms-on-mobile-devices-with-android-os (visited: 30.05.2018.)
6. Software Development Methodologies, IT Knowledge Portal, http://www.itinfo.am/eng/software-development-methodologies (visited: 18.07.2018.)
7. Waterfall Model: What Is It and When Should You Use It?, Airbrake blog (visited: 20.07.2018.)
8. Agile In a Nutshell, http://www.agilenutshell.com (visited: 06.08.2018.)
9. Agile Methodologies, VersionOne, https://www.versionone.com/agile-101/agile-methodologies (visited: 09.08.2018.)
10. What is Scrum?, https://www.scrum.org/resources/what-is-scrum (visited: 09.08.2018.)
11. Bringing Scrum to Education, Learning Facilitated http://www.learningfacilitated.com/2016/04/bringing-scrum-to-education (visited: 11.08.2018.)
12. XP, FDD, DSDM, and Crystal Methods of Agile Development, project-management.com, https://project-management.com/xp-fdd-dsdm-and-crystal-methods-of-agile-development/ (visited: 16.08.2018.)
13. MoSCoW Rules, Forbes, https://www.forbes.com/sites/alastairdryburgh/2015/08/19/moscow-rules/#2036ab3a2910 (visited: 20.08.2018)
14. What is DSDM and the 8 principles, agilekrc.com, https://agilekrc.com/resource/168/what-dsdm-and-8-principles (visited: 26.08.2018.)

Appendix: Illustrations