UNIVERSITY OF OULU

FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING

Markus Heino
Seppo Pakonen
Jaakko Ikäheimo
Mohammed Al-Ani

Notification management

Applied Computing Project 1
Computer science degree program
11/2016

PURPOSE OF THE PROJECT

The purpose of this project is to offer an application that helps smartphone users in their everyday lives. Our main goal is to ease stress of our users by delaying their notifications and give them a chance to fully focus on what they want without getting any disrupting notifications. That is the reason why we are designing this application, which lets notifications to be delayed according to the user's configuration.

During the project we want to push our limits as a group and individuals and by doing that offer as much as we can to our potential future customers. At the same time we want to scientifically study the user response of processing disruptive notifications of their smart devices, to increase our understanding of smartphone usage behavior.

While notifications keep users informed and engaged with events focused on mobile applications, they do not have the same importance level to the user. Also, the timing of a notification is a factor in the perceived importance. A notification has been proven to be attended 12 times more likely if it has any kind of interactivity associated with it. [1]. Delivering a notification immediately after finishing a phone call make users to react to them faster. Also transitioning between physical activities have been considered more positive situation to deliver notifications [2].

In one study, interviewed participants revealed that delaying group of low priority notifications until a specific time of day is a desired feature [1]. However, classifying content and context information of notifications have shown to lead to more accurate predictions of interruptibility of users, instead of user-defined rules. [3]

We are implementing the feature what previously mentioned participants desired. The emphasis in the user experience of our application is to allow the most effective processing of notifications in the most user friendly way.

In the process of development we use a prioritization technique called the MoSCoW method, which allows the dynamic planning of the implementation. We gather user feedback to determine the (minimum) requirements. Collecting sufficient user feedback is challenging, but we will put the effort to it in the early phase of the development to gain as much information as possible.

GROUP MEMBERS

Name	Student number	Email
Markus Heino	2433882	Markus.Heino@student.oulu.fi
Seppo Pakonen	2265199	mail@sppp.heliohost.org
Jaakko Ikäheimo	2380964	j4sk4@hotmail.com
Mohammed Dalanny	2510129	mohammedalanny@gmail.com
The Project Manager	Heino Markus	Markus.Heino@student.oulu.fi

TABLE OF CONTENTS

PURPOSE OF THE PROJECT
GROUP MEMBERS
TABLE OF CONTENTS
GLOSSARY
1. DESIGN PROCESS....6
2. STATE OF THE ART....7
3. SCENARIOS AND USE CASES....8
4. REQUIREMENTS....14
5. SYSTEM DESIGN....15
6. INTERFACE DESIGN....17
7. USER COMMENTS....22
8. ANALYSIS....24
9. RISK ASSESMENT....25
10. REFERENCES....26
11. CONTRIBUTIONS....27

GLOSSARY

Push notification	A push notification notifies the user of new messages or events in the application. The devices might show an icon and a message in the status bar. Tapping the notification opens the main view of the cause.
The MoSCoW method	This is a prioritization method, which has four categories for requirements. Categories are *must have*, *should have*, *could have* and *would be nice to have*.
The AWARE framework	AWARE is an Android framework dedicated to instrument, infer, log and share mobile context information, for application developers, researchers and smartphone users. AWARE captures hardware-, software-, and human-based data.
Notification filter	Notification filter is an app from Google Play-store that is similar to ours.

1. DESIGN PROCESS

The group has been divided into the user interface team and the background system team.

The user interface team has created illustrations of menus for the application. They estimate different ways to show the most important aspects of the system while keeping the ease of use and the current state of smartphone applications in mind.

The background system team has focused to the system design and how to allow complex use cases later without making major changes to it. They have been reading publications about other similar studies and searching existing applications. Feedback from other team members and the supervisor has actually caused the design to be simplified.

The user interface design has been done using Balsamiq Mockups 3 software. Simple prototypes have also been made using HTML, CSS and Javascript (Jquery). These prototypes were shown for some friends and family members to get more opinions about the layouts and the simplicity. All comments and tips have been more than helpful and these were taken into consideration when designing the final version.

Together the group has been negotiating about the connections between the user interface and the system. Many practical issues have also been as subjects of discussions.

After negotiations with the group we were able to combine the user feedback and our ideas into one. As a group we came to a result that, user interface has to be simple, but the system behind it does not need to be. While the detailed features of the system are being found in the implementation phase, we might want to test them via the user interface. We concluded that advanced features would be available in a separate menu.

The coarse system design is handled in a following chapter, but the more detailed information will be available only in the following implementation document.

2. STATE OF THE ART

In the literature, many studies cover notification interruptions as a source of disruptions [4,5,6]. Users are susceptible to interruption overload. Waiting of an opportune moment is also a social behavior as interrupting a visibly concentrating person is considered rude. The interrupted user performs slower, experiences annoyance, anxiety and feels that the task is more difficult to complete [5,6].

An interruption might also have a monetary implication. Getting interrupted while being focused to an application has shown to be key influencer of expected costs. [7] The interruption diverts cognitive resources to the interrupting task and reallocating resources to the suspended task becomes increasingly difficult. [8]

The information about user's attention could be accessed using additional sensors. Microphones and gaze-tracking systems can be used to build probabilistic attention models. [9] In typical office environment, a workspace camera and recognition of the presence of other people leads to high accuracy of the interrupt estimation. [10]

The general processing of notifications have been available in only in recent years. In Android, the NotificationListenerService class has been added in API level 18 (July, 2013). In iPhone, the similar feature is included in the Apple Notification Center Service, which has been available since September 2013.

Several notification manager applications have also been covered by software patents [11,12]. That might cause difficulties in exporting the application to USA. Some of those patents might cover our application entirely, but as an implementation to benefit users from getting overloaded by interruptions, our application is very unique.

The market place offers various similar solutions already. The most common feature is to allow blocking notifications of an application. Other features might include the customization of the sound and vibration or keeping the log of all notifications. Upcoming version of the Android operating system will also have a similar built-in feature.

3. SCENARIOS AND USE CASES

As a prelude to use cases, one user story might be, when Mary have been very annoyed about notifications during school. All she wants is a one click solution to fulfill her need for focused learning. She doesn’t want to go through all settings in all apps to silence all useless notifications. She thinks it is time for action.

The “sunny day” use case is when a consumer wants to delay his notifications while he is working. He goes to the Play-store and searches for notification manager. He finds our application and installs it. He runs the application, set his working hours and clicks 'delay'. All notifications are then delayed until his working day ends.

The other use case is to filter working notifications after work. The steps are same as in previous, but instead of working hours he sets evening hours. Also he might want to customize settings and add messages from his employer and coworkers to be filtered.

A religious person with regular praying times might want to delay all notifications during praying time. He just adds multiple times and sets 'delay all'. Also a person going church regularly might want to do that.

The failing use case might be, when the user tries to delay all notifications during his working day, but notices that some important notifications have been delayed, which shouldn't. Also the working day of the user might end in different times and the time for notifications might be wrong too often. Some users might expect the more intelligent predictive opportunistic notification pusher and they would not be impressed about our application.

Use Case ID:	**1**
Use Case Name:	**Set notification filter by time**
Actors:	Primary actor: User
Description:	Setting the time of day and days of week to delay notifications
Preconditions:	• Application must be installed • Main screen must be active
Post conditions:	Success condition: all notifications are delayed while time matches the given time.
Normal flow:	1. User adds new filter setting (optional) 2. User sets the beginning and end time of the day 3. User selects the days of week 4. User adds multiple times of week (optional) 5. User selects 'Filter options' to select what to delay (optional) 6. User selects 'On' to enable delaying
Alternative flow:	5a. I. User selects 'Select by Category' II. System shows list of categories of notifications III. User selects e.g. 'Communication' 5b. I. User selects 'Select from notification log' II. Systems shows all notifications what has been pushed III. User selects past notifications from the log 5c. I. User selects 'Select application' II. Systems shows all installed apps III. User selects applications 5d. I. User selects 'Select contacts' II. System shows all contacts from phone and instant messaging apps III. User selects persons from the list

Use Case ID:	**2**
Use Case Name:	**Set notification filter by location**
Actors:	Primary actor: User
Description:	Setting a map location to delay notifications
Preconditions:	• Application must be installed • Main screen must be active
Post conditions:	Success condition: all notifications are delayed while user is at the given location.
Normal flow:	1. User adds new filter setting (optional) 2. User selects 'delay by location' 3. System shows the map 4. User selects the area in the map 5. User adds multiple locations (optional) 6. User selects 'Filter options' to select what to delay (optional) 7. User selects 'On' to enable delaying

Use Case ID:	**3**
Use Case Name:	**Set notification filter by automatic opportunes**
Actors:	Primary actor: User
Description:	Probabilistic model of better user attention is being calculated from user data and that is being used to delay notifications to high attention times.
Preconditions:	• Application must be installed • Main screen must be active • Enough initial user data must be available
Post conditions:	Success condition: all notifications are delayed while user's attention has been estimated to be low.
Normal flow:	1. User adds new filter setting (optional) 2. User selects 'delay by low attention' 3. System hides the time of day widget or the map 4. User selects the method of automatic attention estimation (optional) 5. User selects 'Filter options' to select what to delay (optional) 6. User selects 'On' to enable delaying.

Use Case ID:	**4**
Use Case Name:	**Start instantly delaying of notifications**
Actors:	Primary actor: User
Description:	Enables the delaying of all notifications instantly from a single button.
Preconditions:	• Application must be installed • Main screen must be active or shorcut button must be installed to the Android desktop, which is visible.
Post conditions:	Success condition: all notifications are delayed instantly
Normal flow:	1. User presses 'Delay all now' button. 2. System instantly begins to delay all notification 3. The application shows 'Active' text.
Alternative flow:	1a. I. System is showing the Android desktop screen. II. User press pre-installed 'Delay all now' shortcut button. III. System instantly begins to delay all notification. IV. The button in the Android desktop screen changes color to red.

Use Case ID:	**5**
Use Case Name:	**Stop instantly delaying of notifications**
Actors:	Primary actor: User
Description:	Disables the delaying of all notifications instantly from a single button. Also any delaying by other settings is instantly stopped.
Preconditions:	• Application must be installed • Main screen must be active or shortcut button must be installed to the Android desktop, which is visible. • The application is currently delaying notifications.
Post conditions:	Success condition: delaying of notifications has been stopped, and all delayed notifications are being shown.
Normal flow:	1. User presses 'Stop delaying' button. 2. System instantly stops all delaying 3. The application shows 'Inactive' text. 4. All delayed notifications are being shown.
Alternative flow:	1a. I. System is showing the Android desktop screen. II. User press pre-installed 'Delay all now' shortcut button, which has red light on it. III. The button in the android desktop screen changes color to green.

Use Case ID:	**6**
Use Case Name:	**Release all delayed notifications without stopping delaying**
Actors:	Primary actor: User
Description:	Shows all delayed notifications and empties the list after it.
Preconditions:	• Application must be installed • Delaying of notifications is active • 'Show delayed' shortcut button have been installed to the Android desktop.
Post conditions:	Success condition: all delayed notifications have been pushed normally.
Normal flow:	1. The 'Show delayed' shortcut button at the desktop has a small number on it, which tells how many notifications have been delayed. 2. User press the 'Show delayed' button 3. All delayed notifications are pushed and shown normally. 4. The 'Show delayed' button shows now that zero notifications have been delayed.

Use Case ID:	**7**
Use Case Name:	**Configure a daily notification free period**
Actors:	Primary actor: User
Description:	User modifies the begin or the end time of the notification free period for current day.
Preconditions:	• Application must be installed • Main screen must be active
Post conditions:	Success condition: the time of the notification free period have been updated.
Normal flow:	1. System shows the default view, which includes widgets for setting the begin and the end times. 2. User updates the begin and/or the end time. 3. User unsets the 'Selected days' option button. (optional) 4. User unsets the 'Customized filter' option button (optional) 5. User press 'On' button if it is not already active.

Use Case ID:	**8**
Use Case Name:	**Asking user feedback**
Actors:	Primary actor: User Support actor: Developer team
Description:	Asking user feedback about the application
Preconditions:	• Application must be installed
Post conditions:	Success condition: User has filled our survey.
Normal flow:	1. The request to fill survey is showed to user as push notification. 2. User presses the notification and attends it. 3. System shows the survey form. 4. User fills the form and press send. 5. System shows 'Thank you' message.
Alternative flow:	4a. I. User press 'Not now' button. II. Systems hides the survey form

4. REQUIREMENTS

Functionality	The interface should provide robust means to enable delaying of notifications by user indicated time, by geographic location or by automatic estimation. Network connected users should be able to be requested to fill surveys.
User interface	The interface should include widgets to modify the beginning and the end of the time of day and days of week. It should have a map widget for setting geographic location and a option widget to allow automatic estimation of the notification free time. The menu for selecting matching notifications should include different ways to select matching notifications.
Usability	The user interface should be consistent, intuitive and aesthetically pleasing.
Security	The feedback system must be anonymous and the connection to our server must be encrypted.
Management	Our group have meeting with our supervisor once in every two weeks. We are using a group chat to communicate.
Standards compliance	The application must be in compliance with the Google Play-store terms of service.
Portability	The code must be written entirely in Java to use it with other architectures than ARM.

5. SYSTEM DESIGN

The main task is to intercept the normal way of flowing of notifications and use our configurable processing instead.

Notifications are being analyzed and classified. All information about the notification is collected and stored structurally in the memory.

The classified notification is then being matched to existing settings. If it does not match, then it will be forwarded and shown to the user as regularly. If it does match, then it will trigger the processing according to the setting.

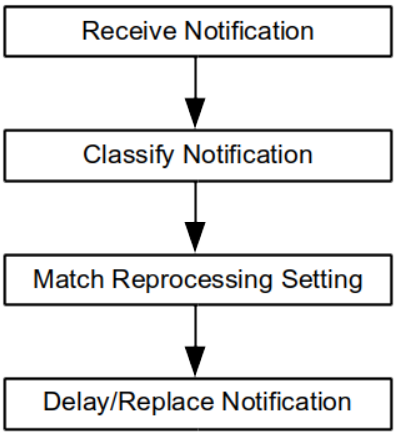


Figure 1: the sequence of the notification handling

The whole application will be running from event based function calls instead of a fast loop, which is found e.g. in many games. If there is a need to call some function periodically, then scheduled system calls will be used.

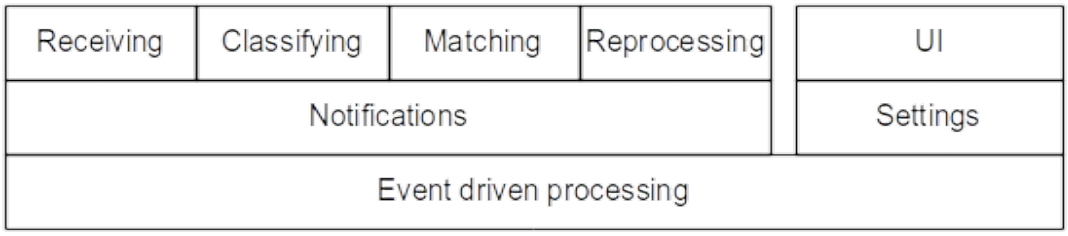


Figure 2: the system hierarchy

While the application itself does not require an internet connection, the user feedback system requires. Gathering the evaluation data is one of our primary objectives, and it requires a separate server and database connected to the Internet.

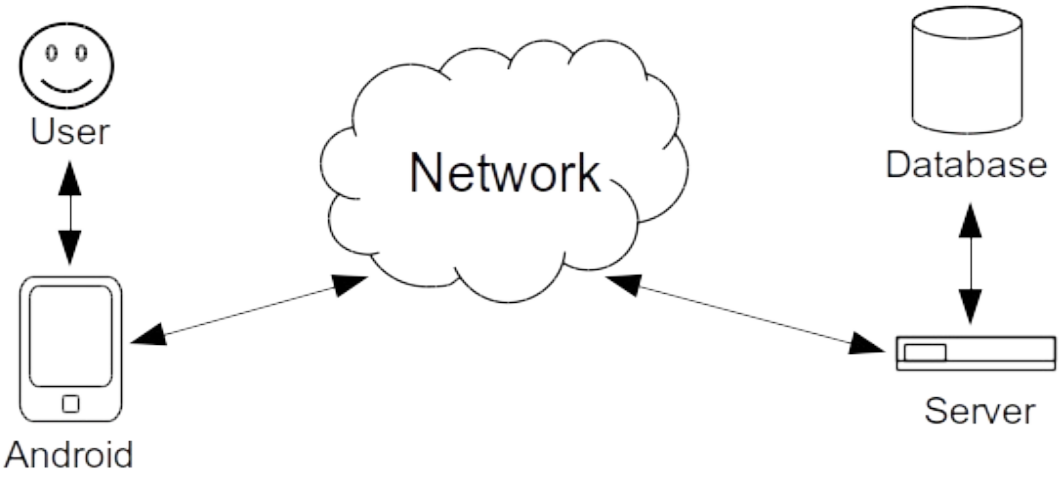


Figure 3: the network flowchart

6. INTERFACE DESIGN

The interface of the application is designed to be as simple and clear as possible so it is accessible to as many users as possible. The structure is also made to be simple and it is formed by two screens: the main screen and the application screen.

The main screen of the application consists of two main parts: the toolbar and the main container. The toolbar only contains the name of the application and a button to access help menu which is detailed later in this chapter. The main container contains application's main functionality which allows user to modify and change notification settings. As the first element main container has delay and date modifiers which allow user to change the amount of delay they want to have between notifications and dates when the notifications should be delayed. In addition, the main container has a dropdown button for selecting one of the default delaying options: 30 minutes, 1 hour or 2 hours. The notification delay on/off buttons are switched if the user wants to have notifications delayed or not. The last element for the main screen is the button for app menu which is fully described later in this document.

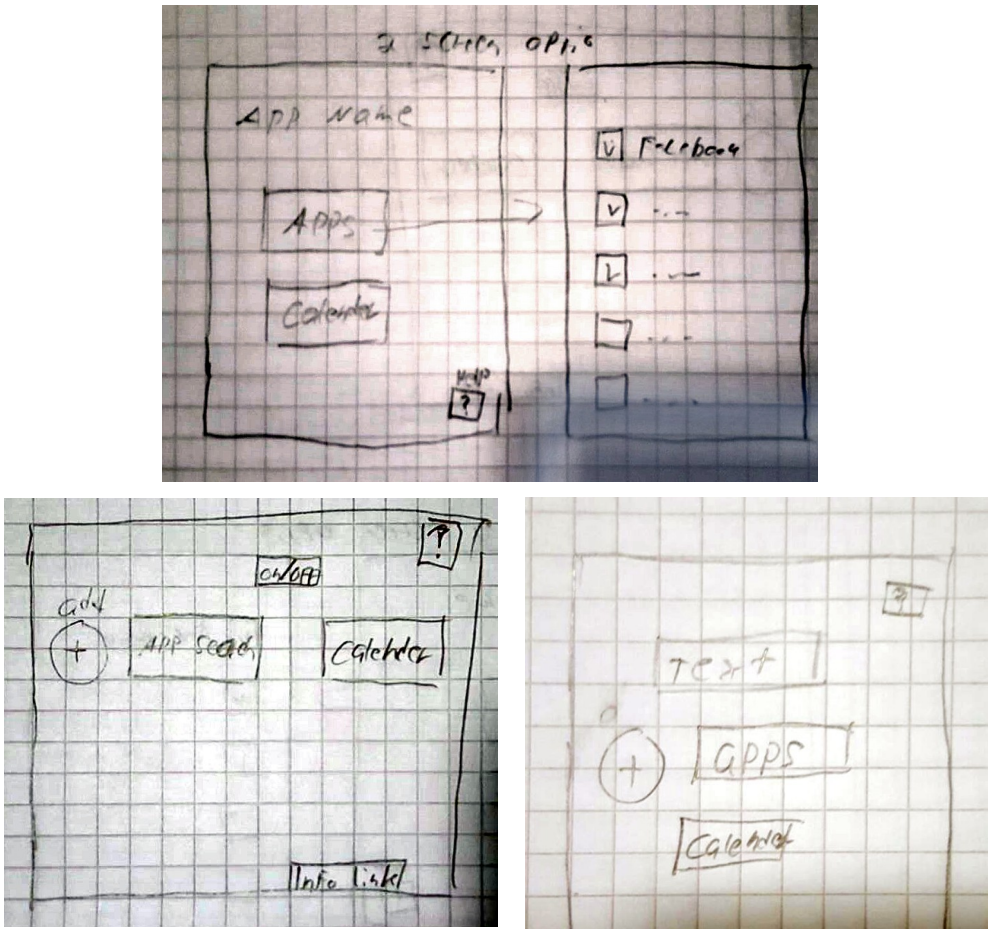


Figure 4: user interface sketches

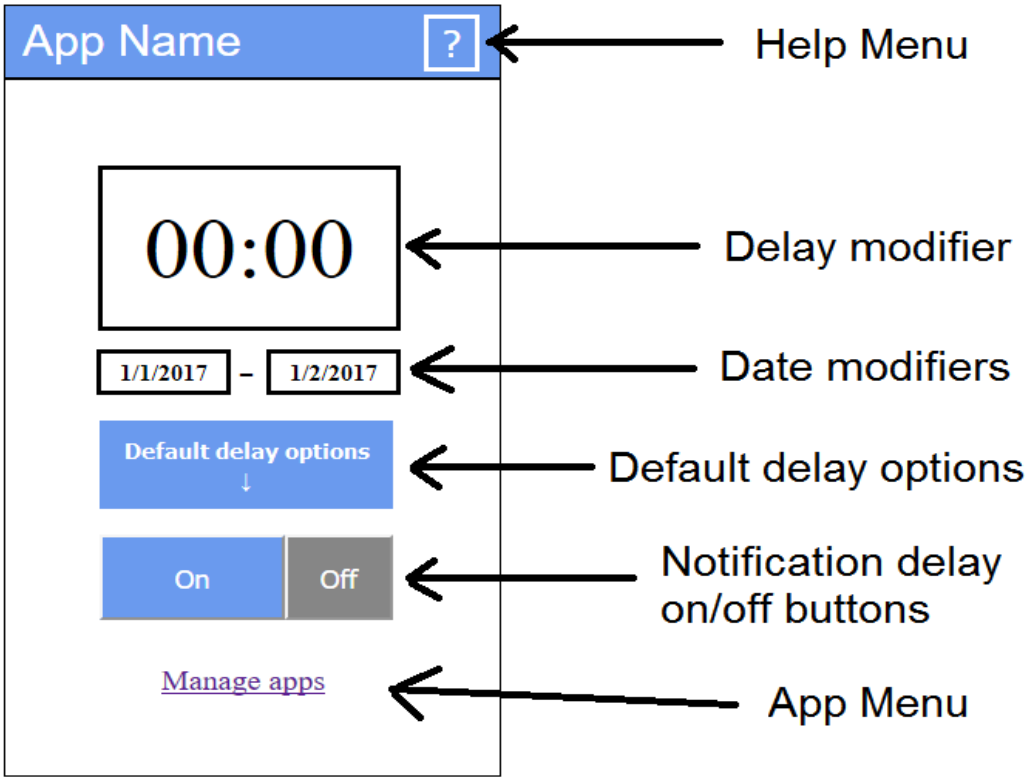
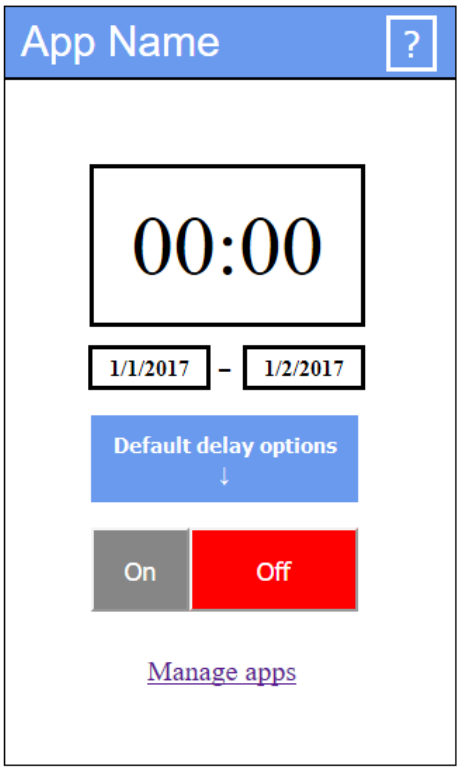
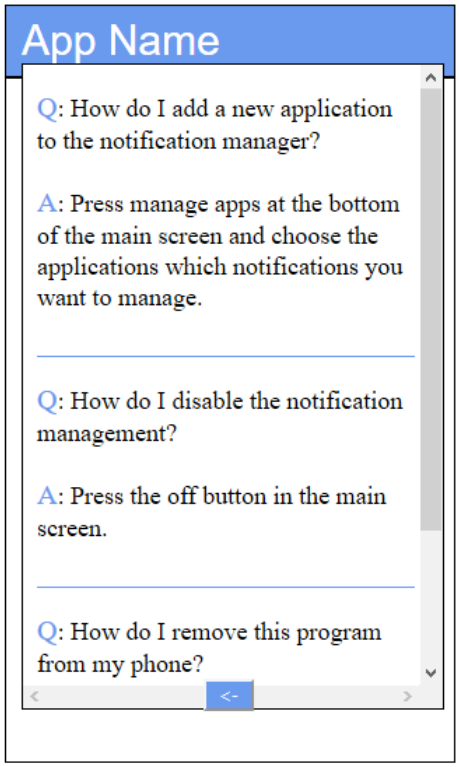


Figure 5: the main screen layout and explanations

a)



b)



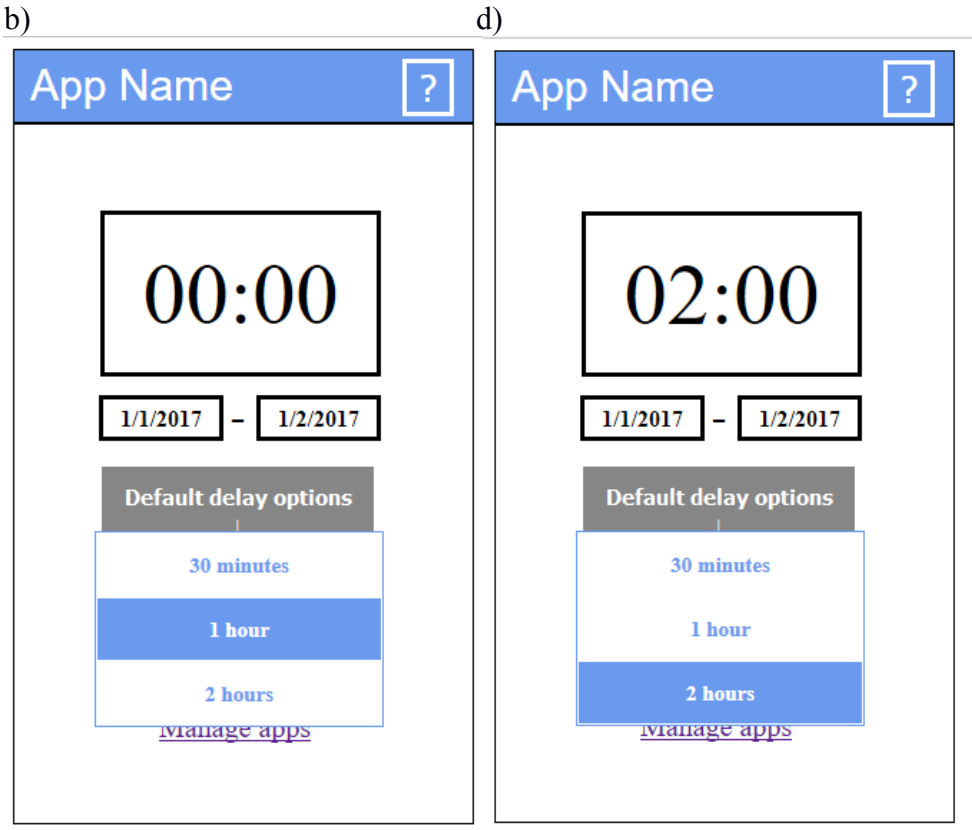


Figure 6: different states of the main screen: a) notifications off b) help menu c) default delay options dropdown-menu d) selecting default option

The application selection screen is very simple and contains only three main elements. The most important elements for this selection screen are the application selectors which control the applications user wants to enable or disable for the delaying. Disabled applications are displayed with white background and blue text, and enabled applications with blue background, white text and checkmark in front of the names. The arrow located to the left in the toolbar takes the user back to the main screen and the question mark works similarly to the one located in the main screen.

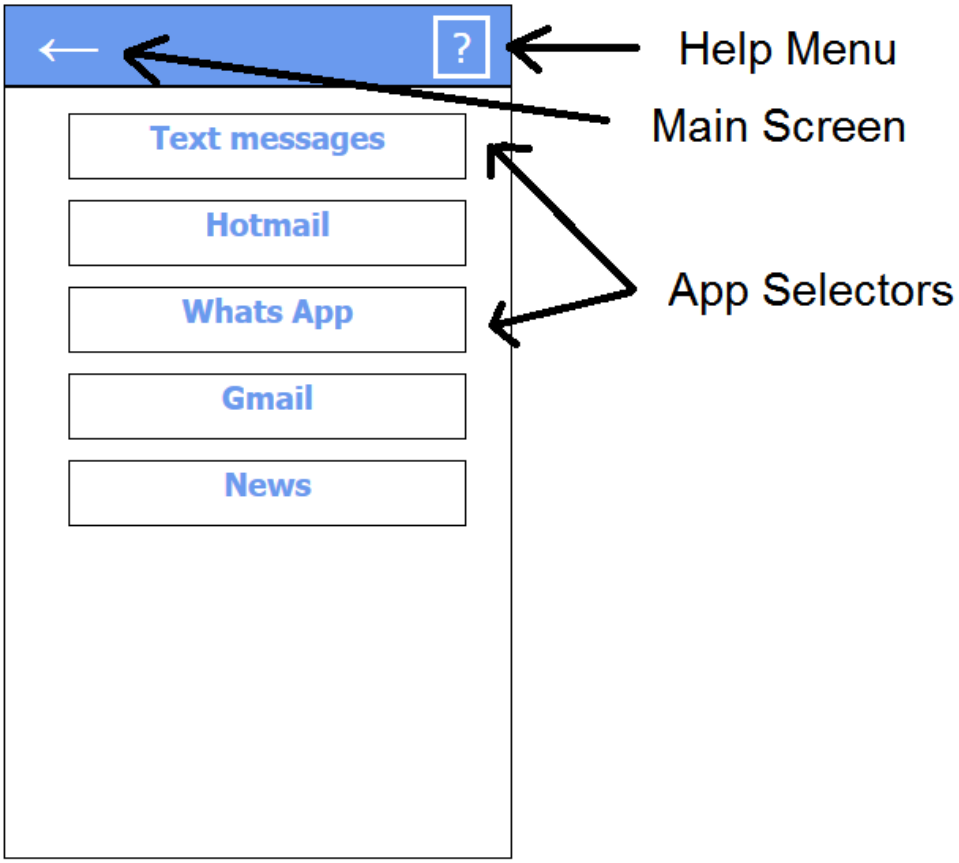


Figure 7: the app screen layout and explanations

The help menu for the application is a simple pop-up window which displays the most common problems and answers to them for the user. It is accessible through the question mark button in the toolbar in the both main and application selection screen. User can scroll down the help menu for more questions and answers. When user wants to exit the menu there is a button with an arrow in bottom of the menu which enables user to exit whenever it is wanted.

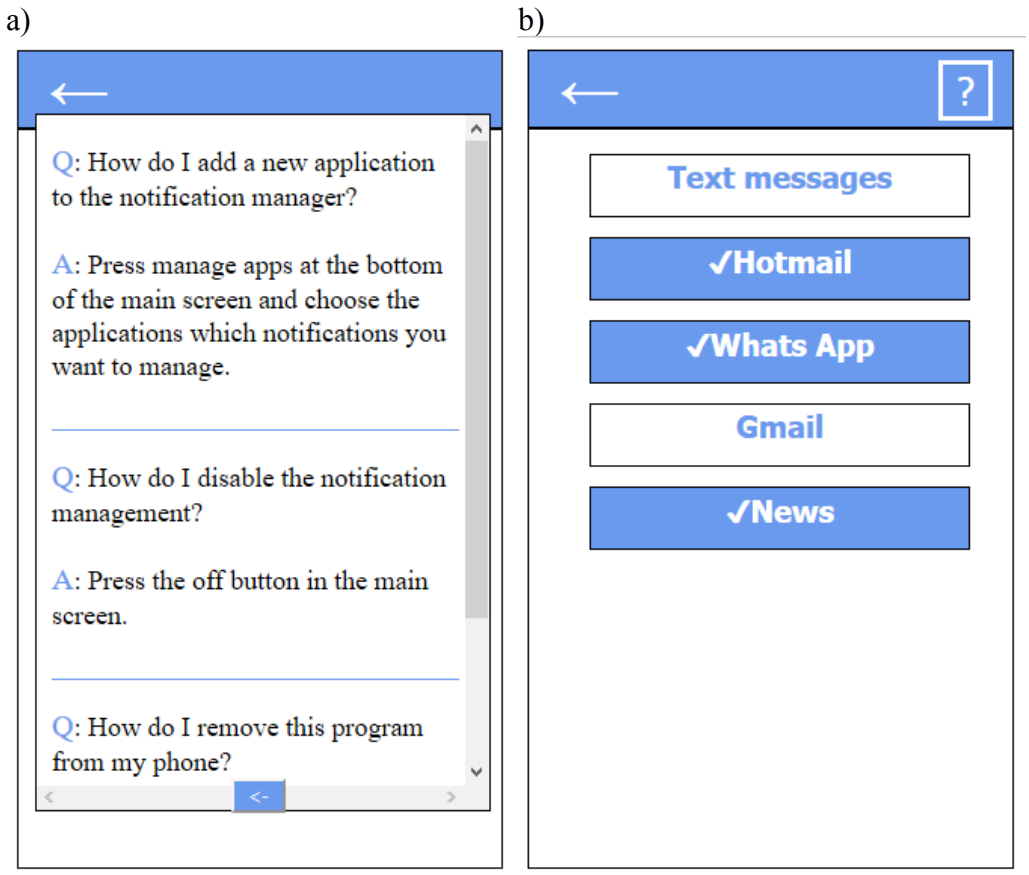


Figure 8: different states of the app screen: a) help menu b) some apps have been selected

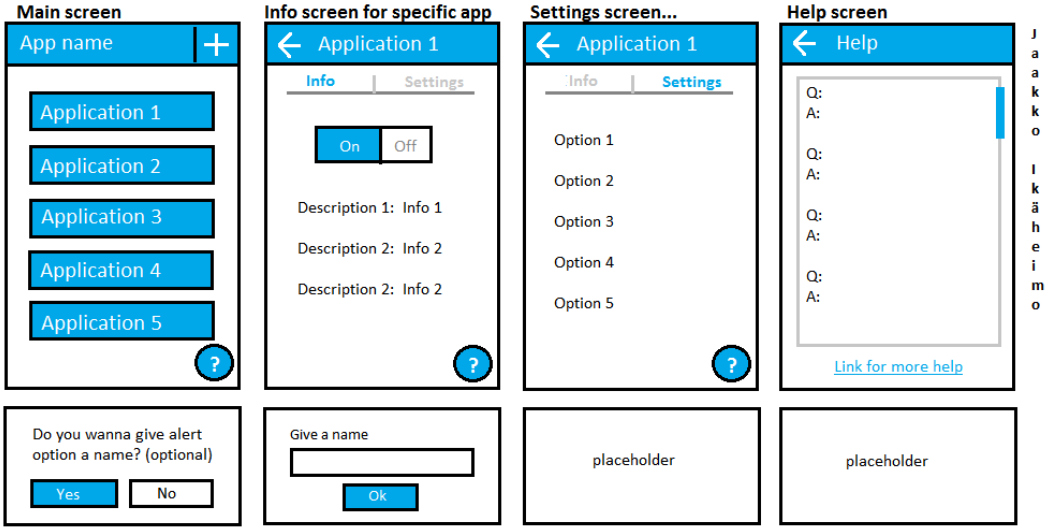


Figure 9: Some old mock-ups for the interface

7. USER COMMENTS

To fully understand our future customers and their needs, we asked questions from our potential customers. The purpose of these questions was to give us more information about what kind of features users wish to have and what options the app should have to satisfy the needs of our clients.

Conversations with every potential client took only around 10-15 minutes to keep answers neutral. The answers pointed to the most important features in our app instantly, but they can also inspire in our future research.

Questions we asked our study participants:

- *"What you think about our layouts? Is there too much in the screen, should there be more and is it enough simple?"*
- *"What could it be (referring to layouts here) colors? Or maybe the shape of our boxes "?*

- *"If you had an app that could delay Facebook or Emails for a time, at what time would you use this? (referred to examples like work and school)".*
- *"Do you think this could help you to focus better on your workday or maybe in a meeting, if you couldn't be disturbed by your phone unless it is an emergency?"*
- *"Which apps would be most important to delay and why these?"*
- *"Do you have anything you would like to add, now that you have a full view of what you could have"?*

After feedback here are the main points our potential clients had:

- *"It would be actually very nice to have an option to delay my Facebook , Email, WhatsApp, notification when I go to class, but I'm worried that I won't remember to turn it back on"*
- *"The settings I make in the app should be saved so I don't have to change them every time"*
- *"App should be easy to use, like it could have a button to turn it on and off, maybe widget ?"*
- *"Delaying for an hour or to only and then the app would turn off by itself would be nice so I could just select "delay for 2 hours" and that's all"*

It was hard to squeeze anything during the conversation from the layout or how it could be better. In this area people didn't really have ideas on it. Some answers covered that maybe you could use green/red coloring on boxes, but couldn't focus it better than that. Even after giving some examples answers were somewhat "I don't know" or "Yeah I think that would be better" but the answer sounded a little unsure. This area is where we do lay a lot on our own expertise and research from internet and after watching the layout Notification Filter had.

The conversations once again proved that people have a lot of opinions and they can vary from one direction to another. Getting feedback is always crucial and therefore it should be always done no matter what kind of project you are doing. These comments do open a whole new view for your project. As a designer this is crucial for the project no matter are the answers negative or positive. As a group we had a method “If the feedback is positive, well that is good”, “In the other hand if the feedback is negative, that should be, what will arouse interest in whole group”. Those negative feedbacks are what make the group better, and this way the project can progress forward without having to think are we going into the right direction. All the answers were taken into consideration during design progress. Especially the feedback for fast-delaying option and how it should work. Feedback gave us more to think and more option how we can keep improving the design.

After research analyses, we are more enlightened as a group of our potential customer needs. We have a full picture of the features people want and how they should work in order to satisfy our clients in a best way possible.

8. ANALYSIS

The annoyance of push notifications is scientifically known subject. The response time to notifications have been measured [1,3] and the subjective experience of the user have been polled [2]. Our application will do both of those measurements.

There are some similar application in Play-store [13,14]. Our application has the unique feature of delaying notifications.

Notification Filter [9] was the first similar application we found. The first impression after opening the app is that you have to be using this app and you had to modify your apps. This is not what we want at all. We want that the user has an option to use the app if she/he wishes. There is no need to use it all the time and keep consuming your battery for nothing. Also the notification filter had three steps that you had to do before using the app. We want offer max 3 different sides. The front page includes easy access, on/off, time for how long to delay. Second page is for settings. Which apps you can delay and the checking is easy as it could be as shown in interface design. The third page is for more detailed settings for certain apps. This page will be for more advanced users so not a mandatory for user.

As overall we want the user feel like this is just one app to help her/him in everyday life without thinking what kind of data will be this app collecting from me and how much it consumes my battery. We want to offer something that is easy to use, don't feel mandatory and can be used only when you want!

This project is a combination of known parts, both in marketing and scientific perspective, which makes it a good design.

9. RISK ASSESMENT

Brief	The amount of user feedback is not enough
Likelihood	Common
Impact	Minor
Preventive action	Making a working beta version in early phases of the development, and outreaching friends and relatives as test users.
Corrective action	We focus to data which doesn't require user interaction.

Brief	Functionality is not easy enough
Likelihood	Common
Impact	Moderate
Preventive action	Gathering as much data from potential users as possible. Making the app as simple as it can be.
Corrective action	Changing functionalities after feedback to satisfy customers.

Brief	Project member leaving
Likelihood	Unlikely
Impact	Major
Preventive action	Keeping the atmosphere in a good level during the whole project.
Corrective action	Trying to change the mind of leaving person or to keep going without one (depending what is the reason for leaving)

Brief	Technical difficulties blocks implementing
Likelihood	Unlikely
Impact	Moderate
Preventive action	Making sure the whole group does have enough information and research. Helping each other the best way possible.
Corrective action	Review prioritizations. Search alternative ways to implement. Simplify the design more.

10. REFERENCES

[1] The myth of suble notifications. In UbiComp ’14 (September 2014)

[2] Investigating episodes of mobile phone activity as indicators of opportune moments to deliver notifications. In MobileHCI ’11 (September 2011)

[3] Designing content-driven intelligent notification mechanisms for mobile applications. In UbiComp ’15 (September 2015)

[4] If Not Now, When?: The Effects of Interruption at Different Moments Within Task Execution. In CHI ’04 (April 2004)

[5] The Effects of Interruptions on Task Performance, Annoyance, and Anxiety in the User Interface. University of Nevada, 2001.

[6] Notification, disruption, and memory: Effects of messaging interruptions on memory and performance. Microsoft study (January 2001)

[7] BusyBody: Creating and Fielding Personalized Models of the Cost of Interruption. In CSCW '04 (November 2004)

[8] Disruption and Recovery of Computing Tasks: Field Study, Analysis, and Directions. In CHI '07 (May 2007)

[9] Models of attention in computing and communication: from principles to applications. In Communications of the ACM (March 2003)

[10] Predicting human interruptibility with sensors. In ACM Transactions on Computer-Human Interaction (March 2005)

[11] Temporal incoming communication notification management. Patent US 8855723 B2

[12] Systems and methods for push notification management Patent US 20150120849 A1

[13] Notifications filter. Android Play-store.

[14] Filter notifications (beta). Android Play-store.

11. CONTRIBUTIONS

Name	Markus	Jaakko	Seppo	Mohammed
Illustrations	9h	20h	3h	
Meetings	8h	8h	8h	10h
Use cases	1h	1h	17h	2h
Document	16h	8h	20h	
Research	12h	4h	13h	25h
Additional	6h	14h (coding)		10h
Total	52h	55h	63h	47h

Markus Heino

I started with mockups with Jaakko. In the first meeting we decided what kind of mockups will be the best for us and Jaakko started to modify mockups the way group decided. I started to work more on the design template and making sure template has everything it should. First a lot of research from design templates from others and then making very coarse version of my own for the group. Version had everything explained what we should have in our template and which ones are optional but from perspective mandatory for us. Template started to change after we added content on it and have been changing during the whole project. My main goals have been cover page, purpose of the project, analysis, risk assessments, mockups and making sure we have everything on the template that should be and even a bit more. Everything been proofread by every member and there for text have been changing a bit from every section.

Jaakko Ikäheimo

From the beginning me and Markus decided to work on illustration and design of the application's interface. I started to work on the prototype and Markus with research and documentation. After evaluation and some changes I came up with a complete prototype and started to work with documentation and interface design part in the template. The most important part for me in design phase has been trying to come up with a good design for interface and improving the prototype and my designing skills while doing that. I have been also trying to modify the document so it is more readable and adding some stuff here and there. You can see the prototype we used for our design on: jaskaweb.com/proto_main.html

Seppo Pakonen

Together with Mohammed we formed the background system team, which planned use cases and system design. We discussed a lot of the content before writing. I wrote partially or entirely the purpose of the project, design process, state of the art, scenarios and use cases, requirements, analysis and risk assesment. I researched and wrote all references to publications and formatted the text to thesis format. I was heavily corrected, though. Much learning was required for this paper.

Mohammed Al-Ani

Worked with Seppo intensively for the system design and the restful services. I did research on what is suitable design pattern for our notification app. Worked on the cloud9 to run the tomcat server and prepare the restful API using jersey Jax-Rx framework for the implementation of the backend. And I Did some quick research on the ways of connecting the MySQL to Jersey Worked on designing the Data Access Object DAO to make the API in the implementation phase follows the standards.