A Participatory Sensing System for Road Quality Data Acquisition

Author:
Bram van Berlo

Student Number:
s1538233

Thesis Submitted in Partial Fulfillment of the Requirements for the Bachelor of Science Degree in Creative Technology

Commissioned By the Pervasive Systems Research Group

Faculty of Electrical Engineering, Mathematics and Computer Science

Supervisor:
dr. ir. Nirvana Meratnia

Critical Observer:
ir. Hans Scholten

July 8, 2017

UNIVERSITY OF TWENTE.

Abstract

The international roughness index (IRI), a number used to analyze the overall road quality of specific roads, is obtained based on costly and subjective evaluation methods with a frequency basis that has proven to be ineffective in several harsh weather situations. Additionally to that recently emerged systems that have tried to solve this problem introduce entirely different challenges on their own. Some of these challenges include usage of the system for unintended purposes and data contamination as a result of smartphone usage inside a car while driving.

The first purpose of this study, in order to solve the challenges given in the first paragraph, was to integrate a dedicated sensor system inside a participatory sensing network consisting of several subsystems that is able to acquire good quality sensor data, process this data in an efficient manner and visualize IRI numbers for different roads. The second purpose of this study was to determine to what extent the system described in the first purpose statement is able to provide instrumental value for multiple stakeholders.

In order to determine how a dedicated sensor system can be integrated inside a participatory sensing network a technical system analysis of all the subsystems was done. The theory acquired during the technical system analysis was tested by a system design and implementation of a system prototype. In order to determine the instrumental value provision for multiple stakeholders a stakeholder brainstorm was done and based on that brainstorm several one to one interviews were held with potential stakeholders in order to determine how instrumental value could be provided to these stakeholders.

In the end it was determined that, apart from some minor pitfalls that still need to be omitted, the integration of a dedicated sensor system inside a participatory sensing network in terms of a system prototype was a great success. This success is based on several factors that are further explained in depth through the course of this bachelor thesis. The first factor is that the system acquires Good quality gps, acceleration and orientation sensor data. The second factor is that all of the subsystems have been Seamlessly integrated. The third factor is that the system provides effective and efficient local and main data storage. The last factor is that the system is able to visualize IRI numbers in a clear visualization.

It was also determined that instrumental value could be provided in several ways. The first way is through improving governmental maintenance planning and decreasing associated costs. The second way is through decreasing the likelihood of people and/or insurance companies having to spend money on car damage caused by bad road quality. The third way is by helping to improve overall safety and security of roads. The fourth way is by providing a platform for supplying data that can give scientific insight into infrastructure quality other than just road quality and big data research about data heterogeneity. Lastly the system also provides value to ‘user initiatives’ that are coming up with the intention to take responsibility of road quality monitoring and maintenance into the hands of the general public.

Acknowledgement

Before starting off the main content of the bachelor thesis I would like to thank several people for their contribution to this bachelor project. First of all I would like to thank Wim van der Peet for his time, effort and craftsmanship in order to help realize the system casing for the dedicated sensor system version 3. Secondly I would like to thank my family, in particular Frank van Berlo, for their help with testing the complete participatory sensing system prototype and providing feedback and support. Lastly I would like to thank the Pervasive Systems (PS) group, in particular dr. ir. Nirvana Meratnia, ir. Hans Scholten and ir. Fatjon Seraj, for their guidance during the bachelor project execution. The bachelor project would not have been possible without the help of these people.

Table of Contents

List of figures

List of tables

1. Introduction

1.1 Current situation

Road engineers use an international roughness index (IRI) number in order to analyze the overall road quality of specific roads. Based on this analysis it can be determined to what extend road anomalies occur and if road treatment is necessary [5].

The IRI, in [mm/m], is a dimensionless index number [30]. However, after inspecting some IRI number tables, it is safe to say that most IRI numbers vary between 10 (road quality is so bad that road reconstruction is inevitable) and 0 (no treatment is necessary). These index numbers are obtained based on costly and subjective evaluation methods that make use of road inspection and observation by qualified personnel using road analyzers [31].

Recent pervasive system research group investigations have resulted in a neat way of using mobile phone sensors for the purpose of analyzing driving behavior in order to detect possible road anomalies and to compute an IRI based on this information.
The sensing aspect is realized through the use of accelerometers, gyroscopic sensors and GPS contained within a mobile smartphone [5].

The road anomaly data, when multiple phone sensor system nodes are used in a complete participatory sensing end-to-end system, has the potential of providing an objective analysis tool that contributes to the formation of an IRI number with less extensive use of time consuming and costly subjective analysis methods.

1.2 Challenges and objectives

The current situation produces numerous challenges.

Firstly, in order for correct sensor data to be produced by a sensor node the phone is required to stay in a certain fixed position within the car. This is not always possible since the smartphone might be used for multiple purposes in an automotive environment.

Secondly, in order for the system to produce good quality sensor data, e.g. a dataset that is large enough for outlier filtering and that might result in better road anomaly classification results, a system is required that contains multiple participating sensor nodes working together in a network delivering data to a central server.

Thirdly, research has to be executed in order to determine the value of data obtained by an end-to-end participatory sensing system for the computation of an IRI number and potential value among multiple stakeholders such as car owners, car companies, governments and road quality control companies.

In order to overcome the challenges above several objectives have to be reached.

The first objective is to overcome the challenges by building a participatory sensing system that acquires good quality sensor data, processes this data, visualizes IRI numbers and predicts possible road anomalies for different roads.

The second objective is to deliver instrumental value to multiple stakeholders through a data mine[1] the stakeholders can use for data mining purposes.

1.3 Research questions

The following research questions and sub questions will guide the research process towards successfully solving the challenges and reaching the objectives discussed in section 1.2:

How can a dedicated sensor system be integrated in a participatory sensing system network that is able to acquire good quality sensor data, process this data in an efficient manner and visualize IRI numbers for different roads?

- How can state of the art technology be implemented in order to realize a dedicated sensor system that is able to deliver good quality sensor data?

- How can state of the art technology be used in order to automate the data acquisition process?

- How can state of the art technology be used in order to automate the data transmission process?

- What is the most effective SQL or NoSQL database structure when used in a big data application operating context similar to the operating context of the participatory sensing system in terms of ‘big data’ handling?

    - What is the participatory sensing system database operating context?

    - What type of SQL and NoSQL database structures exist?

    - How do SQL and NoSQL database structures store data?

---

[1] Data mine: a large data set that can be used for pattern discovery in order to obtain value in the form of information

    - What parameters can be used in order to evaluate the effectiveness of ‘big data’ handling?

    - In terms of the effectiveness parameters, what is the most effective SQL or NoSQL database?

- How can state of the art technology be used in order to visualize IRI numbers?

How can this particular participatory sensor system provide instrumental value for multiple stakeholders?

- In context of the participatory sensing system, what defines a stakeholder?

- What type of stakeholders are affected by the participatory sensing system?

- What type of instrumental value does the participatory sensing system provide for a focus group of stakeholders?

- How does the participatory sensing system provide instrumental value for a focus group of stakeholders?

- How can potential requirements of the focus group be used as design recommendations in the participatory sensing system?

1.4 Methodology

For each sub question in section 1.3 the following methodologies will be used:

“How can state of the art technology be implemented in order to realize a dedicated sensor system that is able to deliver good quality sensor data?”

State of the art research in wireless sensor-driven hardware platforms will be conducted in order to find a starting point for a dedicated sensor system prototype. Based on the results the platform with the most requirements met by the initial system and the lowest realization price will be picked and tweaked in order to obtain a prototype that meets all of the dedicated sensor system requirements. The prototype system accuracy will be analyzed by data measurements and improved if necessary.

“How can state of the art technology be used in order to automate the data acquisition process?”

Data acquisition will be triggered once the speed of the car is above a certain threshold. First of all the data structure requirements will be defined. Secondly literature research will be conducted in order to determine how to use GPS NMEA data in order to determine the speed of the car. Last but not least literature research will be conducted in order to determine how all GPS clock and internal processor clock timestamps can be synchronized and timestamp deficiencies minimized.

"*How can state of the art technology be used in order to automate the data transmission process?*"

Literature research will be conducted in order to determine how data can be saved in a mobile application if no data transmission is possible and what needs to happen with the data as soon as data transmission is possible.

"How effective are SQL and NoSQL database structures when used in big data applications in terms of handling 'big data', complexity of query statement execution and compatibility with machine learning scripts?"

The effectivity of database structures in big data applications will be analyzed through the use of a literature research. Based on the findings a database will be chosen that is going to be implemented as part of a participatory sensing system prototype. The database choice will be based on effectiveness parameters that are obtained during the literature research. Based on this database decision an appropriate matching local database implementation will be chosen as well.

"How can state of the art technology be used in order to visualize IRI numbers?"

Scripts created by the pervasive systems group will be used in order to process the data stored inside the database in order to obtain an IRI number for different road sections. These IRI numbers will be plotted in a data visualization created with the Google Maps javascript API.

"In context of the participatory sensing system, what defines a stakeholder?"

A literature review will be done in order to determine different stakeholder definitions. Based on this literature review the most appropriate definition will be chosen or a custom definition will be constructed.

"What type of stakeholders are affected by the participatory sensing system?"

A stakeholder brainstorm will be done in order to determine possible stakeholders for the participatory sensing system. Afterwards, based on a small stakeholder interview sample (3 interviews) and stakeholder context observations, a stakeholder focus group for the rest of this report will be determined based on a 'power-interest' grid [7]. The different roles of the stakeholders are general (general public) and professional users (scientists). The context observations will focus on the reach stakeholders have in order to attract other potential stakeholders, the amount of obstructions stakeholders can put in the way when trying to attract stakeholders and to what extent value might evoke interest among stakeholders of the participatory sensing system.

“What type of instrumental value does the participatory sensing system provide for a focus group of stakeholders?”

“How does the participatory sensing system provide instrumental value for a focus group of stakeholders?”

An instrumental value overview will be given based on the stakeholder interview results.

“How can potential requirements of the focus group be used as design recommendations in the participatory sensing system?”

Based on a stakeholder focus group a set of personas will be determined based on a framework presented in appendix H. Based on the different settings in which these personas might use the system to access instrumental value different use situations (scenarios) are determined. In these use situations standard functionality and special use situations are determined that introduce additional system requirements. These requirements will be defined and prioritized through the MoSCoW model [28]. Design recommendations based on these requirements will be given in a requirements section.

1.5 Report structure

Before the start of this chapter an abstract, acknowledgement, table of contents, list of figures and list of tables section were already introduced.

Chapter 1 introduces the research subject, gives a short explanation on how the research goals are going to be reached and lays out the structure of the report.

Chapter 2 gives a state of the art overview on road quality data acquisition and maintenance technology. The overview sections are divided into how the technology works, positive aspects in regard to the research questions and challenges in regard to the research questions. Last but not least the section concludes with an explanation about the innovation the participatory sensing system brings to the state of the art field.

Chapter 3 gives a stakeholder analysis, which is going to be used as a theoretical basis for designing the functions of the GUI and will give some information on how the participatory sensing system provides instrumental value to potential users.

Chapter 4 provides a theoretical basis for the technical aspects of the participatory sensing system. First of all the chapter introduces a state of the art sensor system analysis that will be used in order to construct the dedicated sensor system aspect. Secondly, the chapter provides details on how the data acquisition and transmission process of the system can be automated. Thirdly, the chapter provides an analysis on the effectiveness of SQL and NoSQL database structures in big data applications that will be used in order to determine the database aspect of the participatory sensing system. Fourthly, the chapter shows how the data inside the database is going to be used in order to plot IRI numbers in a graphical user interface (GUI) based on functionality diagrams.

Chapter 5 provides the actual design process of the participatory sensing system prototype.

Chapter 6 will introduce the answers to the research questions.

Chapter 7 will elaborate on the strong points and points of improvement of the research. Additionally the chapter will introduce possible further investigations and will give a short summary on some ethical and societal issues of the participatory sensing system and how future research might be able to mitigate those issues.

Lastly the appendix and references used in the document are introduced.

2. State of the art analysis

The state of the art analysis first of all gives an overview of the current technology used in order to deal with the current situation presented in section 1.1. Secondly the state of the art analysis shows recent investigations by research groups all over the globe that have resulted into smartphone sensor system applications that can be used for road quality data acquisition. Thirdly the state of the art analysis gives an overview of participatory sensing systems currently implemented in other contexts and their suitability for the context described in this thesis.

2.1 Road analyzers

In the Netherlands road analyzers are used in order to investigate the road quality of several roads. Based on the outcomes of these road analyzers an IRI number is computed and a maintenance planning is made in order to improve the quality of roads that do not have a low enough IRI number. Two road analyzer examples are given below.

2.1a ARAN automatic road analyzer

According to a report published by TNO and commissioned by Rijkswaterstaat [32], the Dutch municipality in charge of road maintenance and inspection, the current road analyzer used by Rijkswaterstaat is called the ARAN automatic road analyzer. An image of the road analyzer can be found in fig 1.

Fig 1. ARAN automatic road analyzer vehicle in the Netherlands [1]

Concept summary

According to Furgo, the company that created the ARAN automatic road analyzer, ARAN is “a highway data collection system that is capable of measuring more than 15 different data streams continuously at varying capture rates in a single pass at traffic speeds” [33].

The data that originates from the ARAN automatic road analyzer is processed into a road quality report by Furgo and send to the road municipality that requested road quality information for specific roads.

Positive aspect
Klunder et al. [32] note that the ARAN system, in the use case of the Netherlands, is an advanced system. As a result, the use of mobile road quality sensing applications contained in a car (in case of the participatory sensing system the dedicated system part) should not be used as a replacement for accurate ARAN systems in road quality monitoring [32].

Challenge
However, according to Klunder et al. [32] the ARAN system is also a costly system. Therefore the road inspection with an ARAN system is done on a one to two year basis [32]. This frequency basis has proven to be ineffective in situations where frost causes a decline of road quality in a matter of days [32]. Therefore the use of a mobile road quality sensing application that measures more frequently than the ARAN system is a useful asset next to the ARAN system for road quality monitoring [32].

2.1b Miramap wegdekscan

In addition to the ARAN automatic road analyzer there are also other road analyzers available for road quality monitoring. One example is called the Miramap wegdekscan. An image of the system can be found in fig 2.

Fig 2. Miramap wegdekscan (Note: image is from another miramap application. However, miramap states that wegdekscan uses a road analyzer with roughly similar visual appearance) [2]

Concept summary
According to miramap, the wegdekscan “uses MIRA scanners and professional thermal cameras in order to detect potholes, analyze thickness of asphalt layer, detect deviations in ground composition underneath the asphalt layer and lastly Detect air pockets and moisture contained in the asphalt layer. Afterwards the data retrieved from the system is Delivered as geo-referenced dataset with technical report.” [34]

Unfortunately no data has been published about positive aspects and challenges related to wegdekscan.

2.2 Road quality data management software

Even though the road analyzers above offer a complete solution a road municipality like Rijkswaterstaat could use road quality data management software in order to visualize road quality data and automatically compute maintenance costs. A road quality data management software example is given below.

2.2a Street Saver

Street Saver is a road quality data management software application that can be used in order to plot geographic road quality data on a map, predict different road quality indicators such as an IRI number, compute road maintenance costs based on the road quality indicators and create maintenance schedules [35]. An example image of the Street Saver application can be found in fig 3.

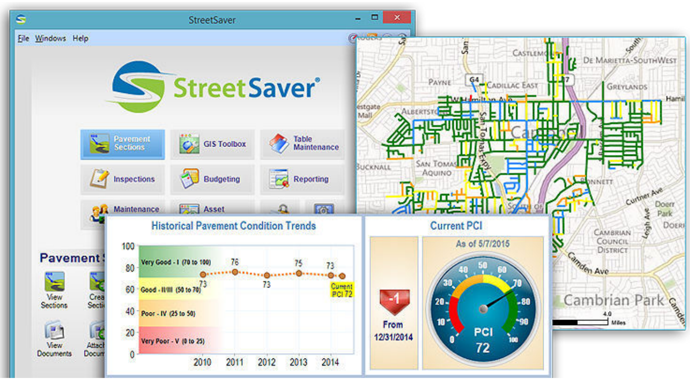


Fig 3. Street Saver application [3]

Even though this application does not come near realizing the objectives described in section 1.2. e.g. providing road quality data and classifying road anomalies from this data, it might prove to be a good starting point for user testimonials of road quality GUIs.

2.3 Participatory sensing systems

In other countries, in contrast to the Netherlands, municipalities that are in charge of road maintenance and inspection have been looking into the use of other methods, in addition to road analyzers, such as participatory sensing systems for analyzing road quality and constructing maintenance planning schemes. An example of a participatory sensing system is given below.

2.3a Waze

The Portland Bureau of Transportation (PBOT), a local road maintenance and inspection municipality in America, has been looking into the use of a smartphone application called Waze (see Appendix E). An image of the application can be found in fig 4.

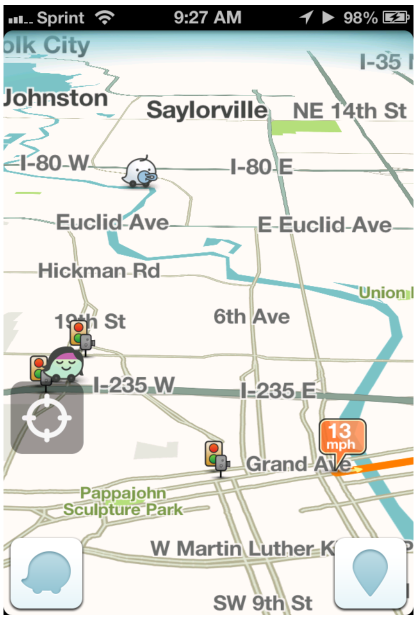


Fig 4. Waze application [4]

Concept summary

According to Waze itself, Waze is “all about contributing to the 'common good' out there on the road. This contribution is made by connecting drivers to one another through a smartphone application. As a result Waze helps people create local driving communities that work together to improve the quality of everyone's daily driving” [36].

The smartphone application is operated by “users just driving with the app open on their phone to passively contribute traffic and other road data, but they can also take a more active role by sharing road reports on accidents, police traps, or any other hazards along the way, helping to give other users in the area a 'heads-up' about what's to come” [36].

Positive aspect

As already stated in the concept summary the most positive aspect of Waze is the introduction of a way for people to contribute information in order to improve the common good on the road. As a result users through a joint effort improve the quality of everyone’s daily driving [36]. Additionally, according to a fact sheet released by Waze, the application is also able to visualize road data real-time in a GUI [37].

Challenges

First of all, according to a few sources, Waze is very prone to system use for purposes unintended by the system developers. According to an article written by Kelsey D. Atherton for Popular Science, an american science magazine, two students at the Technion-Israel Institute of Technology simulated a virtual traffic jam with help of the Waze app. The purpose of this simulation was to show how malicious Waze app users might be able to cause real traffic jams with the Waze app [38]. According to an article written by Michael Fleeman for investing.com, a financial toolbox portal, Los Angeles police chief Charlie Beck claims that “accused gunman Ismaaiyl Brinsley had used the Waze app in the days before an ambush of New York officers Rafael Ramos and Wenjian Liu on December 2014 as a lookout tool for the location of New York police officers” [39].

Secondly, according to Kevin Roose, a report for New York magazine, Waze causes people to have a high level of cognitive distraction from paying attention to road situations while driving due to the speech-to-text system Waze uses for user interaction [40]. This claim is backed up by research on Cognitive Distraction in the Automobile conducted by Strayer et al. [41] commissioned by the AAA Foundation for Traffic Safety. Strayer et al. [41] “found that interacting with a speech-to-text system was the most cognitively distracting in a car setting among several interaction methods”.

Lastly Waze does not mention anything about their system being able to acquire sensor data, either through a smartphone or dedicated system, specifically for classifying road anomalies and in order to compute and visualize IRI numbers in the Waze smartphone application.

2.4 Mobile road quality sensing applications

Next to the current implemented road quality analysis applications a lot of research is going on in other ways to analyze road quality and compute an IRI number. The first way is the introduction of smartphone applications that use built-in smartphone sensors to obtain acceleration and orientation information. Based on the acceleration and orientation information an IRI number can be computed.

2.4a RoADS

RoADS is the smartphone sensing application created by the Pervasive systems (PS) research group. The purpose of this system is to analyze driving behavior based on acceleration and orientation data gathered from smartphone sensors in order to detect possible road anomalies and to compute an IRI based on this information. A RoADS system flowchart can be found in fig 5.

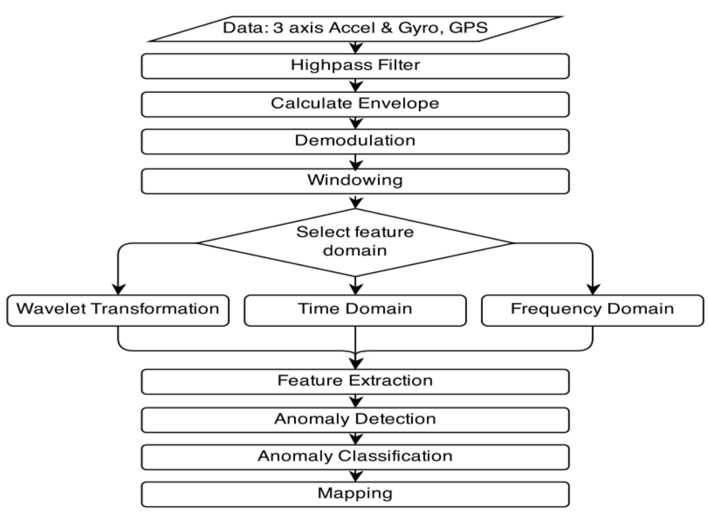


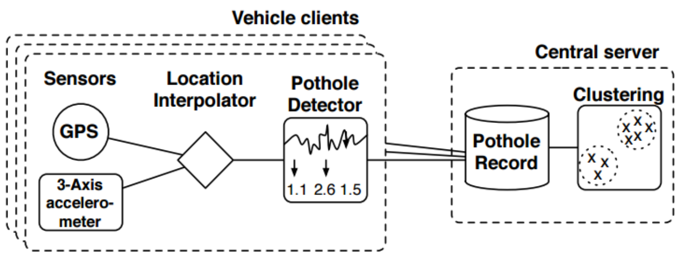
Fig. 1: RoADS flow chart

Fig 5. RoADS system flowchart [5]

The positive aspects and challenges of this system have already been discussed in section 1.1, 1.2 and to some extent in section 2.1a.

2.4b Pothole patrol

Pothole patrol is a mobile road quality sensing application developed by Eriksson et al. [6] from the MIT Computer Science and Artificial Intelligence Laboratory. A broad system overview can be found in fig 6.



Concept summary
According to Eriksson et al. [6] Pothole Patrol "uses the inherent mobility of the participating vehicles, opportunistically gathering data from vibration and GPS sensors, and processing the data to assess road surface conditions".

Fig 6. Pothole patrol system overview [6]

Positive aspects
What can be noted in fig 6 is that the Pothole Patrol system already includes an architecture in which multiple vehicle sensor clients can send information over to a central server. The server then is able to classify road anomalies based on clustering of data.

Challenges
Firstly, based on the overall system overview in fig 6 it can be noted that the system does not include a user terminal in order to view data that is put into the central server. Therefore the system is not able to visualize IRI numbers. Instead the system requires a potential user to extract the data and plot it into existing road quality data management software.

Secondly, the paper does not specify the type and performance of the central server that is used in the system. Therefore it is not known to what extent the application server is able to process road quality data in an efficient manner. This information is vital for the participatory sensing system proposed in this thesis since system efficiency has to be maintained for an increase in participating dedicated sensor systems and users requesting information through a terminal from a central server [6].

Lastly, in the paper written by Eriksson et al. no mention is made of the system being able to compute IRI data based on the acquired sensor data from vehicle clients [6].

2.5 Innovation in regard to state of the art systems

What can be noted in sections 2.1 - 2.4 is that all the systems contained in the state of the art field of road quality data acquisition and maintenance do not completely have an answer to the research questions stated in section 1.3. Therefore the introduction of the participatory sensing system for road quality data acquisition proposed in this thesis is novel, e.g. it introduces innovation to the state of the art field of road quality data acquisition and maintenance.

The innovation is reached through providing a system that is capable of obtaining sensor information from a car environment, based on this sensor information classify road anomalies and compute and IRI number, store this information inside a database and providing a GUI that can be used real-time. The system will also be able to mitigate some of the challenges reached such as limiting use of the system for unintended purposes and providing a road quality indication more frequently. The next section will go deeper into how value can be given from the participatory sensing system to potential system stakeholders.

3. Stakeholder analysis

The stakeholder analysis sheds light on existing and potential future stakeholders of the participatory sensing system and how their interest and wishes can be included in the participatory sensing system design.

3.1 Stakeholder definition

In order to be able to properly analyze possible stakeholders, first a stakeholder definition is needed. Therefore a stakeholder definition was determined. This definition is used throughout the stakeholder analysis as a basis to link the relevance of the stakeholder to the participatory sensing system.

According to an article written by Samantha Miles [42] the amount of different ‘Stakeholder’ concept definitions has been growing drastically from 38 different explanations reported in 1997 by [43] up until 435 different explanations reported in 2011 by [44]. As a result there is a lot of confusion over the ‘Stakeholder’ concept. Therefore the choice was made to use the initial ‘Stakeholder’ concept definition introduced by Freedman in 1984 in his book called ‘Strategic management: a stakeholder approach’ [45]. The definition is as follows: “*A stakeholder is any group or individual who is affected by or can affect the achievement of an organization’s objectives*”. In case of the participatory sensing system the organization’s objectives is the system’s goal.

The ‘goal achievement affectivity’ of every stakeholder can be measured through the use of the ‘power-interest’ grid introduced in a paper written by Ackermann and Eden [7]. Next to that the grid also filters stakeholders on general interest in the participatory sensing system. An example grid can be noted in fig 7.

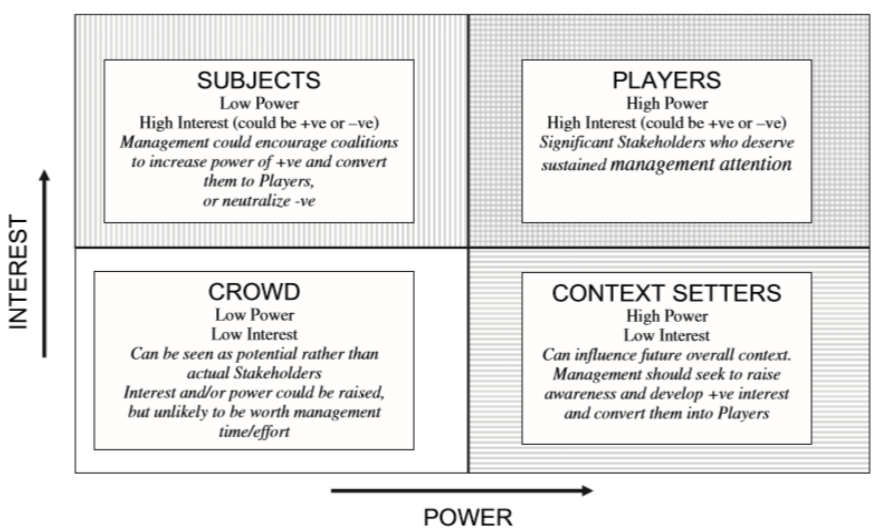


Fig 7. Outline stakeholder power-interest grid [7]

According to [7] there are 4 different stakeholder types with a certain level of interest in the participatory sensing system and a certain level of goal achievement affectivity power. The first type is called a subject. According to [7] a subject “is interested in” the participatory sensing system “but does not have the influence to alter the goal achievement”. A player “is interested in” the participatory sensing system “and has the influence to alter the goal achievement” [7]. The crowd “exhibits neither interest in or power to influence the goal achievement” of the participatory sensing system [7]. Last but not least a context setter has the power to alter the goal achievement but does not have a particular interest in the participatory sensing system [7].

3.2 Stakeholder brainstorm

In order to determine stakeholders that fit in the power-interest grid presented in section 3.1 a stakeholder brainstorm was done. The result of the stakeholder brainstorm can be found in fig 8.

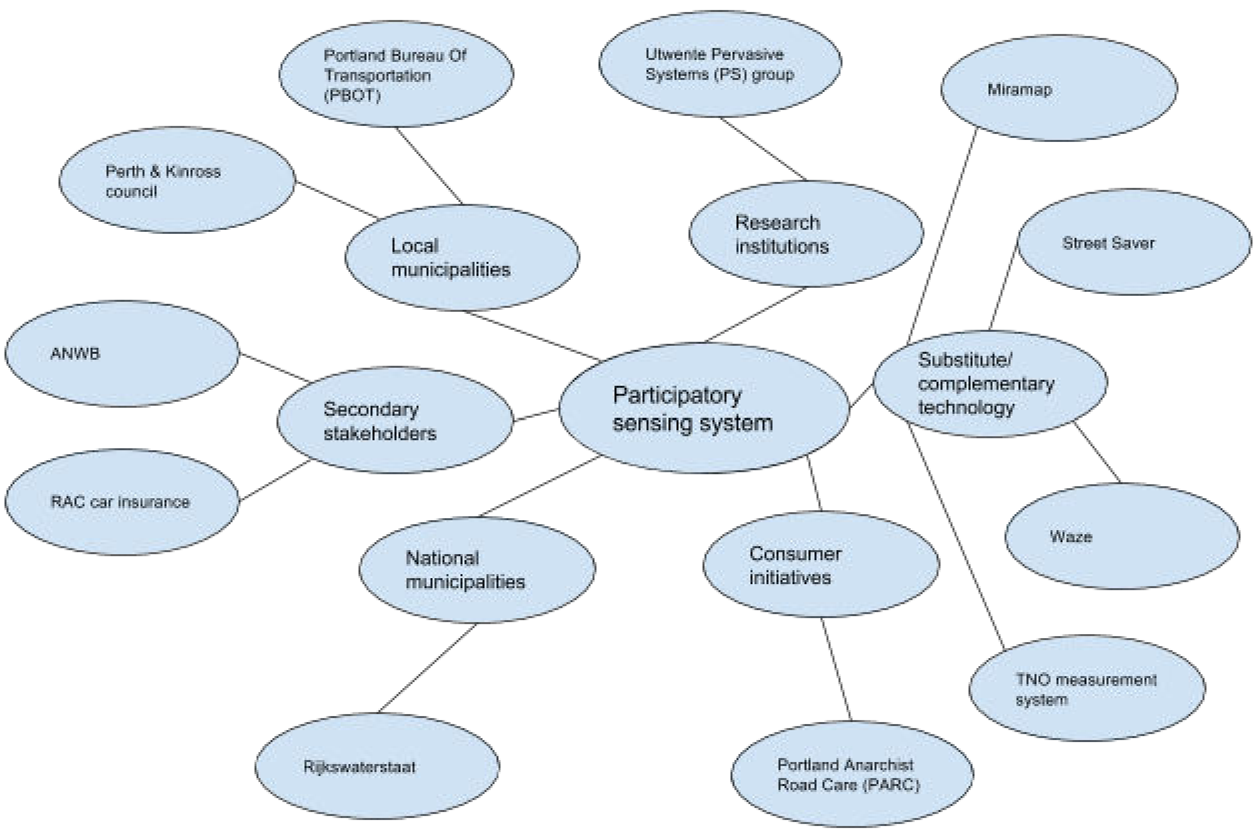


Fig 8. Stakeholder brainstorm results. Note: all abstract envisioned stakeholder groups, apart from the secondary stakeholder group, are primary stakeholder groups.

In fig 8 it can be noted that a distinction is made between primary and secondary stakeholder groups. The definitions of a 'primary' and 'secondary' stakeholder used in this report come from a community toolbox created by the University of Kansas [46]. The toolbox argues that "primary stakeholders are the people or groups that stand to be directly affected, either positively or negatively, by an effort or the actions of an agency, institution, or organization" [46] and that "secondary stakeholders are people or groups that are indirectly affected, either positively or negatively, by an effort or the actions of an agency, institution, or organization" [46].

The following definition illustrations will help with making the definitions less abstract. The Utwente Pervasive Systems (PS) group will be able to use the participatory sensing system directly for other research purposes (see appendix F). As a result the PS group is a primary stakeholder. When the participatory sensing system is implemented better and less costly maintenance schedules can be made. As a result the overall road quality will improve. Therefore less car damage as a result of bad road quality will have to be paid by a car insurance company like the RAC. Therefore the RAC is a secondary stakeholder.

3.3 Interview result highlights

As a result of the stakeholder brainstorm above a few of the envisioned stakeholders were asked to participate in a small interview regarding the participatory sensing system. The interview schemes and results that were given by the stakeholders can be found in appendix A, B, C, D, E and F.

3.3a focus group definition

A stakeholder 'power-interest' grid specifically made for the participatory sensing system can be noted in fig 9. Interview results were obtained from the Utwente Pervasive Systems (PS) group, from the Portland Anarchist Road Care (PARC) group and from the Portland Bureau of Transportation (PBOT). Therefore their position in the stakeholder matrix is based on the answers that were given during the interview and context observations. The position of the other stakeholders is envisioned through context observations.

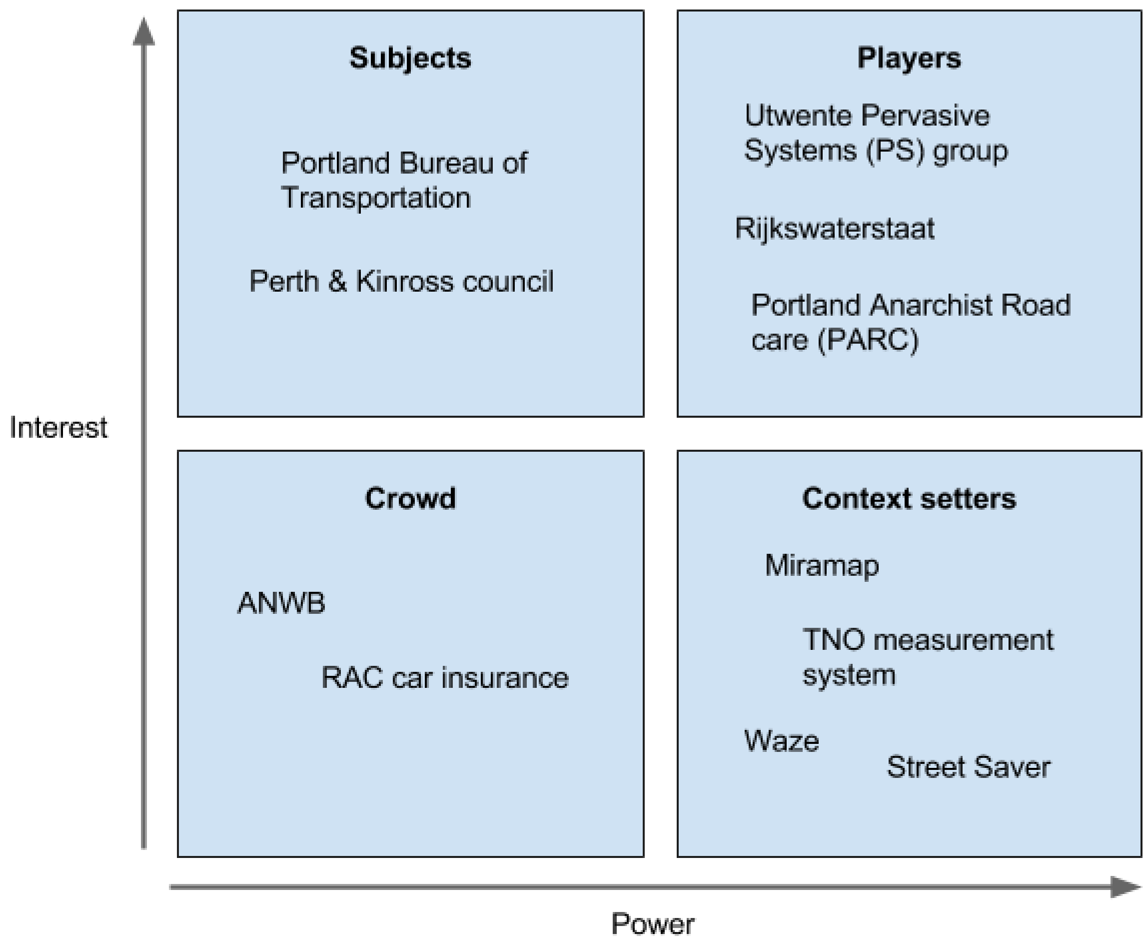


Fig 9. Stakeholder matrix [7].

Firstly, it was decided to include the PS group, PARC group and Rijkswaterstaat in the players section. The first reason why the PS and PARC groups have been put into the players section is that they have reacted to the interview that has been sent to them. As a result it can be noted that there is a direct interest in the solution from their side. The second reason is that in case of the PS group the project is issued by them. In the interview result (see appendix F) it can also be noted that not much similar research projects have been issued by other research groups. As a result they have a lot of power over it. The second reason in case of the PARC group is that they have a lot of reach (essentially people that are situated in similar road quality situations all over the world through social media). If the participatory sensing system turns out to be very useful in their case the PARC group could start using their reach for viral marketing[2] purposes on social media in order to help other people. This gives them a lot of power over the success rate of the participatory sensing system.

[2] Viral marketing (a.k.a word-of-mouth marketing): marketing through people telling other people about things that interest them [47].

The first reason why rijkswaterstaat has been put into the players section is that rijkswaterstaat, compared to local municipalities such as the Portland Bureau of Transportation and the Perth & Kinross council, does have a large reach (Large part of the dutch population using highways (A-roads[3]) and some main roads (large N-roads[4])). The second reason is that the participatory sensing system, compared to the road quality indication systems used by rijkswaterstaat today, is going to be cheaper to use. As a result, when the system is implemented, rijkswaterstaat is able to use a lot of free budget as a result of the participatory sensing system for other important infrastructure matters.

Secondly, it was decided to include the Portland Bureau of Transportation and the Perth & Kinross council in the subjects section. The first reason that the Portland Bureau of Transportation and the Perth & Kinross council have been placed in the subjects section is that their reach is significantly less (these local municipalities only have a reach to people inside their own districts). As a result their power over the success rate of the participatory sensing system is significantly less. The second reason is that both local municipalities are positively affected by better maintenance planning. As a result they have less costs in terms of road maintenance and will gradually experience better road quality results. As a result they will show interest in the participatory sensing system. Additionally the Portland Bureau of Transportation also gave a reaction on the interview that was sent to them.

Thirdly, it was decided to put Miramap, TNO measurement system, Waze and Street Saver in the context setters section. The first reason why Miramap, TNO measurement system, Waze and Street Saver have been placed in the context setters section is that they are substitute systems[5] compared to the participatory sensing system. Therefore they have the power to influence participatory sensing system use since people might already be using the Miramap, TNO measurement systems, Waze or Street Saver or might be more interested in using those systems. The second reason is that in terms of interest these stakeholders are probably more interested in the development of their own system rather than the participatory sensing system.

Fourthly, RAC car insurance and the ANWB were placed in the crowd section. The reason that the RAC car insurance and the ANWB have been put in the crowd section of the stakeholder matrix is due to the fact that they are secondary stakeholders. As a result they will not show a particular interest or have a lot of power over the success rate of the participatory sensing system since they are influenced indirectly.

---

[3] A-roads: Freeways (in the Netherlands highways do not exist due to the fact that we do not have toll booths anymore) [48], [49].
[4] N-roads: Frontage roads [50], [51].
[5] Substitute systems: systems that can be used instead of the participatory sensing system without posing a lot of switching costs [52]

Last but not least it was decided to put a persona and use case focus on the stakeholders that have reacted through email to the interview requests and are located in the ‘Players’ section of the grid in fig 9. These stakeholders are located in the players section, because they have interest in the participatory sensing system and power over the success rate of the participatory sensing system. Additionally these stakeholders supplied good support material for personas and use cases.

3.3b Personas

Based on the interview results (see appendix D, E and F) and the focus group defined in section 3.3a personas have been constructed. These personas can be found in the two sections below. The personas are based on a framework presented in appendix G. The reason stakeholder personas have been determined is to give an idea about the type of users that are likely going to use the system. This idea can later be used for user interface design purposes. All images used in this section originate from [53].

3.3b i Zoey Hamich

Demographic information
- 31 years old
- Finished M.Sc Political science
- Works as a local news reporter focusing on political themed subjects
- Possesses the ability to activate crowds to help solve communal problems (high interpersonal intelligence[6])
- Learns through experimenting and analyzing
- Speaks english and chinese
- Does not possess any physical or mental limitations
- Lives in a housing block where roads possess a lot of big potholes that have not been patched yet by the local municipality.
- Actively supports communal initiatives such as PARC in order to fix big pothole problems

Technical information
- Owns a PC and Smartphone and knows how to operate general applications on a PC and Smartphone
- Has a positive attitude towards technology with a low level of privacy risks involved
- Is both a direct and indirect user (e.g. road quality monitoring works automatically when Zoey drives. Pothole fixing and status updating requires providing information actively)
- Will use the participatory sensor system daily in terms of quality monitoring and incidentally to actively fix potholes during mass pothole action days organized by PARC
- Will use the system monitoring aspect in a car setting and pothole view and status access in an on-road setting
- Possesses limited access to tools and resources (tools and resources come in the form of seed capital[7])

[6] Interpersonal intelligence: having good communicating skills, having an analytical eye for mood changing and being good at taking multiple perspectives on subjects [54].
[7] Seed capital: capital (resources) being raised by family, friends or by oneself [47].

Participatory sensing system domain information
- Has little to no knowledge about the participatory sensing application domain
- Currently uses a participatory smartphone application in which people can report a pothole and update the status of every pothole manually
- Thinks the participatory sensing system is a powerful piece of technology that can report road anomalies more efficiently than the current piece of technology she uses

3.3b ii Andres Perrio

Demographic information
- 25 years old
- Doing Ph.D in infrastructure quality data correlation
- Possesses the ability to test hypotheses and quantify data (high logical mathematical intelligence[8])
- Learns through reasoning, analysis and interpreting data
- Speaks english and spanish
- Is colorblind
- Lives in a university accommodation

Technical information
- Owns a PC and Smartphone and knows how to operate applications with any level of complexity on a PC and Smartphone.
- Has a positive attitude towards technology in general
- Is a direct user (extracts data manually from the participatory sensing system that can be used for research purposes)
- Will use the participatory sensing system incidentally when data has to be acquired in order to test a hypothesis based on the data present at the participatory sensing system
- Will use the complete user interface aspect of the system in an office setting both for general data inspection and data extraction
- Possesses good access to tools and resources (tools and resources come from a university)

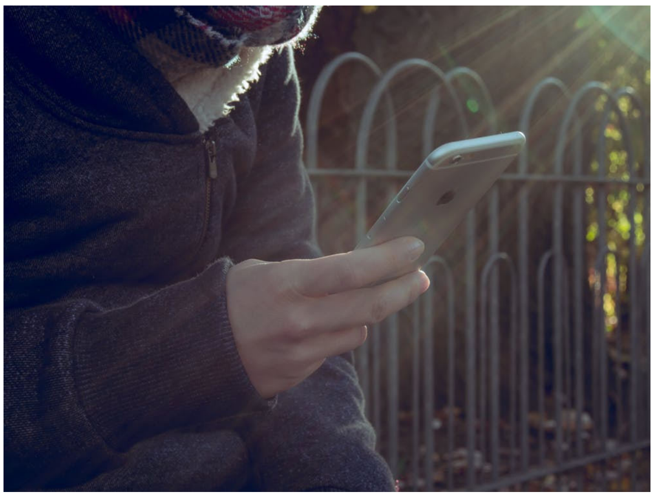
Participatory sensing system domain information
- Has a lot of knowledge about several aspects of the participatory sensing system that share a theoretical basis in data science. Is able to access necessary information if needed for all the other aspects
- Currently uses data gathered from smartphone sensor applications
- Thinks the participatory sensing system is suitable for testing data heterogeneity (e.g. how can data from the participatory sensing system be correlated with data from other systems?)

[8] Logical mathematical intelligence: being good at considering hypotheses, carrying out calculations and quantifying data [54].

3.3c Use cases: scenarios

Based on the interview results (see appendix D, E and F) and the envisioned environments in the persona section (section 3.3b) a few scenarios have been constructed. These scenarios can be found in the sections below. The reason use case scenarios have been determined is to give an idea about the type of use case scenarios that might occur if a stakeholder is interacting with the system. This idea can later be used for user interface design purposes. All images used in this section originate from [55].

3.3c i Accessing interface in an on-road situation



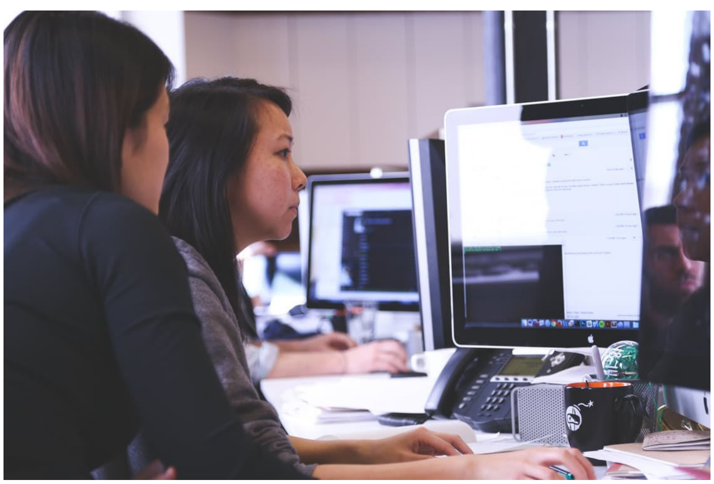
Standard activities

1. Access IRI data for specific area
2. Access road anomaly data for specific area
3. Update road anomaly status for specific road anomaly

Extra situations that need to be accounted for

1. User interface must be accessible in a correct way on multiple screen sizes (not every smartphone and PC have the same screen size).

3.3c ii Inspecting/extracting information from user interface in an office setting



Extra situations that need to be accounted for

1. An extraction interface has to be available in which the user can:

a. Choose a file type.
b. Choose a data structure type.
c. Filter data to be extracted on data type, timestamp range and location range.

2. A color legend has to be added to every user interface map in order to make sure that potential colorblind persons are able to distinguish the colors and their meaning on the screen even if the colors portrayed to the user are not correct.

3.3c iii setting up dedicated sensor system in a car

Standard activities

1. Inserting dedicated sensor system in a car cigarette charger
2. Pairing dedicated sensor system with mobile application through Bluetooth low energy

Extra situations that need to be accounted for

1. A small tutorial should be added in order to explain the setup process of the dedicated sensor system in combination with the mobile application

3.3d instrumental value impact

During the interviews stakeholders were asked to envision instrumental value impact among themselves and other stakeholders when participating in the participatory sensing system. The results have been divided into four value sections: economical, scientific, social and political value. All of the paragraphs in section 3.3d are backed up by the interview results in appendix D, E and F.

3.3d i economical value

The participatory sensing system can make an economical impact for local and national governments through improving maintenance planning. Next to that the system might be able to decrease the likelihood of people and/or insurance companies having to spend money on car damage caused by bad road quality.

3.3d ii scientific value

Firstly, the participatory sensing system could be used in research about governmental issues such as road safety and road quality monitoring. Secondly, the participatory sensing system could be used in research about obtaining a better insight about infrastructure quality other than just road quality (for example determining defect train track segments based on a quality indication). Thirdly, the participatory sensing system could be used in big data (data science) research about heterogeneity of data (to what extent can data from this system be correlated with data from other systems in order to obtain new insights?).

3.3d iii social value

The participatory sensing system can help with improving the overall safety and security of roads.

3.3d iiii political value

The primary focus of one of the stakeholders is to mobilize people to reclaim public spaces for the community. Part of this means, according to this stakeholder, taking responsibility for the conditions of the roads instead of tolerating the deteriorating road conditions until weather improves and letting bad performing municipalities take care. The value the system can deliver here is better mobilization of people due to the fact that potholes and road quality information is reported automatically (faster than the implemented app for crowdsourcing information about potholes that is shared between everyone with the app). People do not have to participate actively in pothole reporting.

3.3e Requirements analysis

During the interview stakeholders were also asked to think about possible requirements they would have when a participatory sensing system was available for use. Based on the interview results (see appendix D, E and F) and the scenarios presented in section 3.3c requirements are defined in this section.

The requirements are presented in table 1. These requirements are prioritized according to the MoSCoW model first presented by Dai Clegg in his book called 'Fast-Track: A Rad Approach (Case Method)' [28]. The MoSCoW method prescribes requirement prioritization according to 4 categories.

The first category is called 'Must' requirements. These requirements are vital for the successful realization of the dedicated sensor system in the given development time frame and failing to meet them will result in project failure [28].

The second category is called 'Should' requirements. These requirements are as vital as the 'Must' requirements but do not necessarily have to be met in the given development timeframe due to the fact that failing to meet the requirement does not lead to project failure. The reason that project failure does not occur is that the problem imposed by not meeting the requirement resulting in project failure can be mitigated [28].

The third category is called the 'could' requirements. These requirements can add extra functionality, impact etc. to the development of the dedicated sensor system but are not a necessity [28].

Last but not least there is a category called the 'won't' category. These requirements are deliberately not met due to for example certain design decisions [28].

Requirement	*Must*	*Should*	*Could*	*Won't*
1. Prevent storage of identification data	*x*			
2. Introduce road anomaly status functionality	*x*			
3. User motivation			*x*	
4. Power consumption efficiency		*x*		
5. Seamless subsystem integration		*x*		
6. User interface accessible on multiple screen sizes		*x*		
7. Inclusion of sensing system setup tutorial			*x*	
8. Inclusion of color legend for user interface maps			*x*	

Table 1. Stakeholder requirements prioritized with the MoSCoW model [28]

Now that it is known how value can be delivered to potential participatory sensing system users and what the participatory sensing system requirements are from the user perspective it is important to obtain theoretical knowledge about how a participatory sensing system prototype can be designed. The reason for this is that the knowledge in turn can be used to construct a participatory sensing prototype. This theoretical knowledge is obtained in the next section.

4. Technical system analysis

The system analysis provides a thorough understanding of the different technical aspects contained within the participatory sensing system and plays a vital role in the design decisions that will be taken in the design phase presented in chapter 5. First of all a high level system overview is given in fig 10. Based on this overview different aspects of the system are analyzed in several sub paragraphs.

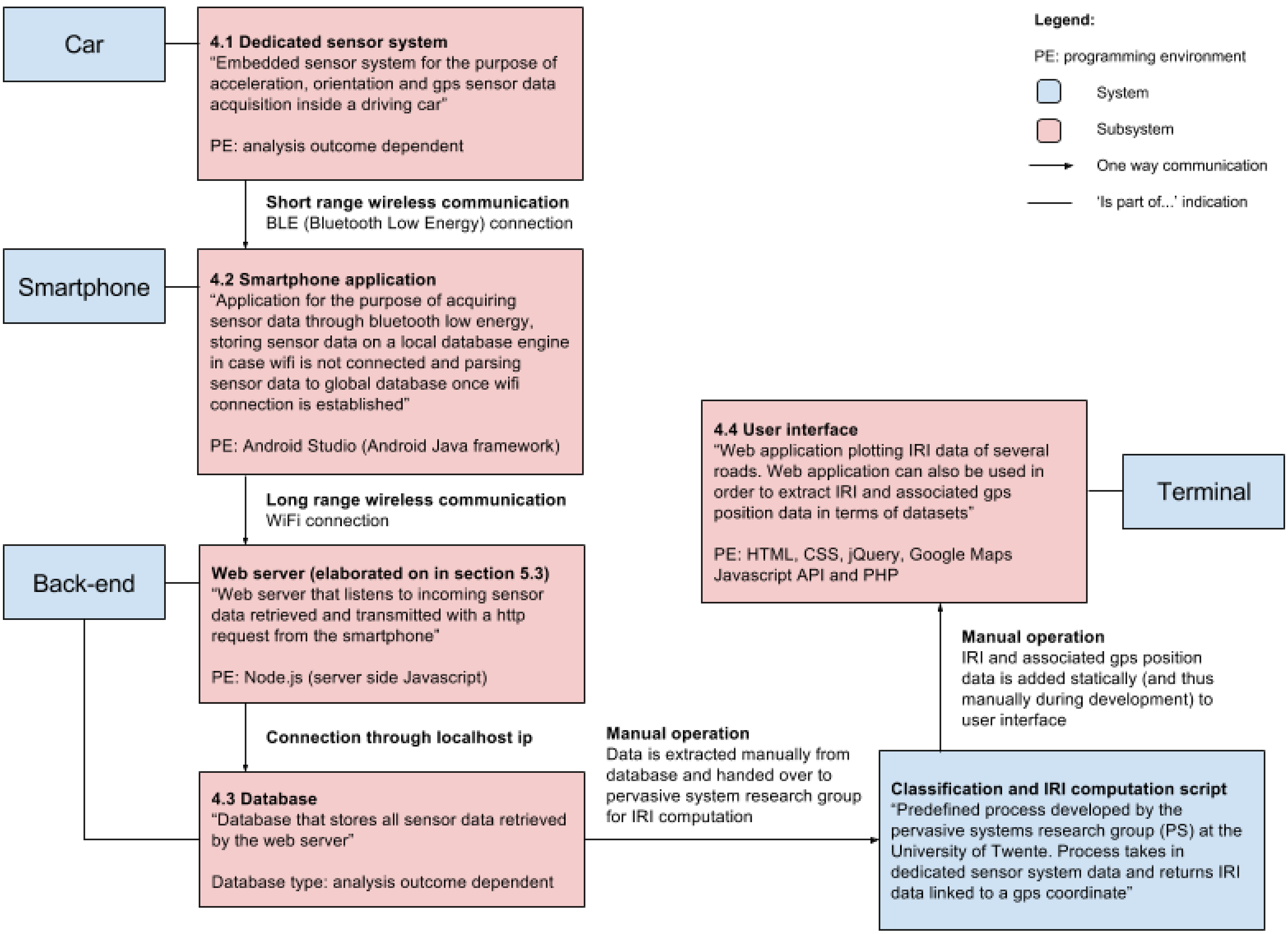


Fig 10. Technical high-level overview of the participatory sensing system

4.1 Dedicated sensor system

4.1a Sensor system requirements

The requirements involving the dedicated sensor system are presented and prioritized according to the MoSCoW model [28] (for an in depth explanation on the MoSCoW model go back to section 3.3e). The dedicated sensor system requirements are presented in table 2.

Requirement	*Must*	*Should*	*Could*	*Won't*
1. Maintain a fixed position inside a car	*x*			
2. Powered by car cigarette socket		*x*		
3. Sense 3-axis acceleration	*x*			
4. Sense 3-axis orientation	*x*			
5. Able to sense immediately after power is connected	*x*			
6. Determine GPS position coordinates from GPS NMEA data	*x*			
7. Determine speed from GPS NMEA data	*x*			
8. Determine if car is driving or not	*x*			
9. Track time	*x*			
10 Synchronize time from GPS NMEA data and processor clock time	*x*			
11. Use short range communication over BLE	*x*			
12. Provide reference work on development of system firmware			*x*	
13. Provide hardware schematics for purpose of determining hardware alterations to fulfill requirements that are initially not met			*x*	
14. Provide user's guide	*x*			

Table 2. Sensor system requirements prioritized using the MoSCoW model [28]

4.1b State of the art wireless sensor-driven hardware platforms

Based on the requirements stated in section 4.1a one of the state of the art sensor-driven hardware platforms presented in section 4.1b will be picked and used in the design phase to construct a sensor-driven hardware platform with all the necessary requirements included that were defined in section 4.1a.

The 3 systems with the greatest amount of requirements met by the initial system that were found during an investigation of the sensor-driven iot hardware kit market were picked. The reason for this is that the more requirements met by the initial system, the more development time can be saved later on in the design process.

The requirement evaluations presented in tables 3, 5 and 7 have been acquired based on subjective analysis by inspecting datasheets and hardware schematics, looking at the volume of development support at forums across the internet and thinking about possible alterations in order to meet requirements not met by the initial system.

4.1b i Silicon Labs Thunderboard React

Silicon Labs defines the thunderboard react kit as “a cloud-connected, Bluetooth® Smart-enabled, sensor-driven platform that enables customers to demo, evaluate, and develop their own unique applications” [56]

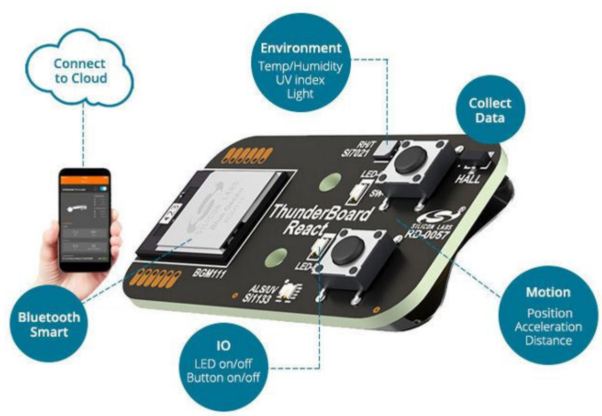


Fig 11. Thunderboard react product image [8]

How does the thunderboard react kit compare to the dedicated sensor system requirements? The results can be found in table 3.

1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	11.	12.	13.	14.

Table 3. Thunderboard react requirement comparison (for the actual requirements see table 2). Note: Red means does not meet the requirement and Green means does meet the requirement

Even though a lot of requirements were not met by the initial system it is very easy to alter the system in order to meet all of the other requirements as well. Requirement 1 can be met by constructing a case that can be mounted inside a car. When taking a look at the thunderboard react schematics (see appendix H) it can be noted that the thunderboard react includes user accessible pads and that the lithium battery acts as an independent voltage source. Therefore requirement 2 can be met by connecting a car to usb power socket in combination with a usb cable with exposed voltage and ground channels and either a voltage regulator or buck converter that will supply a constant 3V output to the exposed terminal pads for the coin-cell battery holder. Next to that requirement 6 and 7 can be fulfilled by interfacing a gps sensor to two free UART ports of the processor. Requirements 5, 8 and 9 can be fulfilled due to the fact that requirement 12 is met and the firmware of the board can be programmed in order to fulfill requirements 5, 8 and 9 digitally.

The price to implement a dedicated sensor system based on this system can be found in table 4.

Component name	**Price (EUR)**
Thunderboard react board	29,93
Ulink2 arm emulator (debug adapter)	12,13
Ublox GY-NEO6MV2 GPS module	9,49
Lm317t voltage regulator	0,19
To220 heatsink	0,30
Car to usb power adapter	0,47
Usb cable	0,37
Casing	2
Total	54,88

Table 4. Dedicated sensor system realization costs. Note: these prices are just an indication and may differ across vendors due to profit margins, shipping costs etc.

4.1b ii Nordic nRF52 BLE development kit

Nordic semiconductors defines the nRF52 BLE development kit as "a versatile single board development kit for *Bluetooth*® low energy, ANT and 2.4GHz proprietary applications using the nRF52832 SoC" [57].

Fig 12. Nordic nRF52 BLE development kit product image [9]

The requirement evaluation for the nordic nRF52 BLE development kit can be found in table 5.

1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	11.	12.	13.	14.

Table 5. Nordic nRF52 BLE development kit requirement comparison (for the actual requirements see table 2). Note: Red means does not meet the requirement and Green means does meet the requirement

Interesting to note is that compared to the thunderboard react the nordic nRF52 development kit does not meet as much requirements as the thunderboard react does. However, the nordic nRF52 development kit does have one very big advantage over the thunderboard react. The advantage is that, in contrast to the thunderboard react, BLE pairing is done through NFC technology. As a result the pairing process is much faster compared to the thunderboard react, essentially making the nordic nRF52 development kit more suitable for requirement 5 when this requirement is incorporated in the firmware.

Requirement 1 can be met by constructing a case that can be mounted inside a car. Requirement 2 can also be met by by connecting a car to usb power socket in combination with either a voltage regulator or buck converter that will supply a constant 3V output to the exposed terminal pads for the coin-cell battery holder (see appendix B). Requirement 3 and 4 can be met by interfacing either a 3 axis accelerometer and gyroscope or an IMU through UART pins to the Nordic nRF52 development kit. Requirements 5, 8 and 9 can be fulfilled due to the fact that requirement 12 is met and the firmware of the board can be programmed in order to fulfill requirements 5, 8 and 9 digitally. Requirement 6 and 7 can be met by interfacing a GPS module to the UART pins of the Nordic nRF52 development kit.

The price to implement a dedicated sensor system based on this system can be found in table 6.

Component name	**Price (EUR)**
Nordic nRF52 BLE development kit	39,23
Gyroscope and Accelerometer IMU	6,25
Ublox GY-NEO6MV2 GPS module	9,49
Lm317t voltage regulator	0,19
To220 heatsink	0,30
Car to usb power adapter	0,47
Usb cable	0,37
Casing	2
Total	58,30

Table 6. Dedicated sensor system realization costs. Note: these prices are just an indication and may differ across vendors due to profit margins, shipping costs etc.

4.1b iii Femtoduino IMUduino

Femtoduino defines the IMUduino as “the world's smallest Arduino Leonardo compatible board with integrated wireless motion and orientation sensing technology … using Bluetooth Low Energy” [10]

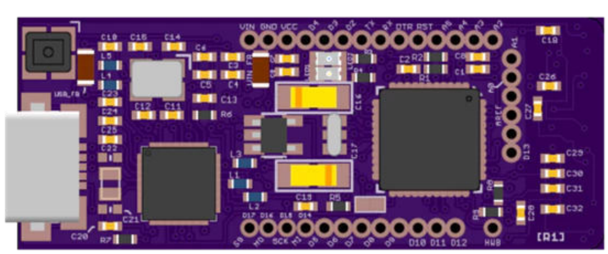


Fig 13. Femtoduino IMUduino product image [10]

The requirement evaluation for the Femtoduino IMUduino can be found in table 7.

1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	11.	12.	13.	14.

Table 7. Femtoduino IMUduino requirement comparison (for the actual requirements see table 2). Note: Red means does not meet the requirement and Green means does meet the requirement

A big advantage the imuduino has over the other two hardware platforms is that it can be programmed using the arduino IDE. As a result firmware development for this board can be done much faster since the firmware is based on a framework and thus less complex than the development of firmware for the other boards.

However, there are 2 reasons why the IMUduino does not meet as much requirements as the other initial systems do. The first reason is that a lot of the hardware documentation that was supposed to be situated on the internet for some reason has disappeared. The second reason is that the IMUduino is only available in packs of 10 while the participatory sensor prototype only requires one. Next to that the IMUduino board is also rather pricey. Even if it would be possible to discuss a deal with Femtoduino to buy just one IMUduino board the price would result in something between 150-200 USD (150 USD is the bottom line since that is the price for one board in packs of 10, therefore the price for one board will go up when only one board is requested since no quantity discount will be possible).

Also in the case of the IMUduino a lot of the requirements can be met. Requirement 1 can be met by constructing a case that can be mounted inside a car. Requirement 2 might be somewhat hard to implement since the design does not already support a second power source next to the micro port. Requirements 5, 8 and 9 can be fulfilled due to the fact that requirement 12 is met and the firmware of the board can be programmed in order to fulfill requirements 5, 8 and 9 digitally. Requirement 6 and 7 can be met by interfacing a GPS module to the UART pins of the Femtoduino IMUduino.

The price to implement a dedicated sensor system based on this system can be found in table 8.

Component name	Price (EUR)
Femtoduino IMUduino	141,43 - 188,57
Ublox GY-NEO6MV2 GPS module	9,49
Lm317t voltage regulator	0,19
To220 heatsink	0,30
Car to usb power adapter	0,47
Usb cable	0,37
Casing	2
Total	154,25 - 201,39

Table 8. Dedicated sensor system realization costs. Note: these prices are just an indication and may differ across vendors due to profit margins, shipping costs etc.

4.1c initial sensor-driven hardware platform board decision

Based on all the data presented in sections 4.1b i - iii a weighted decision matrix can be made. The weighted decision matrix is a method of making decisions based on presenting analytical data ordered in relevant decision aspects and assigning a value based on that data to several options which can be chosen. In the end all the values linked to the decision aspects are added together and the option with the biggest amount of value is chosen. The weighted decision matrix, also known as the Pugh matrix, was invented by Stuart Pugh and first presented in the conference proceedings paper called 'Concept selection: a method that works' [29]. The weighted decision matrix for the sensor-driven hardware platforms can be found in table 9.

items/categories	1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	11.	12.	13.	14.	Price	Total
Thunderb. react	0	0	3	3	0	0	0	0	0	0	3	1	1	3	3	17
nRF52 DK	0	0	0	0	0	0	0	0	0	0	3	1	1	3	2	10
IMUduino	0	0	3	3	0	0	0	0	0	0	3	1	0	0	1	11

Table 9. Sensor-driven hardware platform weighted decision matrix. Note 1: weights are linked to the MoSCoW model in the following way: requirements not met receive 0 points, could requirements 1 point, should requirements 2 points, must requirements 3 points and won't requirements -1 point. Note 2: the price requirement weights are linked to the price which has to

be paid in order to realize the system. From lowest to largest price: 3-1 points. Note 3: the light green color marks the board which is going to be used in the design phase.

4.2 Smartphone application

Firstly this section gives an explanation about the way a BLE connection between the dedicated sensor system and the smartphone application will be established. Secondly this section will give an explanation about how data can be stored locally on a smartphone through the smartphone application when no Wi-Fi connection can be established on a smartphone. Lastly this section will shed a light on the type of connection between the smartphone and main database system and how data will be transmitted once a Wi-Fi connection is established.

4.2a Dedicated sensor system to smartphone

In order to establish a bluetooth low energy connection on the application layer level between the dedicated sensor system and smartphone application that is ready to exchange data between these two devices bluetooth has specified two technological building blocks called profiles. The first profile is called the Generic Access Profile (GAP) and the second profile is called the Generic Attribute profile (GATT). These two profiles are complementary building blocks to each other. This means that in order for a connection to be established and ready to exchange data both profiles have to be used.

4.2a i BLE GAP

According to the bluetooth core specification the GAP profile “defines the procedures and roles related to the discovery of Bluetooth devices and sharing information, and link management aspects of connecting to Bluetooth devices.” [58]. The bluetooth core specification description for GAP profiles raises the following question: how does it work and what roles are involved?

This question is best answered by a youtube video on bluetooth low energy basics created by Kelvin Aviles [59]. In this video Aviles explains that there are two roles defined in the GAP profile: Centrals and Peripherals. In case of the participatory sensing system the dedicated sensor system acts as a peripheral while the smartphone application acts as a central. The peripheral will transmit advertisement data that can be picked up by bluetooth low energy scanners at the central side. When the central requests a bluetooth low energy connection the GAP profile at the dedicated sensor system side will handle this request [59].

4.2a ii BLE GATT

Once a BLE connection has been established the GATT profile comes into play. The GATT profile is defined by the bluetooth core specification as “a hierarchical data structure that is exposed to connected Bluetooth LE devices.” [11]. An example of this data structure can be found in fig 14.

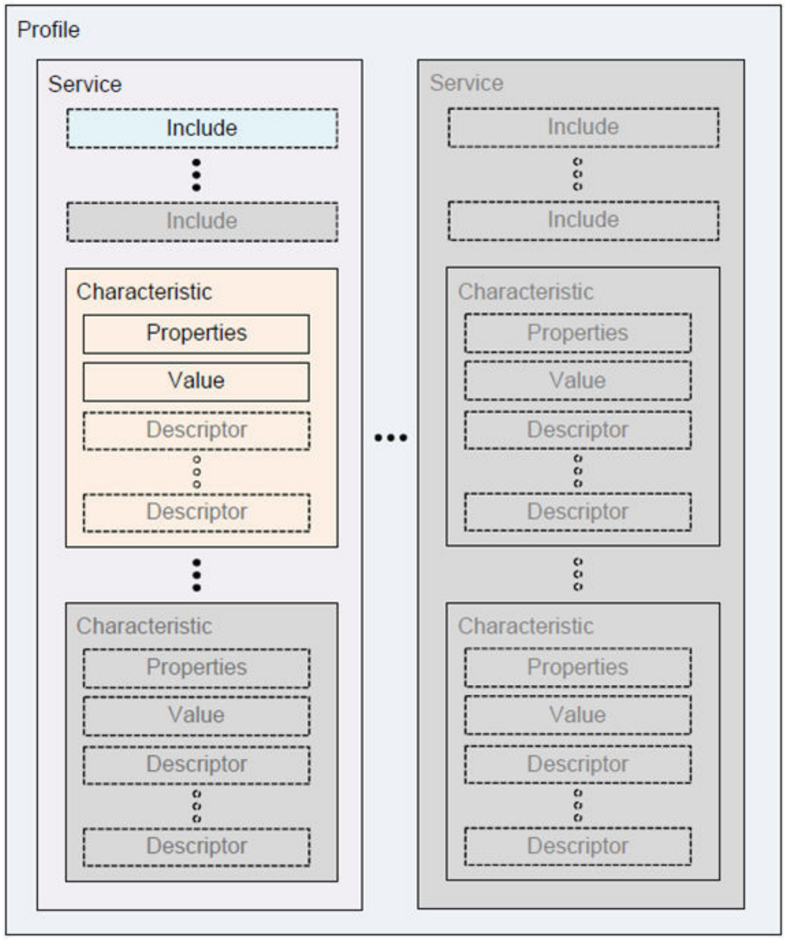


Fig 14. Bluetooth GATT profile data structure [11]

According to the bluetooth core specification “The top level of the GATT profile data structure is a profile, which is composed of one or more services necessary to fulfill a use case. A service is composed of characteristics or references to other services. A characteristic consists of a type (represented by a UUID), a value, a set of properties indicating the operations the characteristic supports and a set of permissions relating to security. It may also include one or more descriptors—metadata or configuration flags relating to the owning characteristic.” [11].

After reading the GATT profile data structure the following question was asked: How does the ble gatt data structure tie into an automated data exchange between the dedicated sensor system and android smartphone application? The reason to ask this question has to do with the fact that one of the system requirements determined in section 3.3e is to make the participatory sensing system ‘seamless’.

In order to answer this question first the different GATT profile roles and ways these roles can interact with each other have to be defined. According to the bluetooth core specification GATT defines two roles: the client and the server [11]. In case of the participatory sensing system the dedicated sensor system is the server while the android smartphone application is the client. The server is the place where the sensor data is generated and afterwards send to the BLE chip that in turn updates the service and characteristic value that specify the sensor data. The client can subscribe to a service and determine which procedure it wants to use in order to receive updated characteristic values associated to that service. Afterwards the server sends asynchronous notifications to the client as soon as characteristic values are updated at the server side [11].

The different procedures the client can request are read, write and notify [58]. It was determined that in order to automate data exchange between the dedicated sensor system and android smartphone application the best procedure is the notify procedure. The reason for this can be explained through the nature of the notifying procedure. The notify procedure will establish an event-driven asynchronous subscription to a service and characteristic connection from the client to the server. Every time the characteristic value associated to the sensor is updated a notification is sent to the client from the server. This notification on the client side will trigger a callback containing the updated characteristic value that can then in turn be used at the client side for whatever purpose suits the application (e.g. save the data to a local or main server).

4.2b Smartphone to server

4.2b i Data transmission is not possible: local data storage

Data transmission is not going to be possible if there is no active Wifi connection established at the smartphone side. The first reason why this decision has been made is that if the smartphone application would be allowed to send data over a cellular network people could get into trouble due to high roaming costs. Secondly not every telephone has access to a cellular network.

Therefore the choice has been made to include a local data storage feature into the smartphone application. A local data storage (e.g. an embedded mobile database), according to Katerina Roukounaki, is “a lightweight, self-contained library with no server component, no need for administration, a small code footprint, and limited resource requirements. Mobile applications can be (statically or dynamically) linked to one, and then use one in order to create and manage their own – private or shared – databases locally on the device.” [60].

The type of local data storage that will be used in the participatory sensing system will depend on the type of main database that is going to be picked. The reason for this is to make the data transfer between the local data storage and main database as easy as possible without having to change the data structure the data is stored in. The local database storage decision can be found in section 4.3d.

4.2b ii Data transmission is possible

Data transmission is possible once a Wifi connection has been established. In order to transfer all the data that has been stored into the local data storage to the main database over the internet once data transmission is possible two eligible communication protocols were found. These protocols are called the HTTP protocol and WebSocket protocol.

In order to determine which protocol is best suited for the communication between the smartphone and the main database first some requirements have to be defined. Firstly when looking at the main database operating context in section 4.3a it can be noted that large amounts of data have to be sent. Secondly, since data transmission is not always possible, the protocol should not be trying to maintain a live connection between the smartphone application and main database. Instead data transmission should happen incidentally when there is a Wifi connection on the smartphone. Another reason for this requirement can be found in section 4.3c i. The reason has to do with the fact that more active connections parallel to each other to the main database cause latency in a single connection to the main database. Lastly sensor data is only communicated from the smartphone to the main database. Therefore bi-directional communication is not necessary.

A blog post created by the Windows Apps team [61] gives a good understanding on how both the HTTP and WebSocket protocol work best in certain situations. HTTP works well in situations where considerable amounts of saved data need to be send and/or retrieved incidentally while WebSocket works best in situations where small data packets from an ongoing data stream need to be saved and/or retrieved continuously [61]. Next to that HTTP works according to a request-response mechanism that does not establish a connection between the transmitting and receiving end. A WebSocket is bidirectional and will first establish a connection between transmitting and receiving end before data is exchanged [61]. Based on how both protocols work in relation to the requirements that were sketched in the paragraph above the decision was made to include the HTTP protocol in order to facilitate the data transmission between the smartphone and the main database.

4.3 Database

In order for the participatory sensing system to be build a suitable database structure is required that is able to handle massive amounts of raw sensor data (the way this raw sensor data is handled by the participatory sensing system is explained in the database operating context section). Therefore the main objective of the database literature research is to determine the most effective database structure in terms of data handling according to a few database examples that use this type of structure. Based on the result of this objective an example database is chosen that will be used in the participatory sensing system.

The literature review has been divided in three sections. The first section of this literature review provides the operating context of the participatory sensing system in which the database system has to work. The second section of this literature review presents an overview of different database structures and associated database examples used in big data applications. The third section of this literature review will present a means for reaching the objective by presenting a data handling effectiveness analysis of the database structures presented in section two in relation to the database operating context of the participatory sensing system presented in section one. Lastly a conclusion based on the literature review in order to pinpoint the most effective database structure and associated database example will be presented.

4.3a Database operating context in the participatory sensing system

In this section the technical participatory sensing system database context is sketched. This context determines requirements that a database structure and associated database examples have to meet in order to be suitable for use in the participatory sensing system.

A high level overview of the participatory sensing system database operating context can be found in fig 15. The diagram has been inspired by a dataflow diagram for a sensor network presented by van der Veen et al. [24].

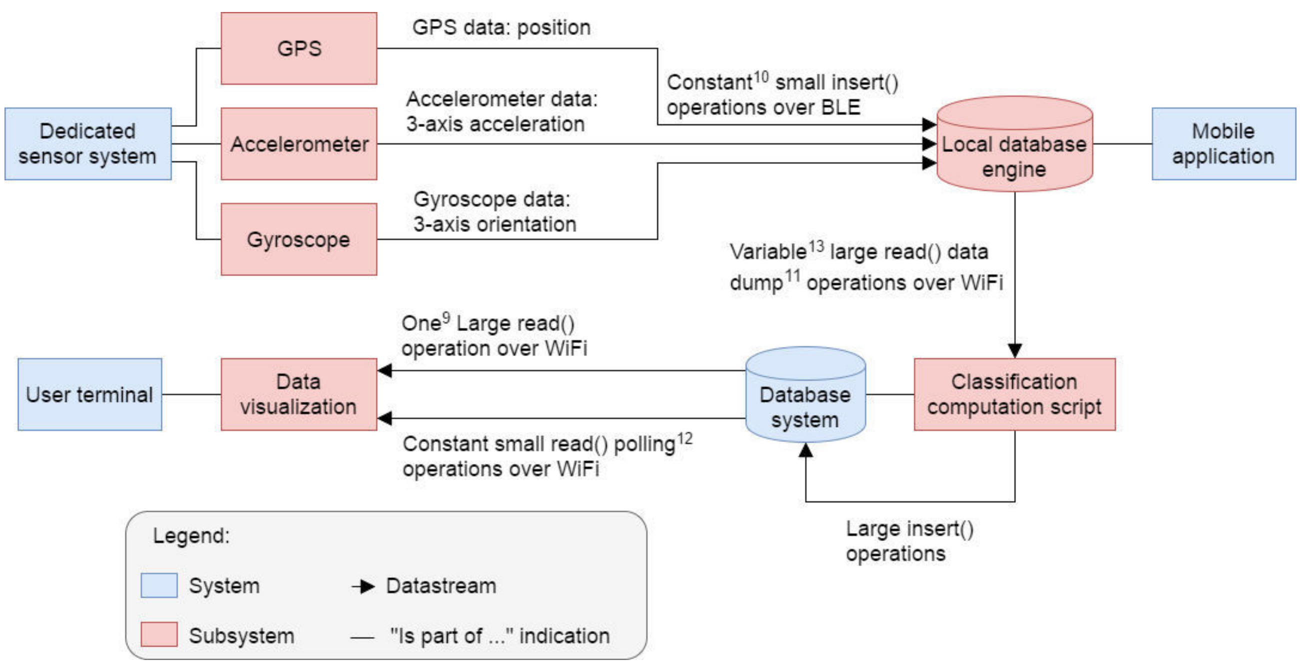


Fig 15. Participatory sensing system database operating context [12]

[9] One: one operation is executed.
[10] Constant: operations are constantly being carried out
[11] Data dump: reading all data from one system, inserting it into another system and afterwards dropping all data in the initial system (involving an extra delete() operation)

What can be noted in fig 15 is that the participatory sensing system is a distributed network of a local database engine connected to one central database system. In most cars today no stable Wifi connection can be made when the car is driving. Therefore the participatory sensing system requires a distributed database network that is able to carry out variable[13] style CRUD[14] operations.

Next to that both the database system and local database engine should be able to show reasonable CRUD operation performance in three ways. The first way is through many operations per second requiring a low amount of records in case of a single thread[15] environment (in case of the dedicated sensor system saving data in the local database engine). The second way is through many operations per second requiring a low amount of records for multi-thread connections (in case of the polling procedure from the terminal to the database back and forth). The third way is through a small amount of operations per second requiring a very large amount of records in a multithreaded environment (in case of the communication between the local database engine and the global database and the initial data retrieval between the terminal and the global database).

Additionally the system also requires the distributed database network to be horizontally scalable. The reason for this is that when more users start to participate in the system context multiple local database engines will be connected to the database system. As a result, in order to keep the user experience for every user as smooth as possible, more and more simultaneous thread connections will be necessary to handle all the database requests as soon as possible.

The GPS, Accelerometer and Gyroscopic data elements, acquired in case a certain threshold speed has been noted and no drop below the threshold speed has been noted, first are going to be saved inside the local database engine and afterwards transferred to the database system can be noted in fig 16. These data elements will be used later on in the database structure overview in order to provide, additionally to the textual explanations, a graphical explanation of every database structure.

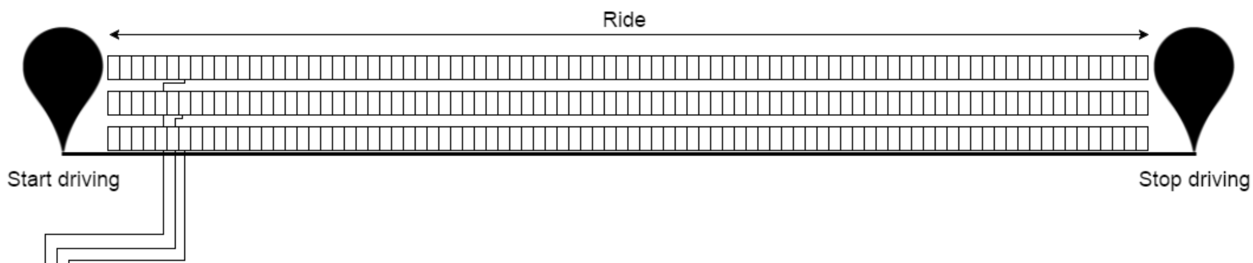
---

[12] Polling: terminal checks if new data has been inserted into the database system. In this case there is new data in the database system the data is pushed to the terminal

[13] Variable: operations are executed as soon as a wireless Wi-Fi connection has been established

[14] CRUD: The four basic database functions Create() (a.k.a Insert()), Read() (a.k.a Select()), Update() and Delete()

[15] Thread(s): simultaneous active database connection(s) from which an operation can be carried out parallel to an operation being carried out in another thread(s).



Id	$GPGGA (GPS time)	$GPVTG (GPS speed km/h)	$GPGGA (GPS position Lat)	$GPGGA (GPS position Lon)
1	172409.00	63.7	63.195764	-45.374012

Id	Processor time	Acceleration x-axis	Acceleration y-axis	Acceleration z-axis
21	172409.00	2.1356897	4.2356879	6.02345698

Id	Processor time	Orientation x-axis	Orientation y-axis	Orientation z-axis
89	172409.00	0.00156257	0.0236587	0.3658745

Fig 16. GPS, Accelerometer and Gyroscopic data in array form. Note: both the GPS and processor time will be synchronized. This means that in the system both values portray the same time parameter.

In order to give an idea about the scale of the CRUD operations that are going to be executed around the database system suppose we have a user that has installed a dedicated sensor system inside his car. According to the GPS module data sheet the maximum sample rate is set at 5Hz (# of samples per second). According to the datasheet for the accelerometer and gyroscope the maximum sample rates are 4000Hz and 8000Hz respectively in normal filter mode [62],[63].

In order to put less stress on the database while maintaining a proper data resolution the sample rates will be limited to 93 Hz (93 Hz is the highest sample rate sensors inside mobile phones can use, in this case used as a reference [5]). When the user, inside his car, drives around for approximately 2 hours before he stops driving in order to take a break the amount of samples that are going to be processed by the local database engine is as follows:

$GPS : 5 * 60 * 60 * 2 = 36.000 \ samples$
$Accelerometer : 93 * 60 * 60 * 2 = 669.600 \ samples$
$Gyroscope : 93 * 60 * 60 * 2 = 669.600 \ samples$
$Total \ amount \ of \ samples : 2 * 669.600 + 36.000 = 1.375.200 \ samples$

Therefore both the local database engine and database system from fig 15 have to be able to transmit and process CRUD operations involving a big amount of records in a reasonable amount of time.

4.3b SQL and NoSQL database structures overview

In general SQL is a term used to refer to all databases that use the query language SQL in order to handle server requests [64]. Among all of the SQL databases only one structure is used to store data called the relational structure [64]. All of the implementable SQL database examples use this structure [64]. In general NoSQL is a term used as a reference to all databases that do not solely use the query language SQL in order to handle server requests [64]. Among all NoSQL databases a few database structures are used to store data: key-value structure, document structure, graph structure and lastly column-family structure [64]. Implementable NoSQL database examples use either one of these structures [64].

In order to determine the data handling effectiveness of these different SQL and NoSQL database structures and associated database examples in relation to the database context presented in the section above SQL and NoSQL database structure types have to be defined. The following database structure sections contain a short description for every SQL and NoSQL database structure type, implementable database examples that use the structure and a graphical example explaining how the sensor data presented in fig 16 can be saved in every database structure type.

There are two reasons why a structures overview has been put into the literature review. The first reason is to give the reader an understanding of every database structure before database examples operating with these database structures are going to be analyzed. The second reason is that the overview provides a basis on how data could be structured in a database example during the design phase of the database part in the participatory sensing system.

4.3b i Relational database

A relational database contains multiple tables, each of which is called a ‘relation’ [65]. The values contained in a single table (relation) are organized into rows (tuples/records). Rows are uniquely identifiable by key values [66] that are commonly stored in the first field (column). Other fields might be filled by other types of relevant information. A relational database only containing one table (relation) is often referred to as a ‘flat-file’ database [65].

If a primary key value from one table appears in several different records of another table there is an implied relationship between the records inside that table [65]. In this case the primary key from one table acts as a foreign key in the other table since foreign keys define relationships between tables by referencing a primary key in another table [66]. A relational database example for the GPS, accelerometer and GPS data presented in fig 16 can be found in fig 17. Notable examples of relational databases include: PostgreSQL, MySQL and Microsoft SQL [24], [66], [22], [26].

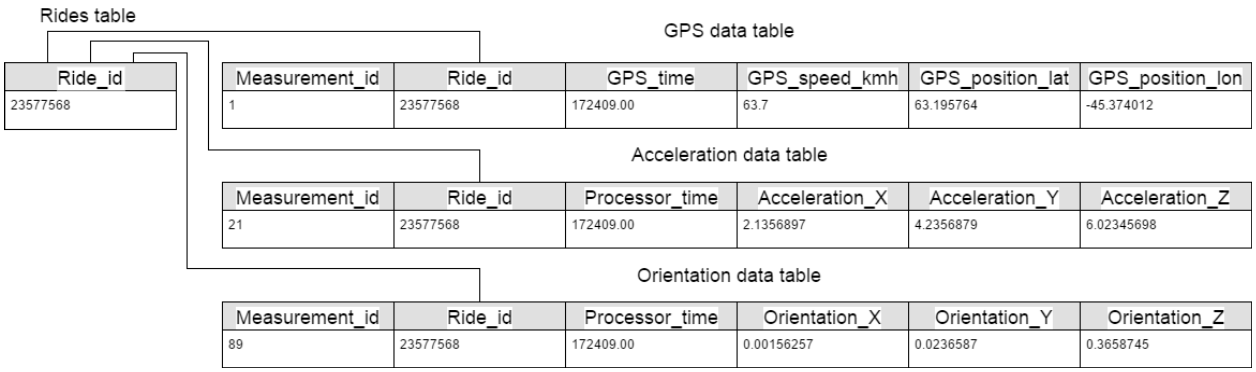


Fig 17. Relational database storage example for sensor data presented in fig 16

4.3b ii Key-value database

Key-value store databases will map a key to a value, or a set of values [67], [68]. A key-value store database will accept any binary value within specific size limits as a key and is capable of storing any binary data as a value [66]. Keys are unique and atomic which means they are indestructible. Keys are used to query the entries in the key-value storage databases [67] as a means of interacting with the database. A key-value database example for the GPS, accelerometer and GPS data presented in fig 16 can be found in fig 18. Notable examples of key-value databases include: Hazelcast, Redis, Membase/Couchbase, Riak, Voldemort and Infinispan [69].

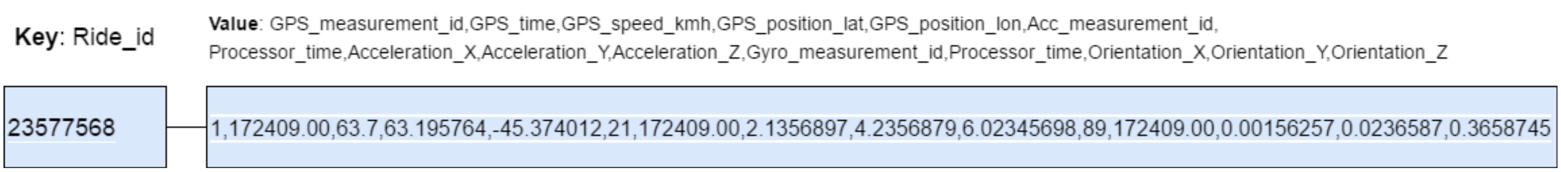


Fig 18. Key-value database storage example for sensor data presented in fig 16

4.3b iii Document database

Document oriented databases store semi-structured data existing in the form of XML (Extensible Markup Language), JSON (Java Script Object Notation) or in other similar formats [70], [66], [68]. If the data is not yet structured in one of these formats some data wrangling has to be done in order to get the data in the appropriate form. A document database example for the GPS, accelerometer and GPS data presented in fig 16 can be found in fig 19. Notable examples of document databases include: CouchDB, MongoDB, RavenDB and Terrastore [69].

Fig 19. Document database storage example for sensor data presented in fig 16 (JSON format)

4.3b iiii Graph database

A graph database gives more importance to the relationship between data items stored in the database rather than the data itself [66], [67]. The data model of a graph database is conceptually quite simple. Nodes (or “vertices”) are connected to each other by relationships (or “edges”). Both nodes and relationships may have properties—sets of name: value pairs [66], [68]. Graph databases are highly optimized for fast traversal and make efficient use of graph algorithms such as the shortest path first algorithm in order to find the link between information stored in the database [67]. A graph database example for the GPS, accelerometer and GPS data presented in fig 16 can be found in fig 20. Notable examples of graph databases include: Neo4J, InfiniteGraph, InfoGrid, HypergraphDB, AllegroGraph and BigData [69].

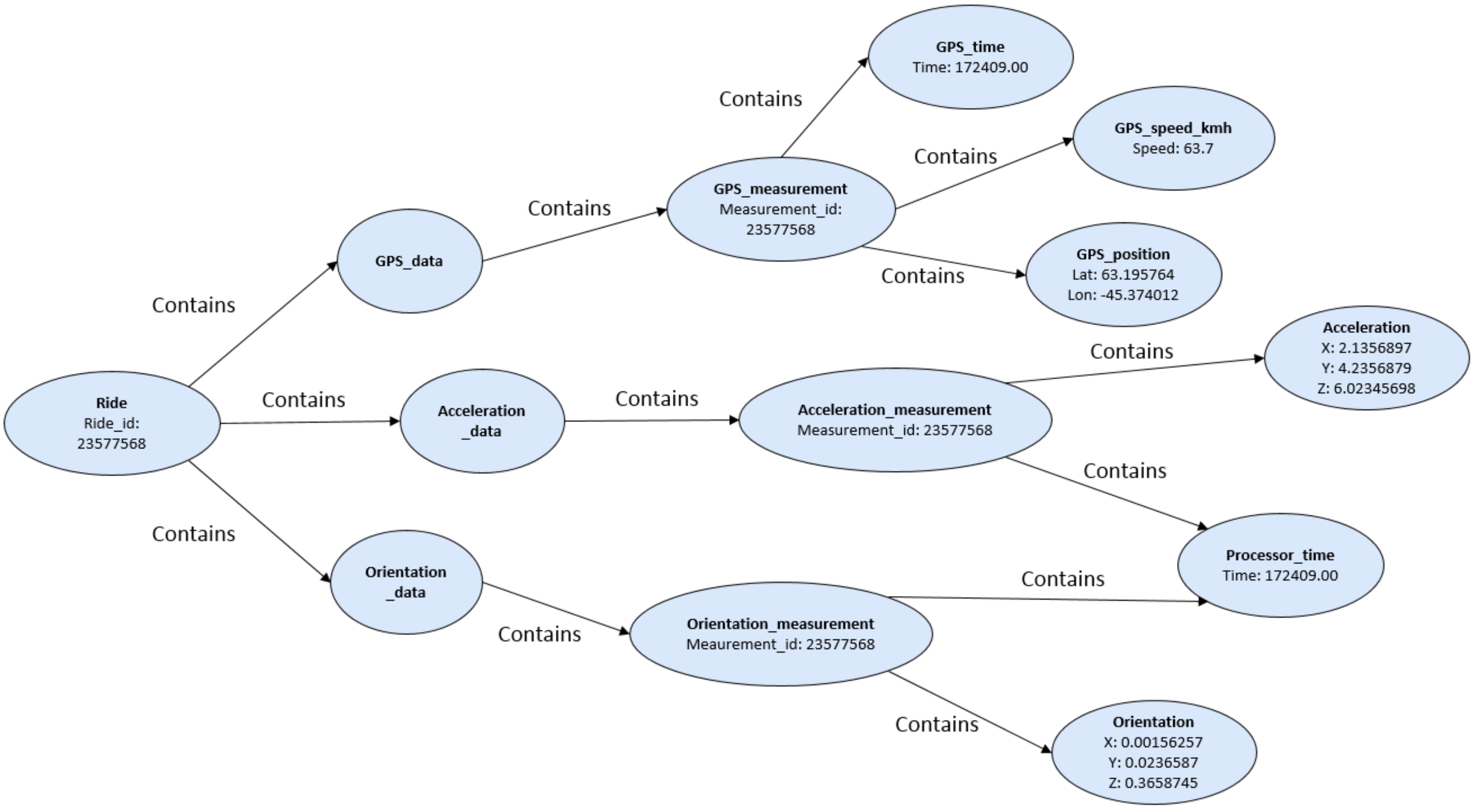


Fig 20. Graph database storage example for sensor data presented in fig 16

4.3b iiiii Column family database

Column family databases are organized in tables (relations) that—like relational tables—have fields (columns) and rows (tuples/records) [66]. In case of the column family database an entire column of a table is uniquely identifiable by a row key. Due to this database structure it is possible to search in only a part, for example a few columns, of a table [67] instead of having to cross all columns in order to compute an analytical query.

A column inside a column family database can have hierarchies of nested columns inside it. A column that contains nested columns is often referred to as a super column [71]. The family aspect of the database refers to the fact that column families are used to group related columns for convenience for storage disk input/output optimization by co-locating columns that are frequently accessed together on a storage disk [66], [67]. A column family database example for the GPS, accelerometer and GPS data presented in fig 16 can be found in fig 21. Notable examples of column family databases include: HBase, Hypertable and Cassandra [69].

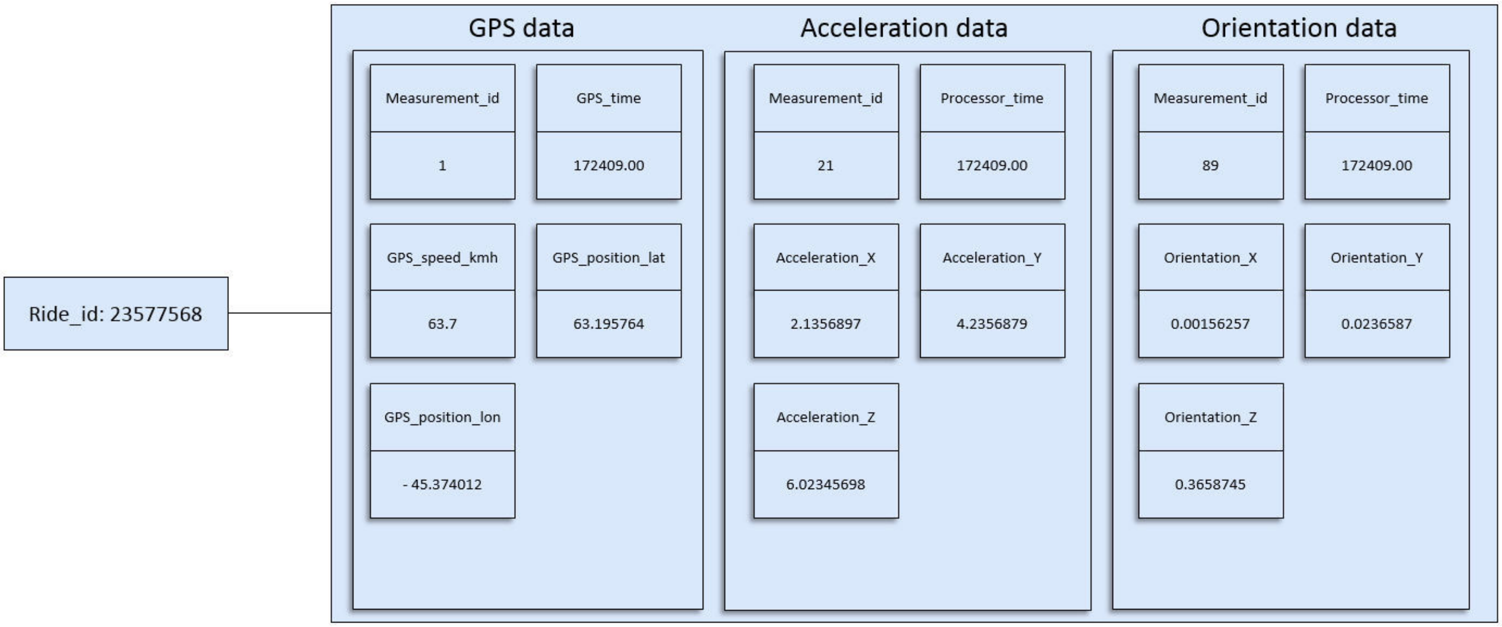


Fig 21. Column family database storage example for sensor data presented in fig 16

4.3c SQL and NoSQL data handling effectiveness results

The evaluation of every database structure and associated database examples in relation to the database operating context is done through a performance comparison of different database handling effectiveness parameters: latency, throughput and CRUD performance time. A comparison is done between implementable database examples that use one of the SQL or NoSQL database structures. The sufficiency of test results for the different databases is determined by the mean performance of most database examples. In case there are clear outliers in both positive and negative context this will be noted during the database evaluation. If no clear distinction is given between types of CRUD operations involved in the analysis it can be assumed that the described result is applicable for both read and write operations.

A few database operation context requirements were determined in the database operating context section. First of all the database system should show reasonable data handling results for an increasing amount of multiple thread connections. Secondly the database should be able to show reasonable CRUD operation performance in three ways database operations are being carried out. The first way is through many operations per second requiring a low amount of records for a single thread connection. The second way is through many operations per second requiring a low amount of records for multi-thread connections. The third way is through a small amount of operations per second requiring a very large amount of records for multi-thread connections. These requirements will be used in order to guide the effectiveness evaluation of database examples according to the database handling effectiveness parameters.

4.3c i Latency

Latency is referred to as a time delay between the cause and the effect of some physical change in a system being influenced by another system [72]. In case of a database environment latency is caused mainly by multiple threads connected to a database. Therefore horizontal scalability directly causes an increase of latency. Latency can be omitted partly through using bulk insert operations when using a NoSQL database structure (data dumping). The discussion paragraphs dealing with latency below are backed up with graphical data that can be found in appendix I fig 43 and 44.

In total there are two papers that report on latency as a data handling effectiveness parameter. In regard to the system requirements Mai et al. [22] reported on latency when bulk insert operations, multi-user write operations and multi-user read operations are performed. Panda et al. [23], [24] reported on latency when different YCSB[16] test workloads in a multi-user environment are performed.

First of all in terms of read latency performance for multiple thread connections it can be noted that the two papers do not agree with each other. Mai et al. [22] state that CouchDB is a clear loser while the performance of the other databases under test (including MongoDB) show sufficient results (see fig 44). Panda et al. [23] suggest that MongoDB is a clear loser while the other databases under test show sufficient results (not including CouchDB) (see fig 43(a)-43(d)).

Secondly in terms of write latency performance for multiple thread connections it can be noted that the two papers agree to some extent with each other. Mai et al. [22] and Panda et al. [23] show that MongoDB is a clear loser (note: one of the sources refers to MongoDB while using multiple sets[17]. For MongoDB using only one set the database shows sufficient results.) (see fig 43(f) and 44). Mai et al. [22] also show that Redis is a bad choice in this case.

[16] YCSB [24]: a framework and common set of workloads for evaluating the performance of different databases through different workloads, divided into letter categories.
YCSB workload A: Update heavy workload having a mix of 50/50 reads and writes.
YCSB workload B: Read mostly workload having a 95/5 reads/write mix
YCSB workload C: Read only having 100% read operations
YCSB workload D: Read latest workload (new records are inserted, and the most recently inserted records are the most popular)
YCSB workload E: Short ranges (short ranges of records are queried, instead of individual records)
Note: workload A, B and C are relevant to the participatory sensing system context since they deal with a lot of different reading and writing operations

[17] Sets: multiple data stores connected to a single database

Lastly it is interesting to note is that Mai et al. [22] reported that for MongoDB and CouchDB it is possible to omit part of the latency by using bulk insert operations (see fig 43(e)). Unfortunately this was not tested for bulk read operations. Upon investigating this some more it was discovered that MongoDB offers a special JSON framework in order to facilitate bulk operations of time series data [73]. This framework can be implemented on every document oriented NoSQL database. If a document oriented database example turns out to be most effective in data handling this framework will be used during the design phase of the database part in the participatory sensing system.

4.3c ii Throughput

Throughput refers to the rate of production or the rate at which something can be processed [72] (Waaij et al. [24] use a certain amount of operations (queries) per second when dealing with database throughput for example). The discussion paragraphs dealing with throughput below are backed up with graphical data that can be found in appendix J fig 45 – 50.

There are three papers that report on throughput. Panda et al. [23] evaluate throughput in terms of kilo-operations per second versus multi-thread client connections according to several YCSB workloads. Waaij et al. [24] evaluate operations per second for different database examples for single and multi-thread situations in which either single or multiple CRUD operations are performed (in case multiple CRUD operations are performed 10,000 records are used at once). Leung and Zhou [25] also evaluate the throughput of several databases according to several YCSB workloads.

The analysis of the three papers, in relation to the database context requirements, of the throughput parameter has been divided into two sections. The first section deals with the throughput performance of many operations per second requiring a low amount of records in case of single and multiple thread connections. The second section deals with a small amount of operations per second requiring a very large amount of records for multiple thread connections only.

In case of the first section it can be noted that there are two papers that do not agree with each other. Panda et al. [23] report that MongoDB is a clear loser while the other databases show sufficient throughput results for all relevant YCSB workloads in case multiple threads are connected (see fig 45). In case of a single thread all of the database performances are very close to each other. Waaij et al. [24] show that for many operations per second requiring a low amount of records in case of a single thread connection MongoDB is a clear winner (see fig 46(b) and 46(d)). In case multiple clients are used Waaij et al. [24] show that PostgreSQL has the best performance (see fig 46(f) and 46(h)). Even though Cassandra has a higher amount of operations per second according to Waaij et al. [24] Cassandra shows a lot of inconsistency for different amounts of connected threads (see fig 46(h)). This is an unwanted behavior in the database operating context of the participatory sensing system since these inconsistencies lead to throughput change when threads start to connect/disconnect from the database operating context of the participatory sensing system.

In case of the second section it can be noted that there are two papers that do not agree with each other. Panda et al. [23] again show that MongoDB is a clear loser while the other databases show sufficient throughput results for all relevant YCSB workloads in case multiple threads are connected (see fig 45). Waaij et al. [24] show that MongoDB is the clear winner in case multiple threads are connected (see fig 46(e) and 46(g)).

Leung and Zhou [25] do not make a distinction between a single and multi-thread environment. What is interesting to note is that in case mostly read or write operations are required (YCSB workload B and C) MongoDB is a clear winner for different amounts of records involved (see fig 48 and 49). In case of a mix between read and write operations (YCSB workload A) it can be noted that Redis and a SQL server are the winner for different amounts of records involved (see fig 47).

4.3c iii CRUD performance time (basic query performance time)

CRUD performance time (basic query performance time), in contrast to throughput, refers to the time it takes to process a single query. This query may consist of different workloads ranging from a few records up until millions of records. The CRUD performance time definition has been based on the figures used for the CRUD performance time evaluation (see fig 51 and 52). The discussion paragraphs dealing with throughput below are backed up with graphical data that can be found in appendix K fig 51 and 52. The CRUD performance, in relation to the database context of the participatory sensing system, should be sufficient across all number of records.

In total there are 2 papers that report on CRUD performance time. Both Truica et al. and Aboutorabia et al. [26],[27] evaluate the performance time of every CRUD operation versus a certain number of records. This evaluation section has been divided into four evaluations. All of the evaluations report on a different type of CRUD operation.

Truica et al. [26] make a distinction between a single and distributed system. First of all the insert operation according to the test results of Truica et al. [26] is analyzed. In terms of the single system all of the NoSQL databases performed well while the SQL databases did not perform at all (see fig 51(a)). In case of the distributed system only MySQL returned a bad performance time (see fig 51(b)). Secondly the read operation is analyzed. In terms of the single system all of the databases returned a sufficient response except the Microsoft SQL server (see fig 51(g)). In terms of the distributed system no analysis was performed. Thirdly the update operation is analyzed. In terms of a single database system CouchDB and MongoDB perform well while the other databases show bad results (see fig 51(c)). In case of a distributed system only CouchDB shows sufficient results (see fig 51(d)). Lastly the delete operation was investigated. In terms of a single system MongoDB and CouchDB show sufficient results (see fig 51(e)). In terms of a distributed system only CouchDB shows sufficient results (see fig 51(f)).

Aboutorabia et al. [27] note that for all CRUD operations MongoDB outperformed Microsoft SQL server (see fig 52(a) – 52(d)). Only when aggregate functions were used Microsoft SQL outperformed MongoDB (see fig 52(e)). What can be noted is that both Truica et al. and Aboutorabia et al. [26],[27] agree on the fact that MongoDB outperforms a Microsoft SQL server in terms of CRUD performance.

4.3d Database decision

In order to pick the database structure and associated database example that is best suited for the database context based on the data handling results it was decided to use a weighted decision matrix [29] (for more information on the weighted decision matrix go to section 4.1c). The weighted decision matrix for the database decision can be found in table 10.

Databases\ parameters	Latency read performance	Latency write performance	Latency bonus	Throughput section 1	Throughput section 2	Throughput workload B, C for different number of records	Create performance time	Read performance time	Update performance time	Delete performance time	Result
MySQL	**2**	**2**	**0**	**2**	**1**	**0**	**0**	**1**	**1**	**0**	**9**
PostgreSQL	**0**	**0**	**0**	**1**	**0**	**0**	**2**	**1**	**0**	**1**	**5**
Microsoft SQL	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**1**	**1**	**4**
MongoDB	**1**	**1**	**1**	**2**	**1**	**4**	**2**	**1**	**3**	**3**	**19**
CouchDB	**0**	**1**	**1**	**0**	**0**	**0**	**2**	**1**	**4**	**3**	**12**
Redis	**1**	**0**	**0**	**0**	**0**	**3**	**0**	**0**	**0**	**0**	**4**
Cassandra	**1**	**1**	**0**	**3**	**2**	**0**	**0**	**0**	**0**	**0**	**7**
Couchbase	**0**	**0**	**0**	**0**	**0**	**0**	**2**	**1**	**2**	**2**	**7**

Table 10. Weighted decision matrix for database decision [29]

The weights of the weighted decision matrix are determined as follows: for every negative performing database 0 will be given, for every mean performing database a 1 will be given, for every positive performing database a 2 will be given and lastly bonuses of 1 can be given for positively influencing situations. All of these weights are given per article. This means that in the end all of the results of the different article are added up together per section and put into the matrix. The database example with the greatest amount of points in the end will be used in the participatory sensing system in case of both the database system and local database engine.

What can be noted in table 1 is that the implementable database examples that use a document oriented database structure like MongoDB and CouchDB perform best on average. Based on this result it is decided that from the implementable database examples using the document oriented structure MongoDB will be used in case of the database system in the database context of the participatory sensing system since MongoDB outperformed CouchDB. This outcome is based on the fact that data from most articles notes that MongoDB performed best on average according to the weighted decision matrix.

The choice for a local database engine is based on the requirement that the local database should mimic the MongoDB framework. The reason for this is that MongoDB does not have a lite version that can be used on a smartphone. Therefore a local database engine based on the

CouchDB, the second database in the database performance result in table 10, API will be used called Couchbase lite [74].

One downside about the results in table 1 is that a weighted decision matrix in this case introduces influence of database appearance across database tests on the result of the weighted decision matrix. However, in this case the appearance positively influences the outcome since more database coverage means more available implementation knowledge and community support. Therefore this influence does not necessarily have to be omitted.

4.4 User interface

In relation to the stakeholder analysis presented in section 3 user interface design requirements were defined. Based on these requirements a set of design recommendations for the data visualization have been determined.

First of all a color legend could be added in order for people to distinguish information according to color on a map. Secondly the user interface should be made in such a way that it is accessible through terminals with different screen sizes. Thirdly people should be able to inspect IRI data. Fourthly functionality could be added to the user interface in which users can actively update the status of detected road anomalies. Lastly people should be able to generate data sets based on the data presented on a map. These design recommendations, together with the basic functionality of the user interface, have been used in order to construct basic user interface functionality diagrams. These diagrams can be found in fig 22 and 23.

What can be noted from fig 22 and 23 is that a distinction is made between active and passive tasks. The difference between the two is that active tasks require user input before a reaction from the user interface is given while passive tasks return a user interface reaction to the user when the system requests such a return.

Active tasks in fig 22 are presented through use of the GOMS method. The GOMS method, as used in fig 22, is described in a paper written by David Kieras called ‘Model-based Evaluation’ [14]. Kieras describes the GOMS method as follows: “GOMS models describe the knowledge of procedures that a user must have in order to operate a system. The user can accomplish certain Goals (G) with the system. Operators (O) are the basic actions that can be performed on the system such as striking a key or finding an icon on the screen. Methods (M) are sequences of Operators that when executed, accomplish a Goal. Selection Rules (S) describe which Method should be used in which situation to accomplish a Goal, if there is more than one available.” [14].

Passive tasks in fig 23 are presented through use of the FAST method. The FAST method was first introduced by Charles Bytheway in the time period around 1960. Charles Bytheway recently published a book in 2007 called ‘FAST Creativity & Innovation’ [13] in which he describes the FAST method as “a technique for enhancing more productive thinking and creativity in order to solve any type of problem. The system components incorporated in a FAST method diagram can be related to more broad system components by asking why questions and to more narrow system components by asking how questions.” [13]. In order to clarify this an example is given based on fig 23.

In order to relate a good iri to the color green (e.g. a narrow system component to a more broad system component) we can ask the following question: why is the given iri a good iri? The answer to this question is that the color inspected on the color legend compared to the iri indicator on the map was green and therefore a good iri.

In order to relate inspecting the user interface to inspecting the color legend and map (e.g. a broad system component to a more narrow system component) we can ask the following question: how does someone inspect the user interface? The answer to this question is by inspecting both the map and color legend associated to that map.

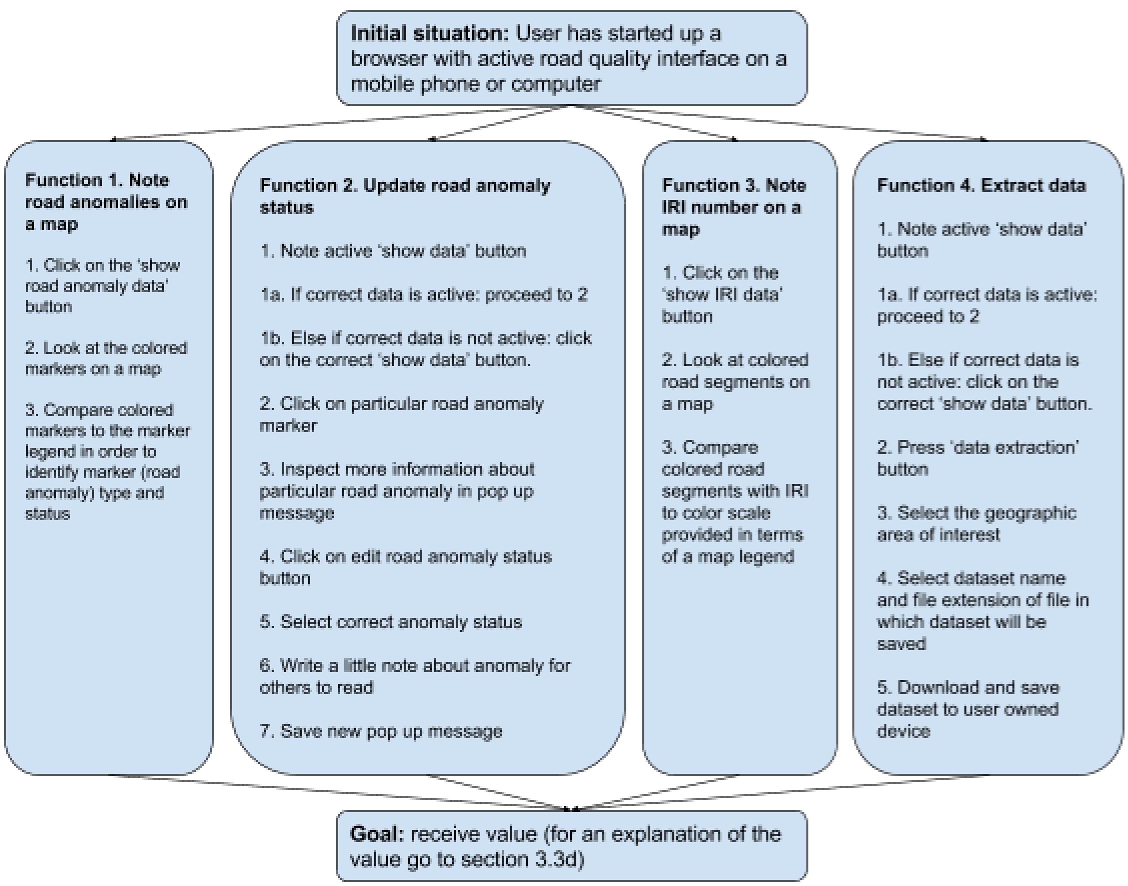


Fig 22. User interface active functionality by means of the GOMS method [13], [14]

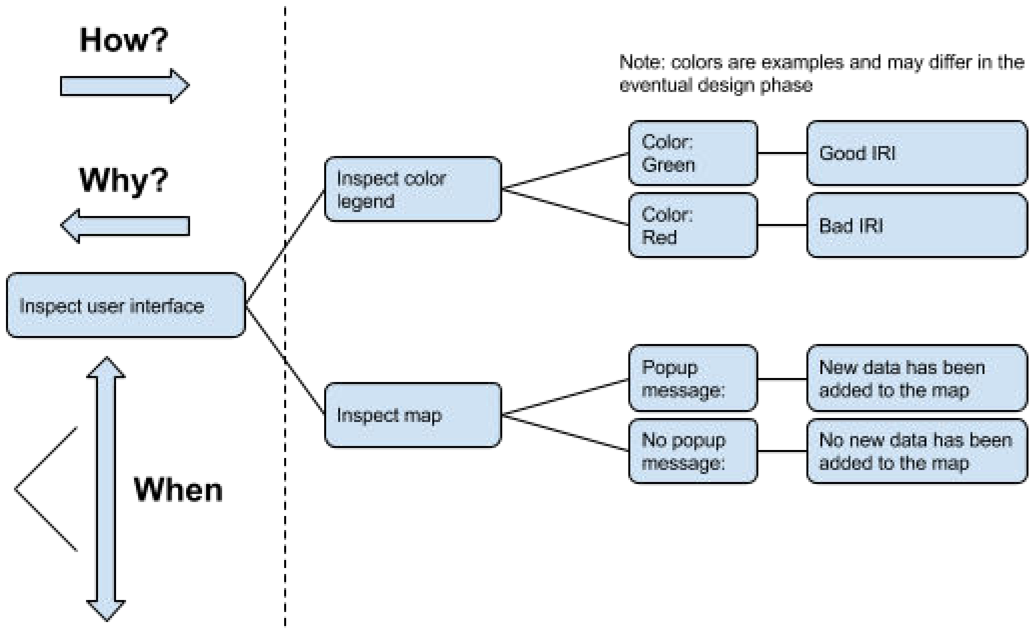


Fig 23 User interface passive functionality by means of the FAST method [13]

In the next section all of the theoretical knowledge obtained in section 4 will be used in order to create a prototype which can be used in order to show how a participatory sensing system might work once it is implemented.

5. System design

The system design section has been divided into the different technical components that together make up the participatory sensing system prototype. Every section explains how the technical component works and describes pitfalls that were encountered in the system design for that specific technical component.

5.1 Dedicated sensor system

During the system design phase the choice was made to develop the dedicated sensor system prototype component based on the arduino platform instead of the thunderboard react platform. The reason for this can be found in the pitfalls section. The hardware components that make up the dedicated sensor system prototype together with the different versions that were developed can be found in fig 24.

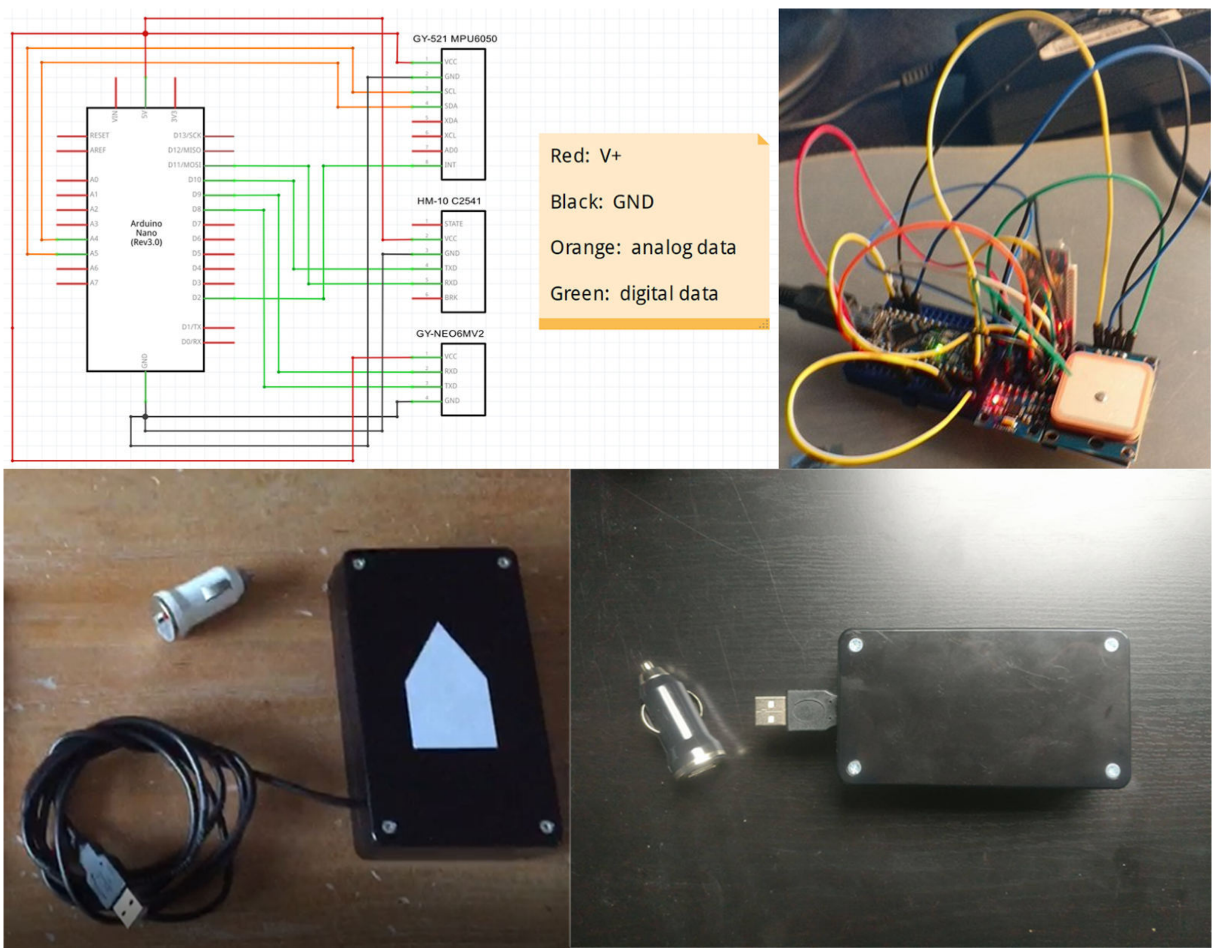


Fig 24. (up-left) dedicated sensor system hardware schematics (up-right) prototype version 1 (down-left) prototype version 2 (down-right) prototype version 3

A list of all the different components that were used and changed in later versions can be found below:

- Arduino nano (atmega328 chip version)
- HM-10 cc2541 bluetooth low energy module
- GY-521 MPU6050 IMU (Inertial Measurement Unit)
- GY-NEO6MV2 GPS module
- Breadboard wires -> changed to standard conducting steel wires from version 2
- USB male to USB mini male cable -> changed to an adapter in version 3
- Universal USB car socket charger -> included from version 2
- Universal black ABS enclosure -> 142x80x30 mm in version 2 and 112x62x31 mm in version 3

What can be noted in fig 24 is that the usb car socket charger is not attached to the dedicated sensor system enclosure. The reason to keep the universal usb car socket charger apart from the internal dedicated sensor system prototype is that it allows for a choice to mount the dedicated sensor system on either a car cigarette charger or usb port. The reason for this choice is that not all cars that are currently developed have a car cigarette charger or usb port available. If either one is or both are available inside a car the user can just pick the option that best suits the situation.

The software that was developed in order to control all of the components from the arduino nano basically represents a basic state machine. This state machine sequentially acquires and transmits sensor data in loop iterations under the condition that the system is in the appropriate state. A UML state machine diagram [15] representing the software can be found in fig 25.

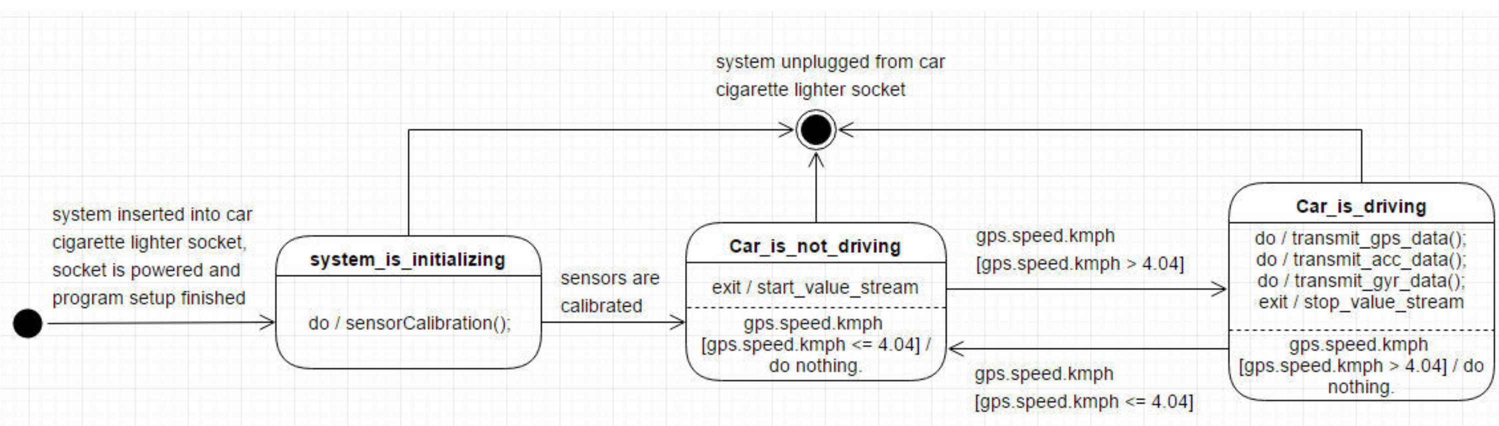


Fig 25. UML state machine diagram [15] representation of dedicated sensor system software

In fig 25 it can be noted that there is a setup before the system goes into the first state. This setup consists of setting up the serial connection between the arduino nano and cc2541 ble chip

in order to be able to transmit data, setting up a serial connection to the gps module in order to retrieve gps data and to initialize the accelerometer and gyroscope part of the IMU.
The road anomaly classification and iri computation script that can be noted at the beginning of section 4 was developed in such a way that the imu sensor data x-axis and z-axis of the dedicated sensor system prototype had to be switched with each other. This has to do with the fact that the developers of the script were using a mobile phone as a sensor system and had to fixate the mobile phone in a certain position (for more reference material on this check [5]). Therefore the mobile phone orientation was changed and thus the acceleration and orientation sensor x-axis and z-axis were switched.

The start_value_stream and stop_value_stream procedures that can be noted in fig 25 are executed using a boolean control system (see appendix L) that executes its procedures before the state machine executes its procedures per loop iteration. In order to get a better view on how the exact functions and state machine logic has been implemented take a look at the source code in appendix L.

Even though the gps speed threshold of 4.04 km/h that can be noted in fig 25 appears to be a random number it is not. The reason why this specific number has been chosen is explained in section 5.1b.

5.1a Pitfalls

The reason that the choice was made to switch from the thunderboard react platform to the arduino platform has to do with experience and reference material. Reference material, apart from a few forum posts and manuals, could not be found. As a result, due to a lack of experience with C, the language the thunderboard react platform code was written with, the chance of finishing the dedicated sensor system on time with the thunderboard react platform faded. Arduino on the other hand does have a good amount of reference material and the experience, based on already using arduino from 2014 onward, was already there.

What can be noted in the dedicated sensor system hardware component list is that, instead of using an accelerometer and gyroscope, an IMU was used. Also the gps module was chosen in such a way that the gps antenna is mounted onto the pcb with supporting electronics. The reason for this was to save space in the universal ABS enclosure that was necessary to fit in the complete dedicated sensor system. This was necessary due to the fact that too big enclosures could not be mounted onto a car cigarette socket charger due to shape and weight. This situation was encountered during the development of version 2 of the prototype. Even after the development of version 3 of the prototype it was noted that the prototype did not fit into all car cigarette socket chargers. This occurred due to the fact that some car dashboards have been made in such a way that the space between the car cigarette charger and switch gear is extremely small. In order to solve this future work has to be done into making the prototype even smaller.

When the sensor data of the IMU was first encountered it was noted that the data retrieved was not meaningful data (e.g. the raw acceleration data was not reported in m/s^2 and the orientation data was not reported in rad/s). The fix for this was found in a stackoverflow post created by Bence Kaulics and Ajmal Moochingal [75] where it was discovered that the raw value first has to be altered by a scaling factor in order to get the values in g for acceleration deg/s for orientation. Afterwards these values had to be changed with a conversion ratio in order to get acceleration in m/s^2 and orientation in rad/s.

The timestamp, the data type used to relate all the other data types together, source for acceleration and orientation data is different from the gps timestamp source. As a result the acceleration/orientation timestamps and gps timestamps would start to drift away when the dedicated sensor system was turned on. In order to fix this it was determined that one of the timestamp sources had to be synchronized to the other source. Initially this was done using an arduino time library. The downside of this library and also other time libraries is that they can only synchronize time with second precision. Since the dedicated sensor system requires millisecond precision these time libraries did not perform good enough. In order to fix this the decision was made to synchronize the smartphone system time with the gps time of the internal gps module in the smartphone and timestamp the data at the smartphone application side once it would arrive via bluetooth low energy. In order for this to work it was assumed that the bluetooth low energy transmission delay between the dedicated sensor system and the smartphone application was lower than 1 millisecond and therefore did not influence the timestamp data.

After testing the dedicated sensor system for some time, the hm-10 bluetooth low energy module started to act weird. This was noted at the smartphone application side where, after connecting to the hm-10 module, the smartphone application was unable to discover services. After some time a dirty fix was found. This dirty fix includes resetting the hm-10 module and deleting the cached data about past connections using AT commands. This fix is a dirty fix due to the fact that it had to be repeated every time after using the dedicated sensor system a few times.

5.1b Gps speed threshold

In section 1.4 it was determined that data acquisition at the dedicated sensor system will be triggered once the gps speed is above a certain threshold and stopped once the gps speed is below a certain threshold. A speed threshold can also be noted in section 5.1 fig 25. The reason for this has to do with the fact that the gps module used in the participatory sensing system prototype at rest keeps reporting a different position every time the gps position information is updated. This phenomenon is also known as gps drift. This phenomenon can be noted in the gps deviation maps in fig 26.

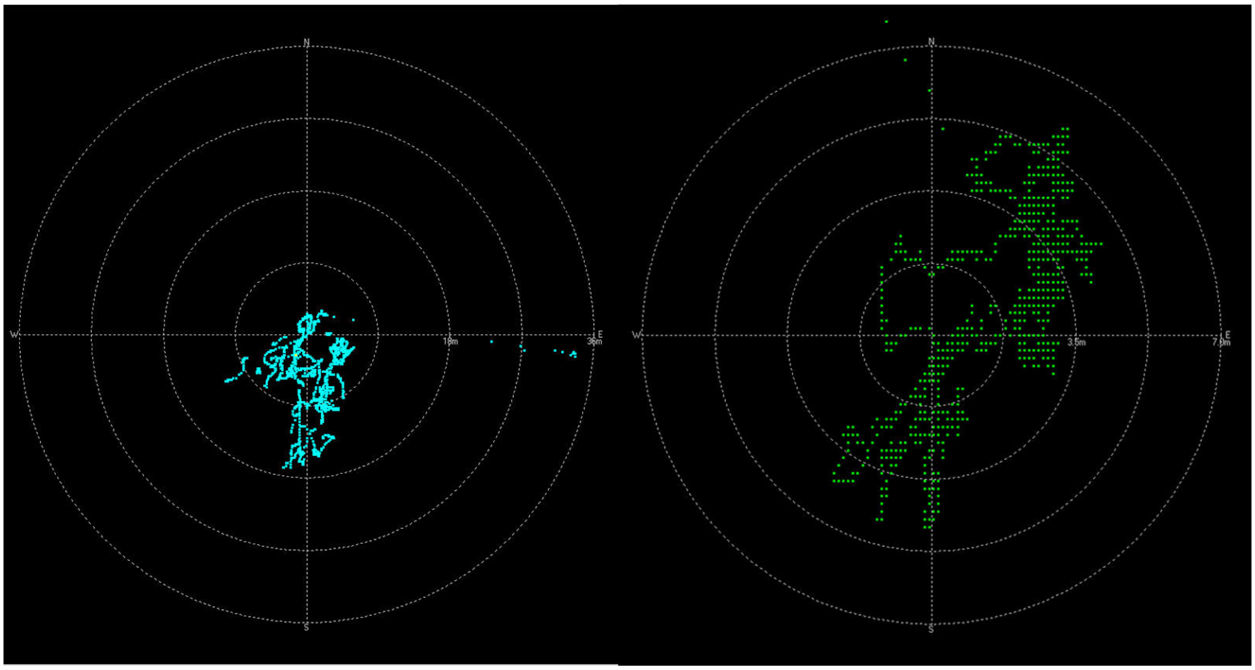


Fig 26. GPS position deviation map [16] (left) inside a house (right) outside a house for gps data at rest. Note: the theoretical ideal value would be if every dot on the map would be perfectly placed in the middle of the map in case of the gps module at rest.

Ideally the gps should keep reporting 0 km/h at rest and as soon as this value increases sensor data will start to be sampled. If this approach would be taken in the real situation where gps drift is happening data would be sampled even in the case that the car is not driving. Therefore the speed threshold is a vital part in establishing good quality sensor data that is only sampled when the car is driving and using the road. Only then the sensor data is able to accurately report on the quality of the road the car is currently driving on.

In fig 26 it can be noted that a gps module inside a house, and therefore not having a clear line of sight to satellites, has a far less ideal position approximation than a gps module outside a house, and therefore having a clear line of sight to satellites. Due to the fact that the gps module was placed in an enclosure in order to be fixed on a cigarette socket charger (see section 5.1) the decision was made to find a speed threshold for the case a gps module is situated inside a house and therefore will not have a clear line of sight to satellites.

The gps drift causes the reported gps speed by the gps module not to be steady 0 km/h at rest but fluctuating very close to 0 km/h. A set of sample gps speed measurements at rest were taken. Based on this set the histogram in fig 27 was made.

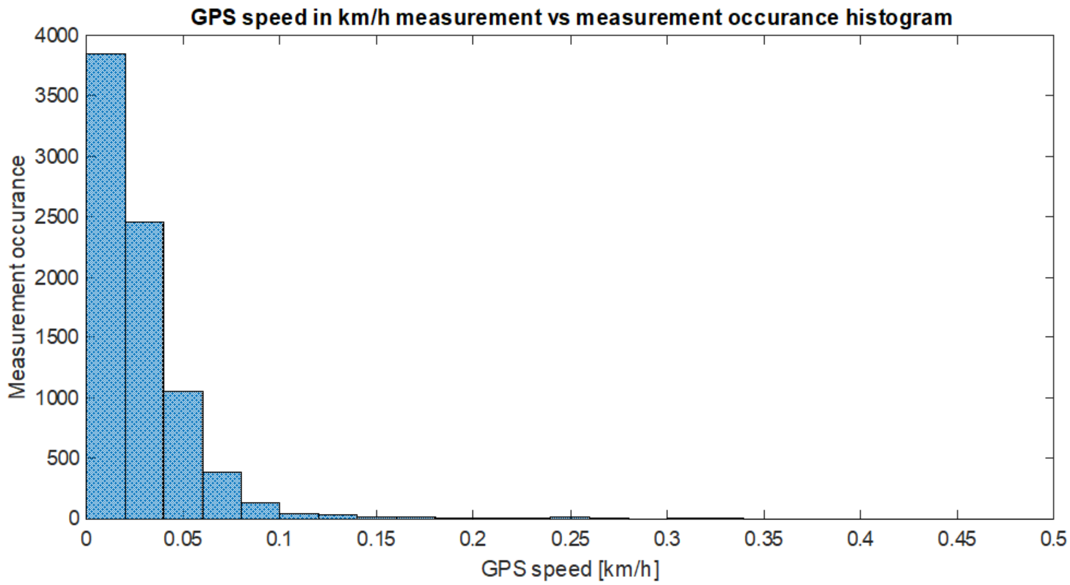


Fig 27. Measurement occurrence histogram [17] for different gps speeds at rest in gps speed data sample.

In order to graphically link the speed threshold to the histogram in fig 27 an example is sketched. The speed threshold could ideally be seen as a vertical line inside the histogram placed in fig 27 in such a way that all (100%) of the gps speed measurement values reported in fig 27 stay at the left side of the vertical line (e.g. are lower than the threshold) so that they do not influence the data sample trigger procedure. All of the data reported in fig 27 could sketch two scenarios: the car is moving very slowly or is at rest. For these values it is indistinguishable which one of these scenarios is the case.

Due to the fact that the reported gps speeds appeared to be reported in random and fluctuating km/h values the decision was made to find a speed threshold based on statistical theory. Before the appropriate theory could be found for computing the speed threshold the data set was tested in order to see if the data set was a normal distribution or not. The results can be found in fig 28.

```
        Shapiro-Wilk normality test

data:  gpsDriftData$V1
W = 0.12512, p-value < 2.2e-16
```

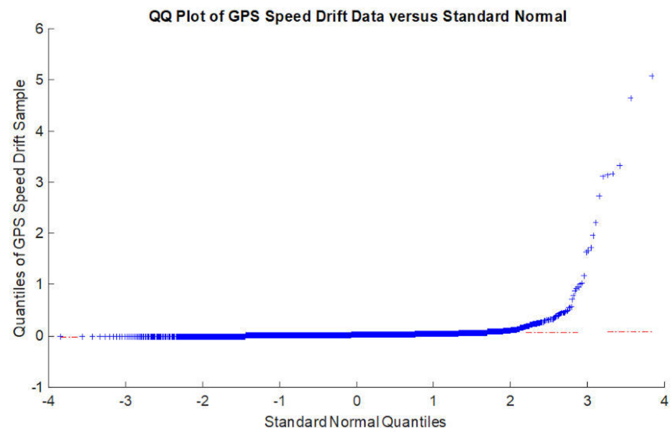


Fig 28. (left) Shapiro wilk normality test [18] and (right) normal qq plot [17] for gps speed data sample

In the shapiro wilk test [76] the hypothesis that the gps drift data is normally distributed (a.k.a the null hypothesis or H0) is tested against the hypothesis that the gps drift data is not normally distributed (a.k.a H1). The significance level used for this experiment, also known as α, is 5% (e.g. 0.005). In fig 28 left it can be noted that the p-value is extremely low and approximates zero. Since p-value < 2.2e-16 < α = 0.05 we reject the null hypothesis and say that the gps drift data is definitely not normally distributed.

Based on the fact that the data is not normally distributed there is a statistical theory called 'Chebyshev's inequality' that can be used in order to compute a speed threshold. Chebyshev's inequality states that "Generally, for any number k greater than 1, at least ($1 - 1/k^2$) of measurements will fall within k standard deviations of the mean [i.e., within the interval ($\bar{x} - ks$, $\bar{x} + ks$) for samples and ($\mu - k\sigma$, $\mu + k\sigma$) for populations]." [19]. What does this mean for the measurement occurrence diagram in fig 33? A graphical example can be noted in fig 29.

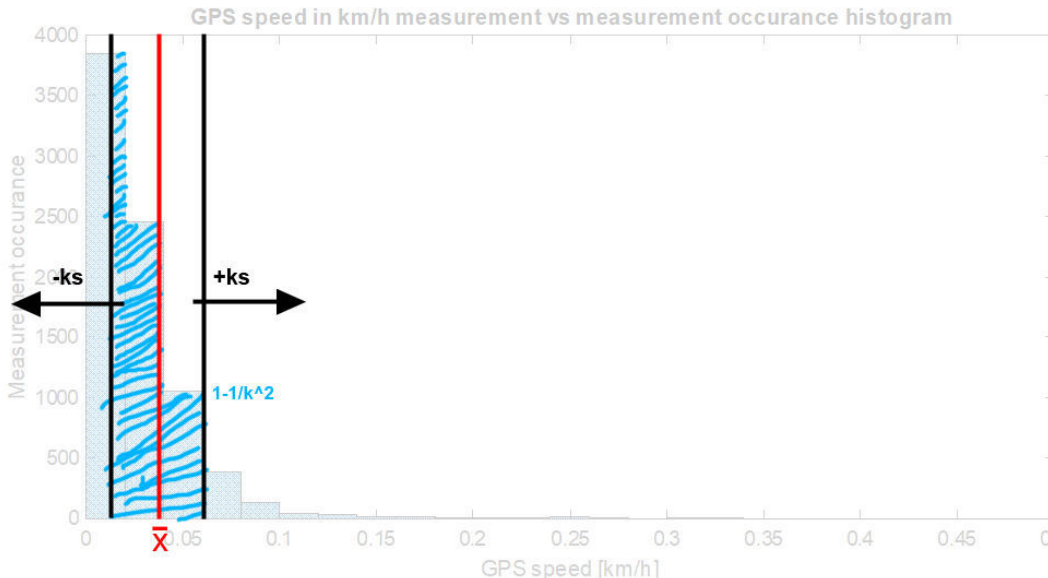


Fig 29. Graphical example of Chebyshev's inequality [19] for measurement occurrence histogram [17] in fig 27.

What can be noted from fig 29 is that the eventual speed threshold is going to equal $\bar{x} + ks$ (e.g. the black line to the right of the red line in fig 29). $\bar{x}$ (a.k.a the sample mean) and s (a.k.a the sample standard deviation) can be found in fig 30.

Name	Value
gpsDriftData	*8097x1 double*
mean	0.0418
std	0.1263

Fig 30. Sample mean and standard deviation [17] for gps drift data sample.

Unfortunately no number k exists for which 100% of measurements fall within k standard deviations of the mean. Therefore a number k is computed so that at least 99.9% of measurements will fall within k standard deviations of the mean:

$$1 - \frac{1}{k^2} = 0.999$$

$$-\frac{1}{k^2} = -0.001$$

$$\frac{1}{k^2} = 0.001$$

$$0.001k^2 = 1$$

$$k^2 = 1000$$

$k = \sqrt{1000} \approx 31.623$

Now that $\overline{x}$, s and k are known the speed threshold can be calculated:

$Speed\ threshold = \overline{x} + ks = 0.0418 + (31.623 * 0.1263) \approx 4.04\ km/h$

5.1c Acceleration and orientation sensor calibration

Due to the orientation of the acceleration and orientation sensor used (for an explanation please take a look at section 5.1) data retrieved from the imu the ideal values measured at rest are as follows:

Acceleration x-axis: 9.80665 m/s^2
Acceleration y-axis: 0 m/s^2
Acceleration z-axis: 0 m/s^2
Orientation x-axis: 0 rad/s
Orientation y-axis: 0 rad/s
Orientation z-axis: 0 rad/s

When the acceleration and orientation sensor data values were first sampled at rest a strange offset for all of the acceleration and orientation values was noted. The offsets can be noted in fig 31 and 32.

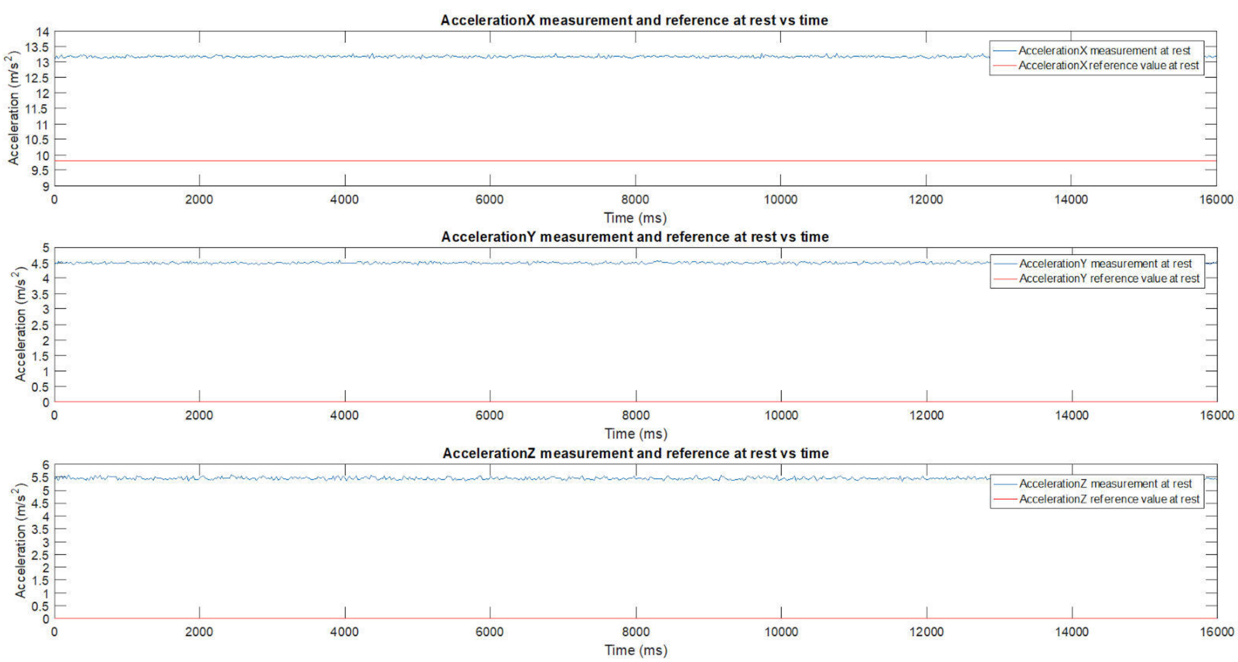


Fig 31. Acceleration values with offset compared to theoretical ideal value at rest in case acceleration is not calibrated [17]

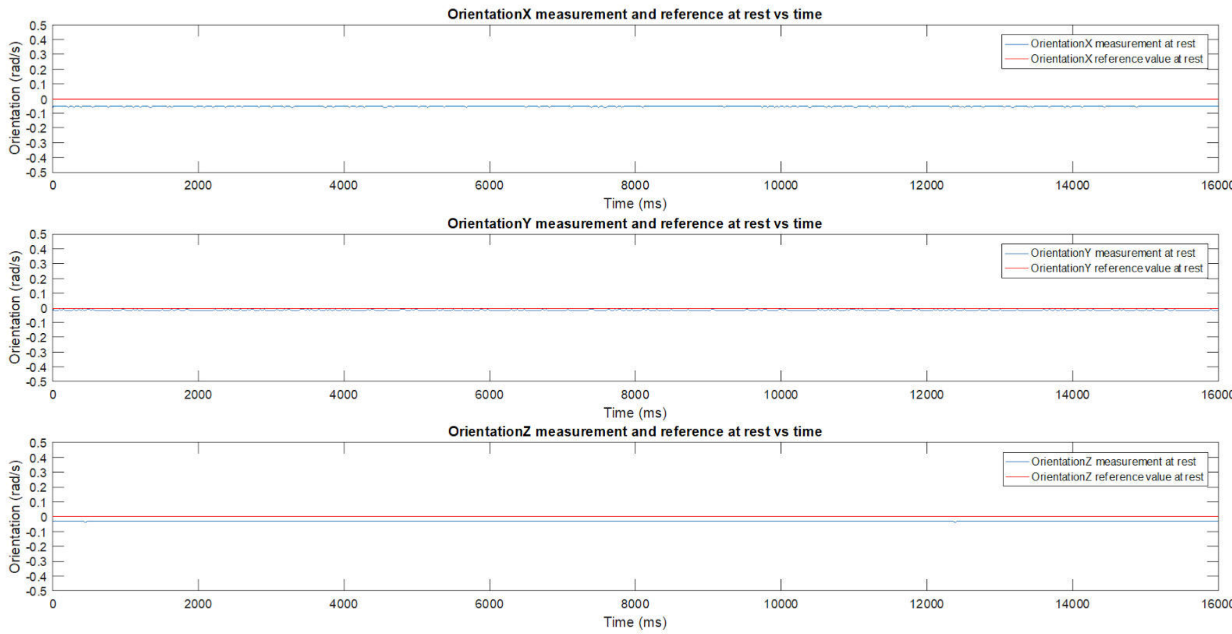


Fig 32. Orientation values with offset compared to theoretical ideal value at rest in case orientation is not calibrated [17]

Eventually it was noted that the imu of the dedicated sensor system first had to be calibrated before good quality acceleration and orientation data could be sampled. As a result the offset would be compensated in order to get the sampled acceleration and orientation value as close as possible to the theoretical ideal value. The only downside to the calibration was that it would take time before the dedicated sensor system could start to sample or wait for the user to start driving a car. Therefore another objective for the calibration, apart from approximating an ideal value as close as possible, was to do it as fast as possible. The reason why calibration is performed every time the dedicated sensor system is turned on is that the dedicated sensor system can be inserted into a car cigarette lighter socket that causes the dedicated sensor system to be fixed into the car cigarette lighter socket at a slight angle. This angle can be different for every car cigarette lighter socket in a car.

The library that was used in order to receive samples from the imu used in the dedicated sensor system already had functionality in order to set an offset value. The only thing that had to be determined was a function in order to compute an offset value. The first function that was realized incorporates sampling a large set of values with offset, computing an average from the set and using that in terms of an offset value. Initially this function did not work but after some time it was discovered, according to a more advanced calibration function created by Tinker Guy [77], that the offset values based on the values with offset first had to be mathematically edited before they could be used in terms of appropriate offset values. The result can be noted in fig 33 and 34.

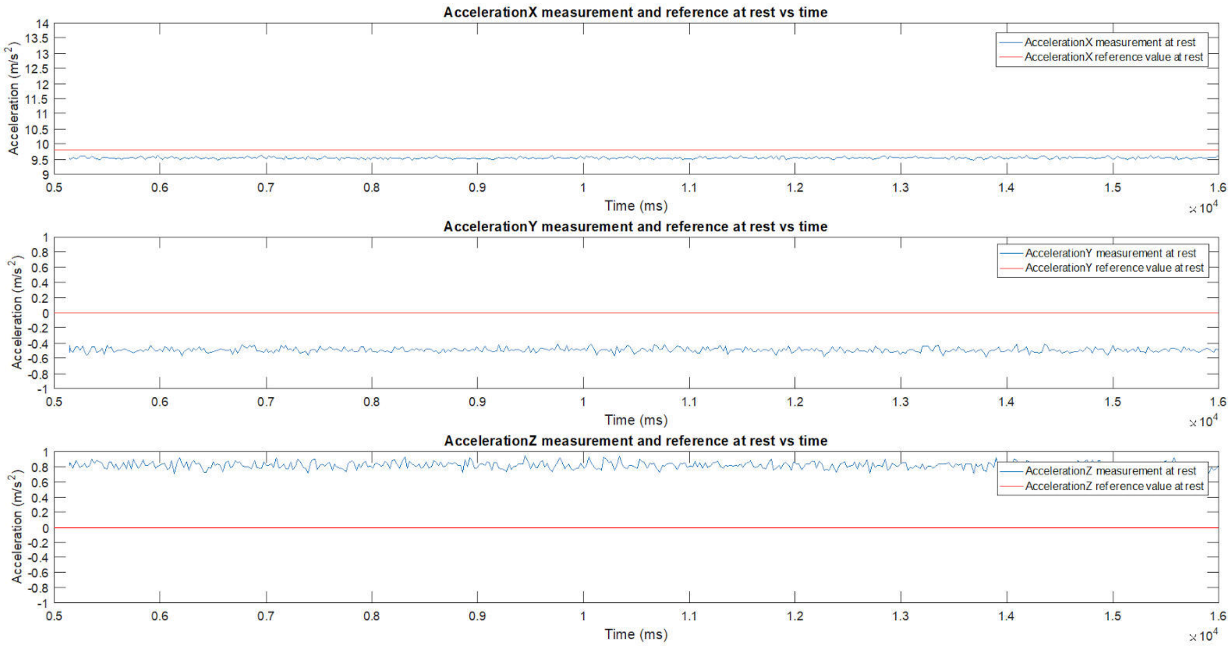


Fig 33. Acceleration value compared to theoretical ideal value at rest in case acceleration is calibrated with simple calibration operation [17]

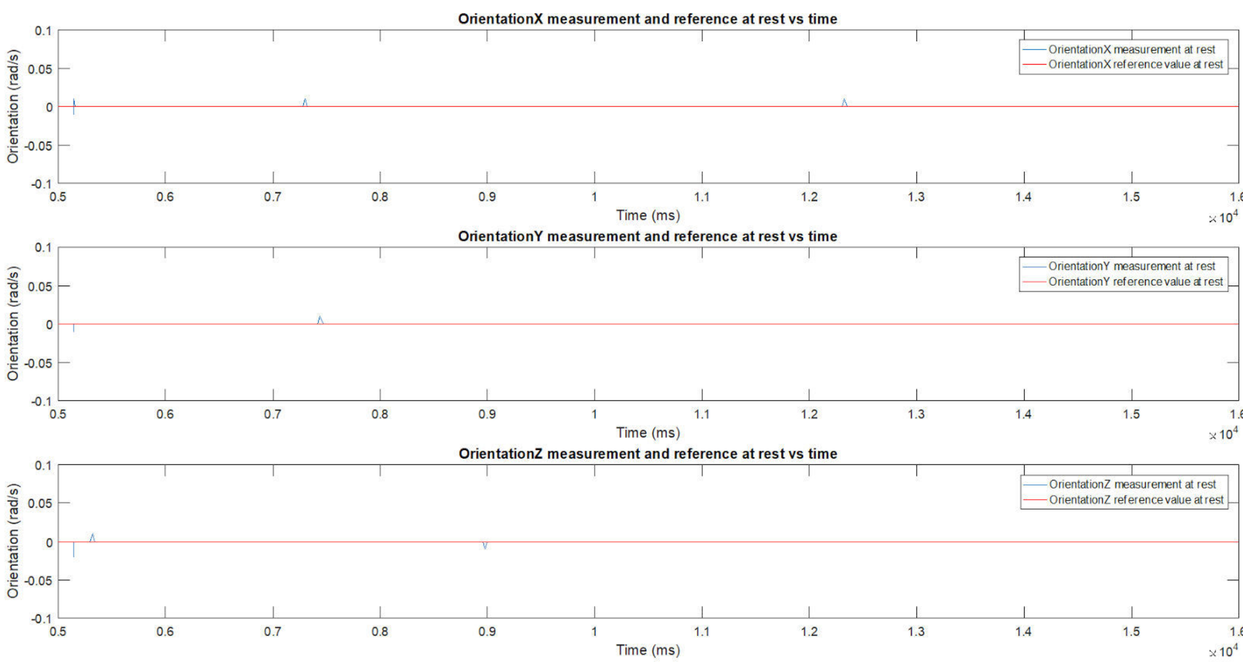


Fig 34. Orientation value compared to theoretical ideal value at rest in case orientation is calibrated with simple calibration operation [17]

The time it took to perform a simple calibration operation involving mathematically edited average values with offset was 5.14 seconds. What can be noted in fig 33 and 34 is that the orientation calibration is near perfect while the acceleration calibration still shows a minor deficiency.

Afterwards the calibration method of by Tinker Guy [77] was tried. The calibration sketch did not return any offset correction values. Eventually it was discovered that this has to do with the fact that the calibration sketch uses the imu library with standard axis orientation. In case of the dedicated sensor system the x-axis and z-axis of this library were flipped and therefore the calibration sketch was unable to return offset values. In an example that is given by Tinker Guy [77] it could be noted that the calibration takes approximately 23 seconds and is therefore much slower than the simple calibration based on edited average values with offset. Therefore the simple calibration based on edited average values with offset was used in the dedicated sensor system. The specific function can be noted in appendix L. The function is called 'sensorCalibration'.

5.1d Dedicated sensor system user acceptance test

In order for test pilots to be executed with an implementation of a participatory sensing system in the future and therefore acquire more data among test users using a dedicated sensor system a sufficient willingness for different people to participate in a participatory sensing system pilot by inserting the dedicated sensor system inside a car cigarette socket has to be realized. After the development of version 2 of the dedicated sensor system it was believed that this version would have been unable to provide such a willingness due to its size. As a result version 3 of the dedicated sensor system was developed with a smaller size factor.

In order to find out to what extend a particular dedicated sensor system enclosure shape size plays a role in the willingness of users to place the dedicated sensor system inside their car during a test pilot a user acceptance test was executed through the use of a Google form. During this test people were asked to look at a picture containing dedicated sensor system versions 2 and 3. Afterwards people were asked which dedicated sensor system they would pick if they were invited to participate in a future test pilot and why. The result to this question can be found in fig 35. The user acceptance test can be found in appendix P.

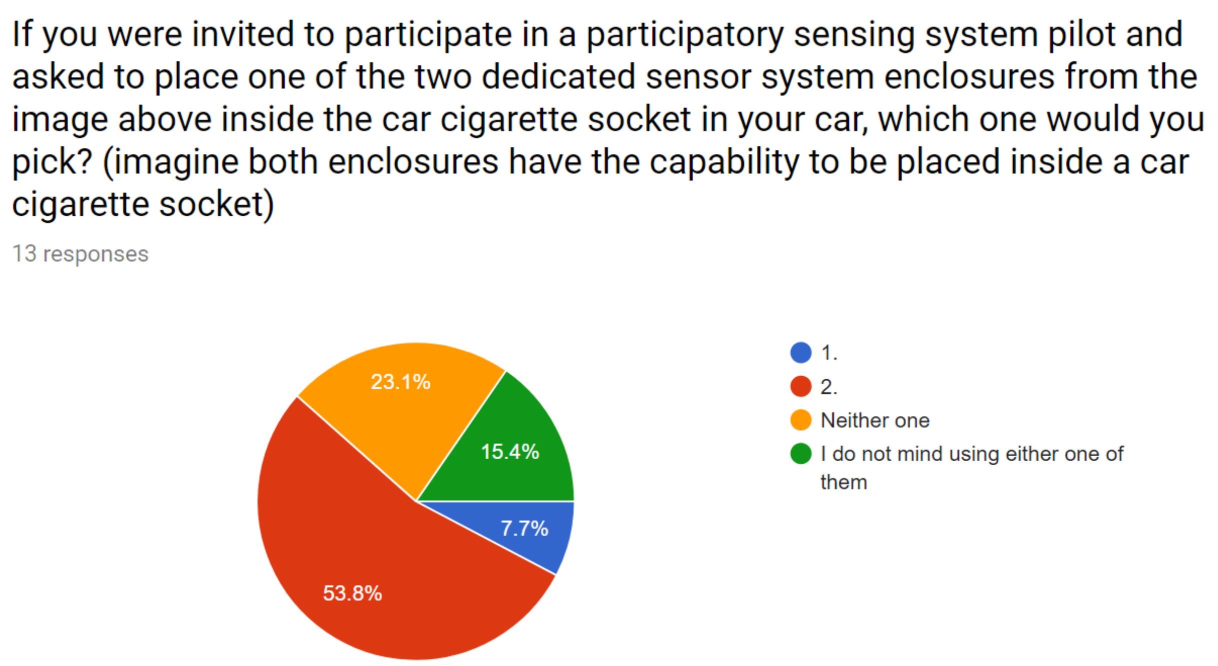


Fig 35. User acceptance test result of question about dedicated sensor system choice. Note: 1. corresponds to dedicated sensor system version 2. 2. corresponds to dedicated sensor system version 3.

What can be noted in fig 35 is that most of the people that participated in the user acceptance test chose for the dedicated sensor system with the smaller form factor. Therefore the change to a smaller form factor from version 2 to version 3 has a positive effect on the amount of people willing to participate in a test pilot.

5.2 Smartphone application

After the dedicated sensor system was finished it was time to move on to the development of the android smartphone application prototype. An overview on how the user would perceive the app, e.g. the front-end, can be found in the User Experience (UX) diagram in fig 36.

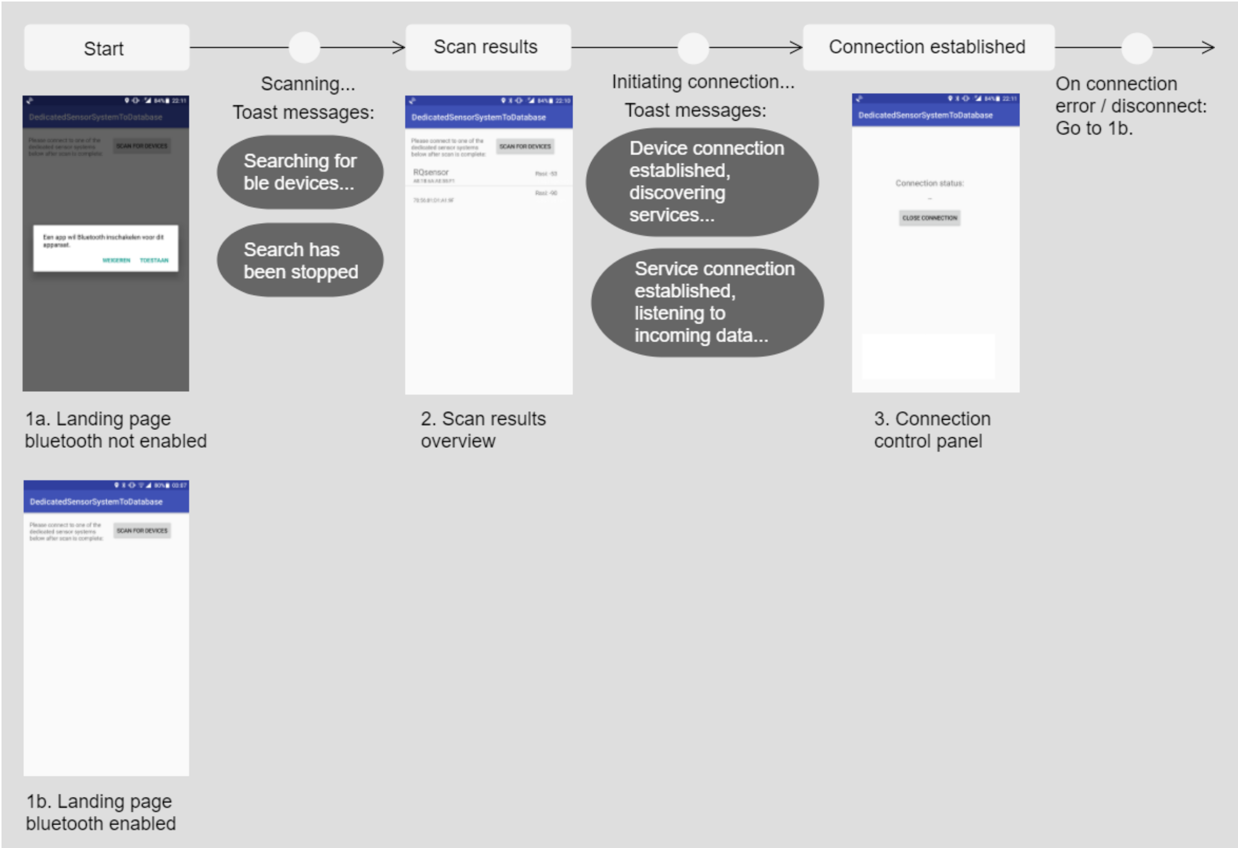


Fig 36. Android smartphone application UX diagram

Due to the fact that the back-end of the smartphone application is rather complex the decision was made to split up the prototype explanation in several parts and to focus on the vital parts that will help in order to answer the research questions specified in section 1.3. Firstly a helicopter overview of the vital parts of the back-end in terms of a UML component diagram [15] is shown in fig 37.

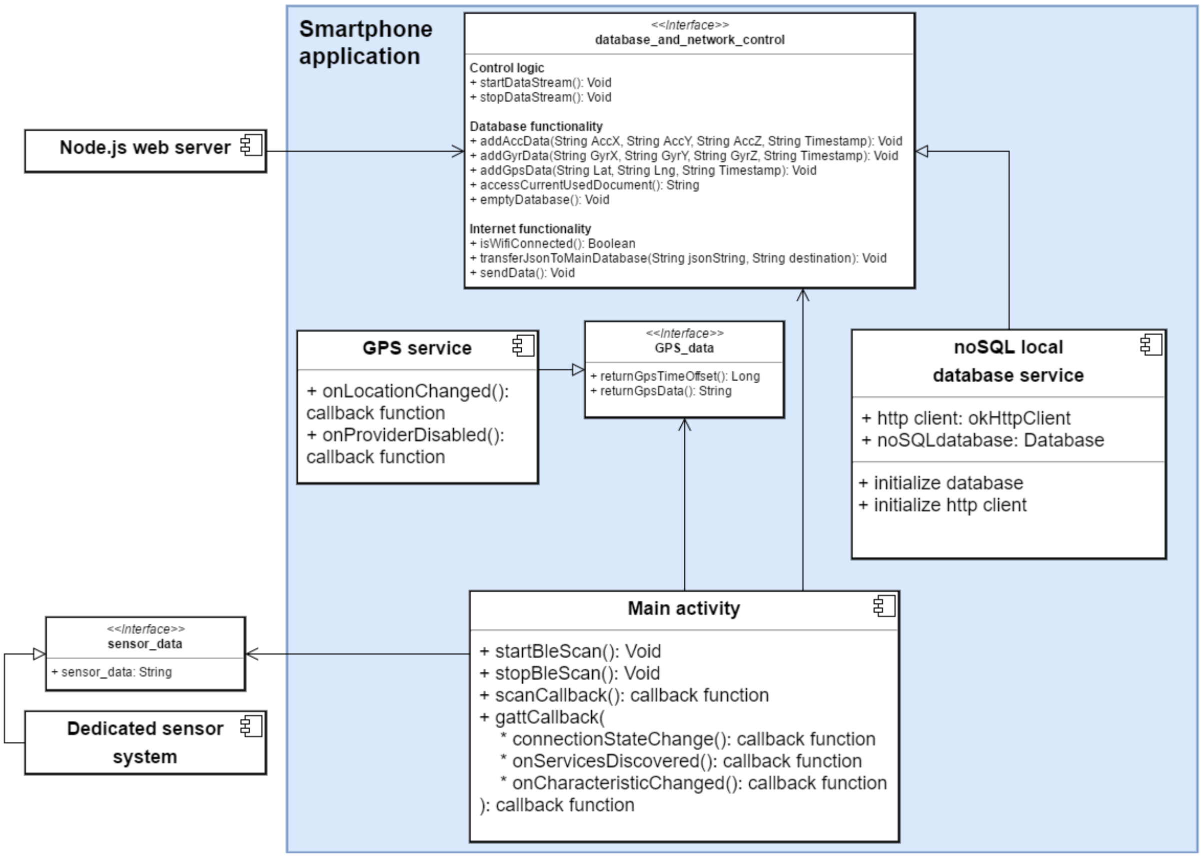


Fig 37. Smartphone application UML component diagram [15] of vital parts

The way the service functions are accessed by the main activity is by binding the main activity to these services. That way the main activity can access the service functions listed in the given interfaces in fig 37. The service functions are executed in the background and do not interfere with the user interface.

In order to automate the internet functionality between the Node.js web server and noSQL local database service located in the smartphone application a handler has been implemented that executes the sendData function based on a fixed time interval. The sendData function checks if wifi is connected, if there is no active data acquisition going on between the dedicated sensor system (this is done by checking of the accessCurrentUsedDocument function is not equal to null) and smartphone application and if there are documents stored in the local database. If all of these conditions are true the sendData function retrieves all of the documents stored in the database, sends these documents to the Node.js web server through a http POST request at a given ip and port and afterwards deletes all of the documents from the local database.

Before data can be acquired first a connection with a dedicated sensor system has to be established. The scan and connection functions with associated callbacks are called based on the input from the user interface (for the user interface see fig 26). Therefore the only part of the participatory sensing system prototype that requires manual input is connecting to a dedicated sensor system. All of these functions with associated callbacks can be found in the 'MainActivity.java' file in appendix M. The vital functions can be noted in the main activity component in fig 37.

The scanCallback function will return a set of devices the user can connect to into a list. This list can be noted in fig 36 frame 2. When the user clicks on a dedicated sensor system (e.g. RQsensor) the smartphone application will connect, discover services and subscribe to a characteristic in order to listen for updates.

The callback function that brings the data storage functionality of the smartphone application together once a connection to a dedicated sensor system is established is the onCharacteristicChanged function. Based on the received input from the dedicated sensor system this function creates a new document through the startDatastream function, closes off a document using the stopDatastream function and updates the current used document with incoming sensor data. Closing off a document means that the document is not accessible anymore from the onCharacteristicChanged function. The way the onCharacteristicChanged function handles incoming sensor data strings is explained in the flowchart in in fig 38.

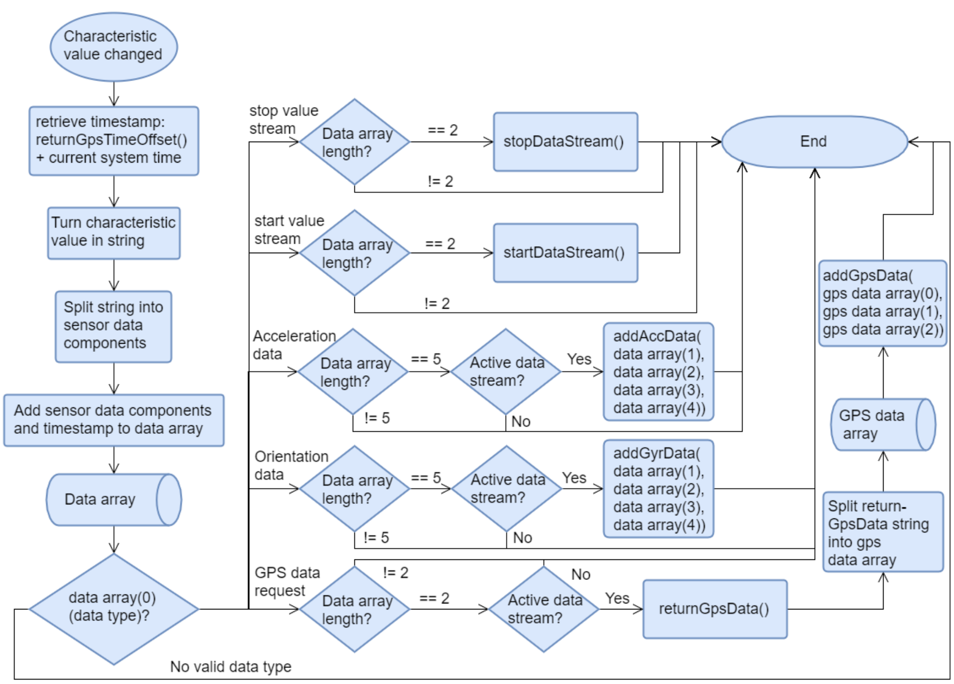


Fig 38. onCharacteristicChanged function flowchart. Note: The sensor data components, apart from the sensor data type that is the first component in the characteristic value string, for gps, acceleration and orientation data can be found in the noSQL database service interface in fig 37.

What can be noted in fig 38 is that the data array length is checked according to a certain fixed number. This number corresponds to the amount of required function parameters + data type constant for the acceleration, orientation data. For the gps data and value stream triggers the amount corresponds to the trigger / request + timestamp.

In order to get an idea about all the non-vital components, all the variables connected to them and the variables connected to the vital parts please take a look at the source code in appendix M.

5.2a Pitfalls

The first problem that was encountered after developing the smartphone application was that sensor data, transferred into data strings, arrived into string parts (An example can be noted in the logcat output in fig 39). The amount of characters in every string part was completely random and caused the characteristic value callback function not to save data into the database (the way sensor data is transferred to the database can be found in appendix M). This problem has partly been fixed by introducing a delay over the serial data transmission to the hm-10 module between consecutive sensor readings with an amount approximating the data connection interval agreed upon between the hm-10 module and the smartphone application. It was impossible to perfectly align the connection interval between the hm-10 module and smartphone application and the serial baud rate between the arduino nano and hm-10 module. Therefore the sensor data still incidentally would arrive in string parts. In the end it was decided to filter out these string parts with a filter mechanism implemented in the characteristic value callback.

```
06-02 15:10:11.489 15018-15030/com.example.bramvberlo.dedicatedsensorsystemtodatabase I/System.out: 0.015,
06-02 15:10:11.489 15018-15030/com.example.bramvberlo.dedicatedsensorsystemtodatabase I/System.out: sw
06-02 15:10:11.525 15018-15145/com.example.bramvberlo.dedicatedsensorsystemtodatabase I/System.out: ek
06-02 15:10:11.525 15018-15145/com.example.bramvberlo.dedicatedsensorsystemtodatabase I/System.out: yolo
06-02 15:10:11.525 15018-15145/com.example.bramvberlo.dedicatedsensorsystemtodatabase I/System.out: 2,
06-02 15:10:11.525 15018-15145/com.example.bramvberlo.dedicatedsensorsystemtodatabase I/System.out: 9.826,
06-02 15:10:11.527 15018-15031/com.example.bramvberlo.dedicatedsensorsystemtodatabase I/System.out: -2.006,
06-02 15:10:11.528 15018-15031/com.example.bramvberlo.dedicatedsensorsystemtodatabase I/System.out: -0.316,
06-02 15:10:11.563 15018-15030/com.example.bramvberlo.dedicatedsensorsystemtodatabase I/System.out: swek
06-02 15:10:11.563 15018-15030/com.example.bramvberlo.dedicatedsensorsystemtodatabase I/System.out: yolo
06-02 15:10:11.563 15018-15030/com.example.bramvberlo.dedicatedsensorsystemtodatabase I/System.out: 3,
06-02 15:10:11.564 15018-15030/com.example.bramvberlo.dedicatedsensorsystemtodatabase I/System.out: -0.0
06-02 15:10:11.564 15018-15030/com.example.bramvberlo.dedicatedsensorsystemtodatabase I/System.out: 80,
06-02 15:10:11.564 15018-15030/com.example.bramvberlo.dedicatedsensorsystemtodatabase I/System.out: -0.038,
06-02 15:10:11.564 15018-15030/com.example.bramvberlo.dedicatedsensorsystemtodatabase I/System.out: 0.017,
```

Fig 39. Data arriving in string parts android studio logcat [20] output example

The second problem that was noted has to do with the connection interval. Instead of letting developers decide their own connection interval android has introduced three fixed connection interval modes in their gatt protocol implementation: low power, balanced and high. The maximum sensor data sample rate that could be realized with a high connection interval mode was 50 Hz. Since the desired data sample rate was 100 Hz this is a clear constraint of this participatory sensing system prototype.

The third problem was encountered after the smartphone application had already received quite a lot of sensor data. As a result the memory allocation on the smartphone would get over flooded with variable references that were already processed by the rest of the smartphone application. This problem did not result in a memory allocation overflood exception. However, it did cause the data sample rate to fluctuate from time to time. A possible fix for this problem was to poke the garbage collector from time to time to clean up all memory allocation to variable references that were already processed by the smartphone application. Since the prototype was tested before this fix was implemented the effectivity of this fix remains unclear.

Last but not least a problem was encountered when trying to save json objects and arrays into documents generated by the local noSQL database. After some time it turned out that couchbase lite, the local noSQL database used in the smartphone application, did not support json objects and json arrays to be saved into database documents. This problem was fixed with a two step system. The first step is to save all sensor data into database documents by using hashmaps. In the second step, when data transmission between the smartphone and main database is initiated, documents would be extracted from the database and turned into a json objects containing json arrays that in turn contained all of the sampled and saved sensor data. These json objects would then be transferred over http to the main database.

5.3 Back-end

The Node.js web server together with connected MongoDB database do not have a front-end. All commands that have to be directly applied to the MongoDB database can be issued by using the so called ‘mongo shell’, a javascript interpreter for running database commands.

The Node.js web server first initiates a connection to a running MongoDB database. In case of the participatory sensing system prototype it was decided to directly download and install a local MongoDB database on a computer and let it run on a port associated to the localhost ip.

Secondly the web server occupies a port associated to a certain ip in order to listen for incoming http requests. Due to the fact that the server was tested outside of the campus this ip was an internal ip given by a router to the computer on which the server was running on. In order to make sure that the web server was accessible from all over the internet the port that the web server was associated to was forwarded so it was accessible from the router’s external ip.

In case of incoming http requests a certain data flow process was initiated. A flowchart explaining this process can be found in fig 40.

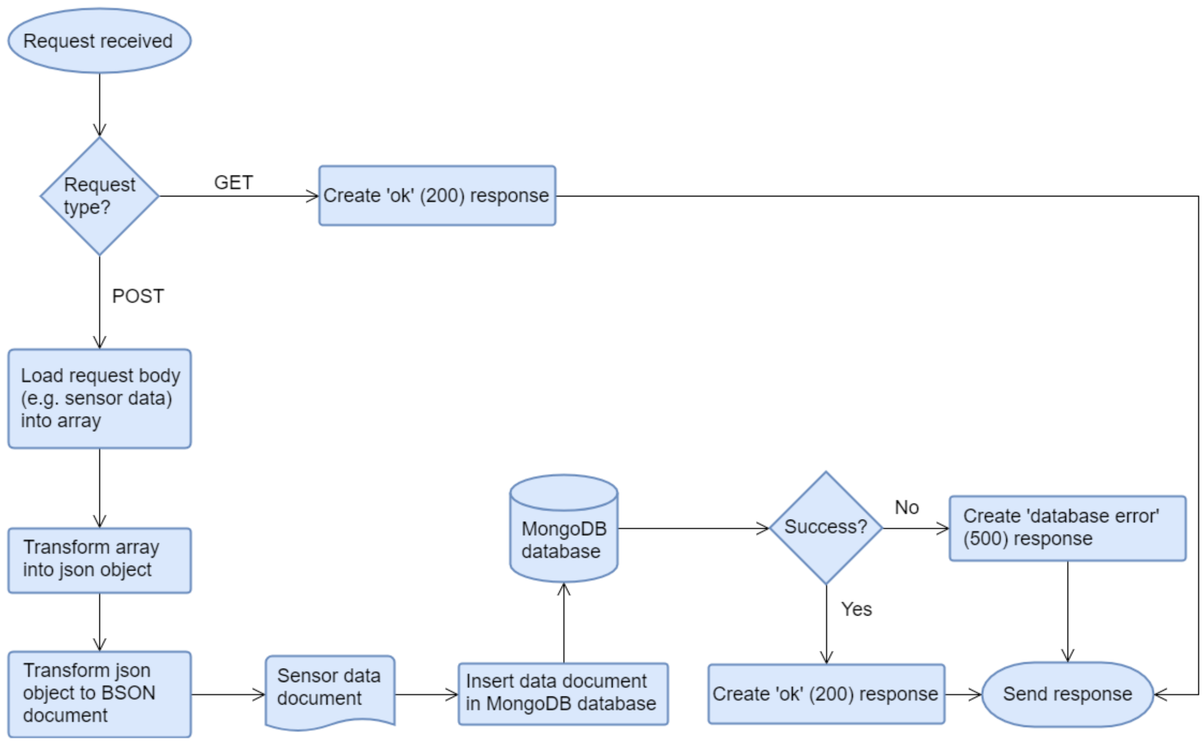


Fig 40. Web server incoming http requests flowchart

What can be noted in fig 40 is that when a GET request is received the Node.js web server immediately responds with an ok (200) response. This was made so that the web server could be tested from a browser if the server was accessible or not.

For a deeper understanding on how the Node.js web server was implemented in terms of code the source code in appendix N can be accessed.

5.3a Pitfalls

The only problem that was encountered during the development of the back-end was that data from the smartphone application arrived in string type and was not ordered on timestamp. In order to fix this a script was developed that could be run in the mongo shell in order to turn all data stored in the database into their respective correct data types and order all the data based on timestamp. This script can be found in appendix N and is called ‘editMongoDB.js’.

5.4 User Interface

The user interface, the main front-end part of the prototype that can be used by users in order to inspect road quality data, can be noted in the UX diagram fig 41.

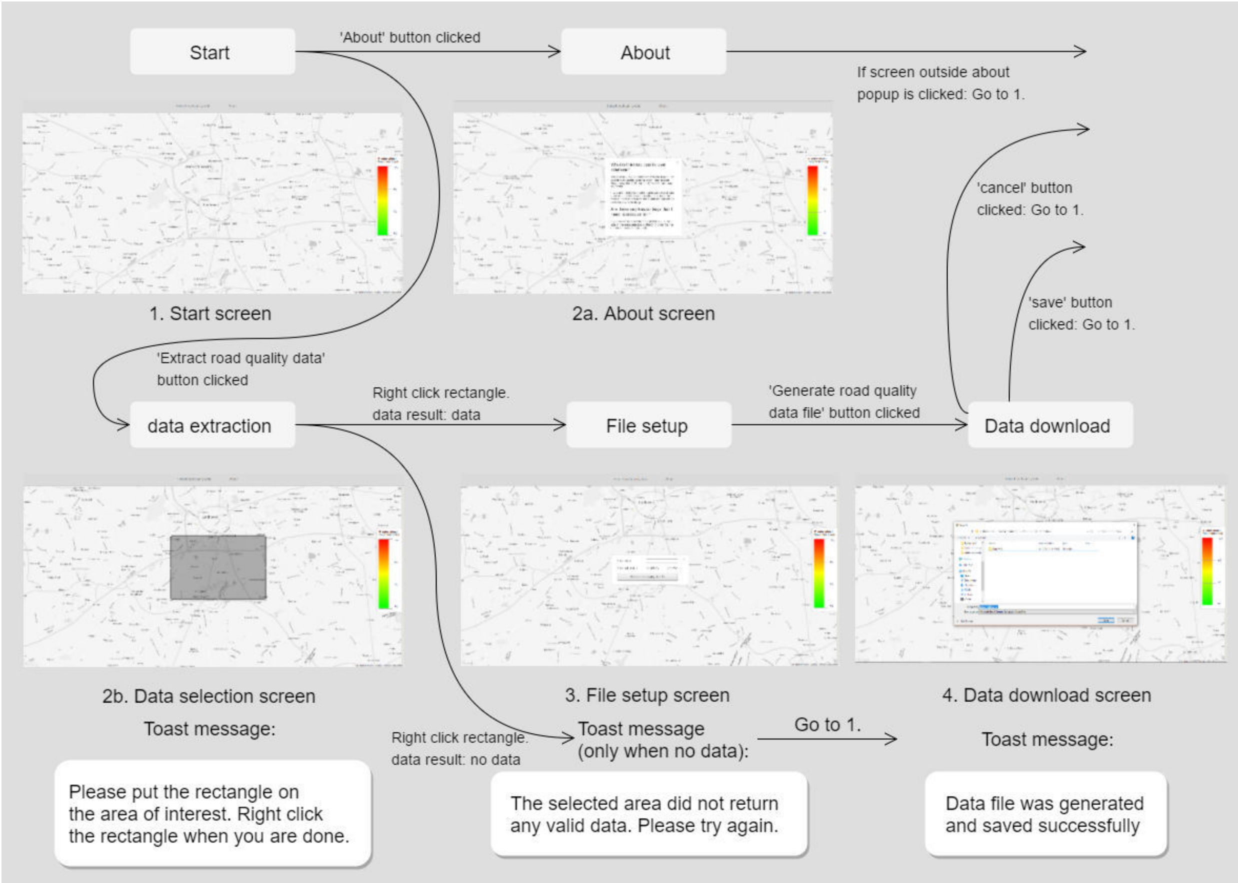


Fig 41. Graphical user interface UX diagram. Note: the live website can be accessed from: http://roadquality.bramvanberlo.com/

Due to the fact that it was not allowed to connect the web server to the iri data generation script at the time the participatory sensing system was created there is no connection between the website and web server where the iri data would have been stored at. The data that is present at the website was manually generated by the administrator of the iri data generation script and afterwards added to the website statically. The iri data plotted on the map is linked to a certain color in order to give a visual impression to the user indicating if certain road segments have either a good, neutral (e.g. not very good but could be much worse) or bad road quality.

Functions 1 and 2 from fig 22 in section 4.4 have not been implemented in the prototype. Even though it is important that these functions are realized when the participatory sensing system will be implemented and used by real users the functions did not have an added value for answering the research questions in section 1.3.

Between the file setup and data download step that can be noted in fig 41 a set of processes are being carried out by the server the website is hosted on in order to generate the data file. First of all the data that falls within the boundaries of the selection area is taken from a static array containing all the iri data plotted on the map. Secondly this data is send via a http request to a php script on the server hosting the user interface that generates a dataset file according to the selected preferences and iri data and echoes back a location where the file can be downloaded. Thirdly at the browser a download procedure (e.g. a callback containing the file location) will be initiated where the user will be asked to pick a location to store the data set file and to either save the file or cancel the download. Fourthly another http request containing the file location is send to a php script that deletes the data set file at the location specified by the incoming http request.

In order to look at the implementation of the user interface in more depth the source code of the complete interface can be accessed at appendix O.

5.4a Pitfalls

The first pitfall that was encountered during the development of the user interface was that, when selecting an area in order to extract data based on that particular area, the data extraction selection area would start to freeze. After some time it was found that this problem was caused due to a wrong width and height definition in the cascaded style sheet of the website where a width and height definition were given in the * section, a section where style can be applied to all website components at once, while the style was only supposed to be applied to the width and height of a div containing the complete map.

The second problem was encountered when testing the part of the user interface where data would start to be extracted based on the selected map area. A user would always have to resize the standard a shape a tiny bit for the data extraction process to work properly. Up until now no fix for this problem has been found. In order to warn users about this bug an explanation in the about section has been added in which the user is told how to extract data in a proper way (e.g. without experiencing this bug).

6. Conclusion

In the end it can be concluded that the integration of a dedicated sensor system into a participatory sensing system network, apart from some minor pitfalls that still have to be omitted, was a success. The state of the art technology that was implemented in order to realize a dedicated sensor system prototype is an arduino nano together with an arduino based gps, bluetooth low energy and acceleration/orientation imu module. In order to realize good quality sensor data that is sufficient to be inserted into an IRI computation script the imu module had to be calibrated every time the system is powered on and a gps speed threshold had to be implemented in order to filter out data that was contaminated due to gps drift.

In order to automate the data acquisition process in the dedicated sensor system the choice was made to sequentially take sensor data samples from the imu and gps module in loop iterations. Due to the fact that transmitting gps data with a suitable precision made the data arriving in string parts pitfall worse the choice was made to generate a gps data request and transmit the request to the smartphone in order to sample gps position and timestamp data from the internal gps module in the smartphone.

In order to automate the data transmission process throughout the complete participatory sensing system prototype two sets of choices were made. The first set of choices deals with automating the data transmission process between the dedicated sensor system and smartphone application through bluetooth low energy. The second set of choices deals with automating the data transmission process between the smartphone application and back-end server with connected MongoDB database through a wifi connection.

In order to automate the data transmission between the dedicated sensor system and smartphone application through bluetooth low energy the choice was made to, next to acquiring data, transmit the data through the bluetooth low energy connection sequentially in loop iterations at the dedicated sensor system side. Afterwards an event-driven data listener based on the notify procedure defined by the bluetooth gatt protocol takes in the data at the smartphone side and performs functions accordingly. These functions could be inserting data into a noSQL data document in a local database or starting/stopping the value stream to the current used noSQL data document in the local database.

In order to automate the data transmission between the smartphone application and back-end server with connected MongoDB database through wifi a function has been implemented that is executed in fixed time intervals. This function first checks if there is an active wifi connection, if there is data present inside the local database and if there is no active data acquisition between the dedicated sensor system and smartphone application going on. If all of these cases are true the function retrieves all of the documents stored in the local database, sends these documents to the back-end server with connected MongoDB database through a http POST request at a given ip and port and afterwards deletes all of the documents from the local database.

The most effective database, based on the latency, throughput and CRUD performance effectivity parameters, in terms of big data handling in the participatory sensing system operating context turned out to be MongoDB. MongoDB is a noSQL document structured database. In terms of CRUD performance MongoDB is extremely efficient and beats every other database that was analyzed. In terms of throughput and latency MongoDB performs extremely well with a low amount of parallel connections to the database but starts to fall off for big amounts of parallel connections to the database. Part of this issue is already tackled due to the fact that the connection between the smartphone application and back-end server with connected MongoDB database is based on the http protocol. This protocol is stateless and therefore does not try to maintain a connection to the back-end server with connected MongoDB database once a request has been sent and a response received.

The state of the art technology that has been implemented in order to visualize IRI numbers consists of a Google maps javascript API implementation together with html, css, jQuery and php. Based on several constructed functionality diagrams two choices were made. The first choice was to plot IRI data in circles that had a specific color depending on the IRI data value. Secondly the choice was made to construct a mechanism that users could use in order to extract IRI data based on a selected area on the map.

In terms of instrumental value provision to multiple participatory sensing system stakeholders it has been determined that value can be provided in terms of four different categories. The first value category is economical value. First of all the participatory sensing system can decrease the amount of money that has to be invested by local and national governmental organizations for road quality data acquisition. Secondly, if road quality improves as a result of participatory sensing system implementation, the likelihood of people and/or insurance companies having to spend money on car damage caused by bad road quality decreases.

The second value category is social value. Due to more efficient road quality data acquisition as a result of a participatory sensing system implementation the overall safety and security for road users increases.

The third value category is scientific value. First of all, a lot of data sets containing road quality data will be generated that can be used by data scientists in order to investigate the heterogeneity of this data and the correlatability of this data to other data sets containing different kinds of information types. Secondly, the participatory sensing system can be used in research about the acquisition of other infrastructure quality data types by for example scientists that are focused on research into pervasive systems.

The last value category is political value. Due to questionable road quality situations in different parts of the world so called ‘user initiatives’ are coming up with the intention to take responsibility of road quality monitoring and maintenance into the hands of the general public. The participatory sensing system can help the initiators of these user initiatives to mobilize people to start monitoring road quality and execute road maintenance where necessary. In the end this will help to improve the effectiveness of these user initiatives.

7. Evaluation

The evaluation section first of all introduces a bachelor project research process evaluation. Secondly the section introduces a critical ethical reflection that introduces ethical challenges that could occur during a participatory sensing system implementation. Lastly the section introduces future work that can be used for further research purposes.

7.1 Process evaluation

The process evaluation sheds a light on what practical components that are part of the bachelor research project went well and what components need to be improved in the next research project. A component that went very well during the bachelor research project is the project planning. During the bachelor research project several other courses had to be finished. For all of the assignments that were part of these courses different parts of the bachelor research project were finished so less time outside these courses had to be spent on the bachelor research project. In the end everything was finished on time.

A component that could be slightly improved in the next research project is the use of financial resources. To give an example too little research was done on the optimal internal dimensions of a universal ABS enclosure for the dedicated sensor system. As a result two additional enclosures had to be bought.

7.2 Ethical reflection

In section 3 it was discussed that the participatory sensing system can bring a lot of value to stakeholders. With this value generation comes a great responsibility to keep the value generation as pleasant as possible for stakeholders while trying to mitigate possible ethical challenges for society as a result of participatory sensing system implementation as much as possible. This section tries to discuss some of these challenges, possible ways to mitigate these challenges and promotion of positive value generation. As a result future developers implementing a participatory sensing system for road quality data acquisition are better prepared when they face some of the scenarios being sketched in this section and can act on the scenarios more effectively.

7.2a Ethical challenges as result of system implementation

The first ethical challenge occurs as a result of the political value described in section 3.3d. People living in a community where road quality is bad and using the participatory sensing system actively in order to take responsibility over deteriorating roads might start to feel more autonomous and believe they can improve the road quality through a joint communal effort. However, the effectivity and safety of solving road quality issues through communal effort rather than professional road maintenance by road maintenance companies issued by local and/or national authorities is questionable. An example is given by the reaction of the Portland Bureau Of Transportation (PBOT) to road maintenance through communal initiatives: “It's generally not safe for folks to be out in the street doing an unauthorized pothole repair” and “that poor repairs can leave the person making the repair liable” [78].

The second ethical challenge is going to occur if the implemented participatory sensing system is badly secured and breached. As a result the complete system is turned into a botnet waiting to be used for large scale DDoS attacks.

This can form a major societal problem due to the fact that the potential scale of the participatory sensing system in terms of connected sensor nodes and the profit for malicious botmasters using the participatory sensing system for unintended use are connected. According to an article written by R. Anderson et al. [79] it can be noted that there is a correlation between the amount of bots acting in a botnet and the amount of revenue a malicious botmaster is able to obtain upon usage of that botnet for DDoS attacks. Therefore breaching the participatory sensing system can be very profitable for people having malicious intentions.

According to a paper written by Kiat Seng Yeo et al. [80] the number of devices connected to the Internet of Things is rising up to a point where 9 Billion devices are connected by 2018. This means that, next to the participatory sensing system, countless other systems with several objects connected to the internet exist that are vulnerable to possible system breaches. As a result it can be argued that malicious botmasters are going to look for systems with the weakest security measures that are extremely easy to breach since the pool of systems is endless. Therefore the likelihood of the participatory sensing network getting breached depends on the security measures implemented as well as the security measures implemented by other systems.

Other people not involved directly in the participatory sensing system, in case of a DDoS attack being carried out by the participatory sensing system as part of a large botnet, are affected due to the fact that services become unavailable for a certain amount of time as a result of DDoS attacks carried out by a botnet facilitated by the breached participatory sensing system. This also affects the services in both financial damage for services and damage to the brand image of a certain service among users.

The third ethical challenge occurs if the generated road quality data displayed in the user interface part of the participatory sensing system is not anonymized up until a certain extent and can be downloaded from the user interface. In this case any external party might be able to use the retrieved road quality data in order to perform dataset linkage procedures among several acquired datasets in order to find information regarding persons, companies, institutions etc. For some users this might not be such an issue due to the fact that the value they are receiving from the system far outweighs the privacy they are giving up. On the other hand for some users data linkage[18] can have severe consequences.

If someone would look at the implementation of the participatory sensing system and the ethical challenge given above on a national perspective, e.g. a Dutch perspective, that person could argue that data linkage does not have to be a problem. In this case data linkage might be useful for tracking down terrorists that hire a car from a car company that has implemented part of the participatory sensing system inside the cars that they have available for hire.

On the other hand data linkage as a result of participatory sensing system implementation creates a moral contradiction when the participatory sensing system is implemented on a global scale. While the national perspective given above creates a situation in which terrorists might be captured faster due to data linkage another national perspective with questionable moral situations might be wrongfully supported. An example of such a questionable moral situation is currently occurring in the Philippines where president Rodrigo Duterte recently launched a 'Philippine drug war' [82]. According to Human Rights watch, in relation to the Philippine drug war, "the Philippine police is responsible for extrajudicial executions—the deliberate killing by state security forces or their agents of a person in custody. A clear modus operandi of police operations emerged …with visits that often proved not so much to be warnings but a method of confirming the identity and whereabouts of a target." [82]. In case of this example the police might be able to use data linkage in order to track down targets more easily.

7.2b Risk mitigation and value promotion

In order to mitigate the risk of negative ethical consequences as a result of the participatory sensing system implementation and to encourage situations that create more emphasis on the positive aspects that result from the implementation of the participatory sensing system a set of measures and design choices can be taken. These measures and design choices are discussed more in depth in the next paragraphs.

Even though the effectivity and safety of solving road quality issues through communal effort is questionable the participatory sensing system has the ability to turn a questionable situation into a situation in which both the community and authorities might benefit from each others help. This situation can be created through promoting the usage of the participatory sensing system in a more indirect way. This indirect way is best described by promoting the participatory sensing system to be used inside a car in order to gather road anomaly data and giving the authorities the opportunity to use this data in order to schedule and execute road maintenance more efficiently.

[18] Data linkage: a process which temporarily brings together two or more data sets from different organisations for data correlation purposes in order to produce a wealth of information which can be used for research and statistical purposes [81].

The promotion described in the paragraph above can be enforced by the implementation of soft paternalism[19] style measures. These measures could mean that the participatory sensing system user interface is going to restrict road anomaly status updates by giving this option only to users with a special account that are logged into the user interface. The reason why this measure is a soft paternalism style measure is that, rather than prohibiting means for solving road quality issues through communal effort, the implementation encourages other types of communal help that contribute to cooperative measures for both citizens and authorities in terms of solving road quality issues.

Mitigating botnet attacks (e.g. botnet protection) is going to be hard. Matta et al. state that "recent emergence of attacks performed at the application layer has multiplied the number of possibilities that a botnet can exploit to conceal its malicious activities" [84]. Pijpker and Vranken state that "ISPs cannot implement advanced botnet detection measures mainly due to privacy concerns of customer data" [85]. As a result the following question was raised: should the participatory sensing system be secured as well as possible, or is it enough to make it a harder target than other big data apps using the internet?

In a scenario where the participatory sensing system implementation should be secured as well as possible there seems to be only one viable option. This option consists of hiring an internet security company like for example Fox-IT [86] in order to monitor in and outgoing http traffic day and night. Monitoring can for example be done by setting up packet sniffers between connections across all system components of the participatory sensing system (e.g. between mobile phone with connected sensor node through Bluetooth low energy to the main database and from user interfaces to the main database).

However, the big downside to this approach is that hiring an internet security company will drain a lot of resources. This will happen due to the fact that an internet security company has to hire a lot of specialized personnel to monitor traffic day and night and therefore will ask pretty high subscription prices.

In a scenario where the participatory sensing system implementation should make itself a harder target compared to other big data applications using the internet botnet security can be done much faster and more efficient. The reason for this is that, according to an initiative note written by D66 Dutch political party member Kees Verhoeven on the Internet of Things [87], Europe has not yet established a cyber security standard for internet of things devices. Therefore it can be assumed that a lot of internet of things devices have not yet incorporated a cyber security standard on their own. Kees Verhoeven suggests the following standard cyber security measures: end-to-end data encryption, password implementation when people try to access the main database, updating known data security leaks and educating users on how to securely participate in the participatory sensing system [87].

---

[19] Soft paternalism: the view that the only conditions under which paternalism is justified is when it is necessary to determine whether the person being interfered with is acting voluntarily and knowledgeably [83].

Based on the two scenarios sketched in the two paragraphs above it can be assumed that the second scenario works best. The first reason for this is that when the participatory sensing system is first implemented there are no resources available in order to hire an internet security company to monitor internet traffic. The second reason is that being a harder target to breach compared to other big data applications as a result of standard cyber security implementations seems secure enough. This is due to the fact that other applications have not yet implemented standard cyber security measures and therefore will be a far easier target to breach than the participatory sensing system.
In order to prevent contradictory moral situations where the implementation of the participatory sensing system leads to positive effects in one perspective and to moral questionable situations in another perspective it is best, from a utilitarian[20] perspective, to eradicate the possibility of participatory sensing system data being used in data linkage operations completely.

In order to facilitate this the choice was made to limit the amount of privacy sensitive data used in the participatory sensing system as much as possible. The only privacy sensitive data type that is necessary for the system to function is GPS data. This data is then correlated with non-privacy sensitive data such as acceleration and orientation data by means of a timestamp so it can be used by several system components. In case of the data that is presented in the user interface a gps coordinate is correlated to an IRI number. It is believed that the data presented in the forms above is not interesting enough for people to use in data linkage operations in order to determine personal information.

However, if the data in the future does proof to be useful for people to use in data linkage operations measurements have to be taken. One of these measurements could mean incorporating so called ‘data anonymization algorithms’. A limitation of this approach is that anonymization algorithms could render vital parts of data for road quality analysis useless. As a result data anonymization can only be applied to data that is not vital for road quality analysis to work. An example of such a data anonymization algorithm is the SHA-2 (Secure Hash Algorithm 2) hash algorithm [89].

7.2c Design implementations

In order to conclude the ethical reflection an explanation is given on how the suggestions for risk mitigation and positive effect promotion have / have not been implemented and why / why not.

The limitation of privacy sensitive data gathering in relation to the moral contradiction based on data linkage in a global perspective has been implemented in the participatory sensing system prototype and design. The reason for this is that this limitation made it easier to develop the participatory sensing system prototype in general.

[20] Utilitarianism: held to be the view that the morally right action is the action that produces the most good for society in general [88].

The limitation of pothole status updates to users being logged into an account and standard cyber security measures have not been implemented. The reason for this is that the software development work that would have to be put in in order to create the functionality far outweighed the potential impact it would make during for example a prototype demo portraying the core functionality of the participatory sensing system design.

7.3 Future work

In the conclusion it can be noted that the integration of a dedicated sensor system into a participatory sensing system network was a success apart from some minor pitfalls. In order for the system to be implemented in a test pilot in the future, some minor pitfalls have to be fixed. The first pitfall is that the dedicated sensor system was still too big for some car dashboards and has to be made even smaller. The second pitfall is that the bluetooth low energy module used in the dedicated sensor system started to act weird after using it for some time. In order to solve this another bluetooth low energy module might have to be installed instead of using the dirty fix that was implemented in this bachelor research project. The third pitfall is that still a small percentage of data arrives in string parts to the smartphone application. The fourth pitfall was the memory allocation problem. The fix for this problem still has to be tested.

Additionally to the pitfall fixes in the paragraph above the gps module in the dedicated sensor system will suffer to keep a connection to active satellites when driving under a bridge or through a tunnel. Therefore during the bachelor research project bridges and tunnels were avoided. In order to solve this problem research has to be done into combining the gps data together with acceleration and orientation data in order to implement for example a kalman filter for sensor data fusion purposes.

Lastly the document oriented structure of the database used in the participatory sensing system prototype offers functionality in terms of being able to dynamically add and/or subtract variable types from documents being inserted and/or updated in the database without having to change a complete database layout structure.

As a result future research could be executed with the already existing prototype into the possibilities of data acquisition of other data types that can be correlated with the road quality data generated during this research project. In the end this could lead to for example the development of a BIS (Business Information System) for different business sectors such as for example the automotive sector. An example of such a BIS is given in fig 42.

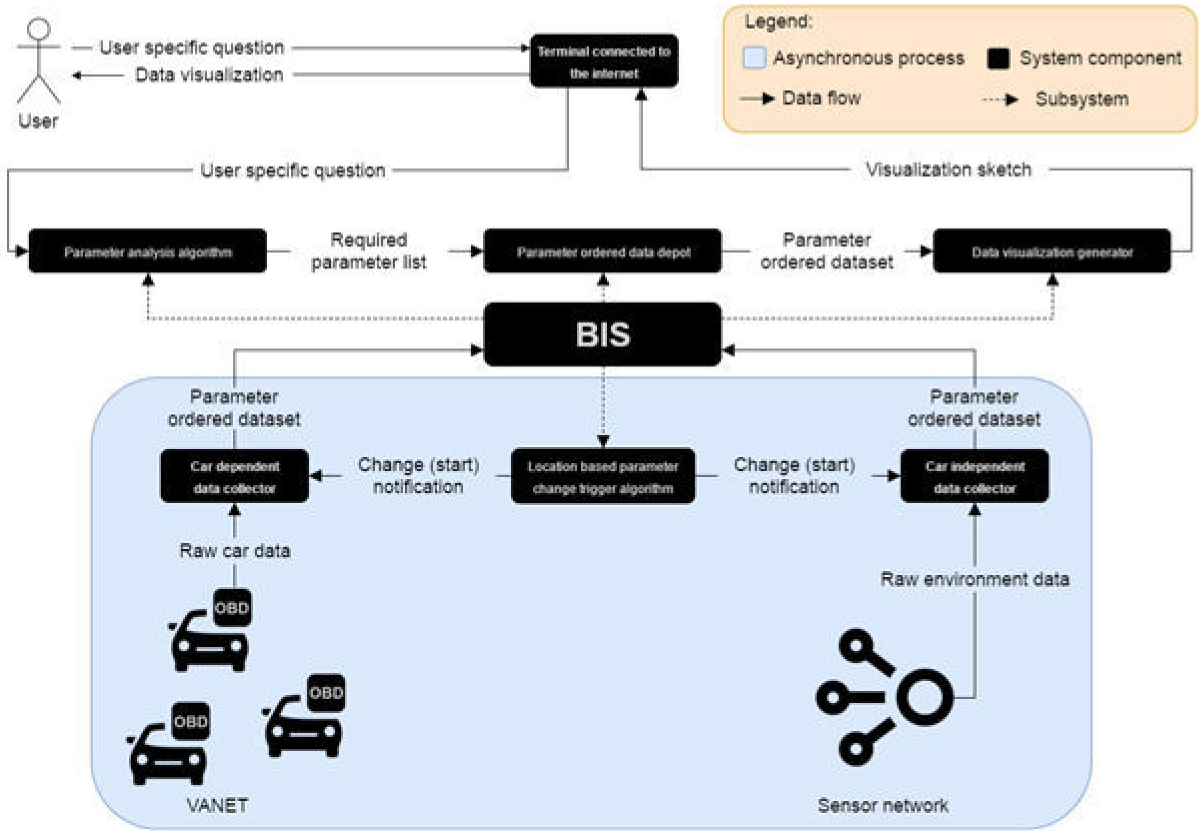


Fig 42. Example of business information system for the automotive sector. Note: icons originate from [21].

Fig 42 defines a business information system that, through the use of predictive analytics and big data, is able to generate useful feedback information for multiple purposes. One purpose is for automotive brands and users using products related to those brands to indirectly provide feedback about their product to the automotive brands without using so called “user surveys”. Another purpose is for local and national governments in order to keep companies in line of the established environmental agreements. Last but not least the information could be used by journalists investigating the Sjoemelsoftware[21] affair to obtain an objective basis for their research.

[21] Sjoemelsoftware: the Dutch word of the year issued in 2015 that describes software used by Volkswagen in order to give their cars a better appearance in terms of car performance. A recent report by The Greens | European Free Alliance [90] shows that, next to Volkswagen, more car companies are being questioned in their involvement of using so called Sjoemelsoftware.

Appendix

A - Stakeholder interview scheme 1

1. What is the current situation regarding road quality of local roads in Portland according to the Portland Anarchist Road Care (PARC) group?

2. What type of solutions did the Portland Bureau of Transportation already propose in regard to the bad road quality in Portland?

3. To what extent are these solutions successful or not and why?

4. Which problems regarding road quality of local roads in Portland does the PARC group try to solve? (e.g. what is the mission of the PARC group?)

For the next set of questions the following context is provided:

Last year several research groups around the globe have conducted experiments with the use of sensors contained in a mobile phone. The purpose of these experiments was to determine to what extend these sensors are suitable for predicting road anomalies and determining a road quality index number for different road segments. The applications that were built were able to predict road anomalies with a success rate of around 85-92% and could compute a road quality index numbers based on the sensor outputs. (For more in depth technical information see: http://eprints.eemcs.utwente.nl/26650/01/Seraj_et_al._-_RoADS.pdf)

However, two big limitations were noted. The first one was that people tend to move their mobile phone in their car while driving. As a result the sensor data obtained is contaminated and unsuitable for the intended purpose. The second one was that the applications were not able to share information with each other and provide an information overview based on all sensor applications publicly on the internet.

I am currently working on a participatory sensor network application comprised of a sensor system that can be put into the cigarette charger of a car and, through the use of a mobile application, will communicate sensor data to a central server. A web application will be connected to this server that will be publicly available. This web application will show the road quality index numbers and possible road anomalies of road segments on a google map.

The goal of this participatory sensor network application is to provide road quality information to different stakeholders that is obtained through a joint effort of different people. The road quality information could be used for looking up road anomalies or having an up to date overview of road quality of a city such as Portland.

5. What is the opinion of the PARC group on implementing a participatory sensor network application as described above in order to discover road anomalies and maintain an overall road quality indication?

6. To what extent does the PARC group think the participatory sensor network application described above is able to provide support to the PARC group's mission?

7. How can the web application aspect of the participatory sensor network application described above be improved in order to provide better support to the PARC group's mission?

8. Any comments, questions, requests etc regarding the participatory sensor network application that come up can be posted here

9. In case I have more questions what is the best way to keep in touch?

B - Stakeholder interview scheme 2

1. What is the current situation regarding road quality of local roads in Portland according to the Portland Bureau of Transportation (PBOT)? (in case of good road quality go to question 4)

2. What type of solutions did the PBOT already propose in regard to the bad road quality in Portland?

3. To what extent are these solutions successful or not and why?

4. How many percent of the annual PBOT budget is spent in order to maintain a reasonable level of road quality? (if this data is not publicly available is there other statistical information publicly available that sheds a light on the volume of budget required in order to maintain a reasonable level of road quality?)

For the next set of questions the following context is provided:

Last year several research groups around the globe have conducted experiments with the use of sensors contained in a mobile phone. The purpose of these experiments was to determine to what extend these sensors are suitable for predicting road anomalies and determining a road quality index number for different road segments. The applications that were built were able to predict road anomalies with a success rate of around 85-92% and could compute a road quality index numbers based on the sensor outputs. (For more in depth technical information see: http://eprints.eemcs.utwente.nl/26650/01/Seraj_et_al._-_RoADS.pdf)

However, two big limitations were noted. The first one was that people tend to move their mobile phone in their car while driving. As a result the sensor data obtained is contaminated and unsuitable for the intended purpose. The second one was that the applications were not able to share information with each other and provide an information overview based on all sensor applications publicly on the internet.

I am currently working on a participatory sensor network application comprised of a sensor system that can be put into the cigarette charger of a car and, through the use of a mobile application, will communicate sensor data to a central server. A web application will be connected to this server that will be publicly available. This web application will show the road quality index numbers and possible road anomalies of road segments on a google map.

The goal of this participatory sensor network application is to provide road quality information to different stakeholders that is obtained through a joint effort of different people. The road quality information could be used for looking up road anomalies or having an up to date overview of road quality of a city such as Portland.

5. What is the opinion of the PBOT on implementing a participatory sensor network application as described above in order to discover road anomalies and maintain an overall road quality indication?

6. Any comments, questions, requests etc regarding the participatory sensor network application that come up can be posted here

7. In case I have more questions what is the best way to keep in touch?

C - Stakeholder interview scheme 3

1. How many research groups around the globe are conducting research in use of participatory sensing systems and smartphone sensor systems in order to detect road anomalies and IRI computation?

2. What are the main reported downsides to smartphone sensor systems and participatory sensing systems currently used in research by research groups around the globe and why?

3. How does the PS group envision use of a participatory sensing system prototype to address societal issues?

4. How does the PS group envision use of a participatory sensing system prototype in both current and further research?

5. Are there dedicated system, smartphone app, database and UI requirements that might positively influence both current and further PS group research?

D - Stakeholder interview scheme 1 results

1. What is the current situation regarding road quality of local roads in Portland according to the Portland Anarchist Road Care (PARC) group?

The roads are in a dangerous state of disrepair. The amount and severity of potholes is causing damage to vehicles, an increase in automobile collisions, and a hazard to cyclists and pedestrians. The state has refused to look for any solutions to the problem outside of hot asphalt, and has insisted that the people of portland simply tolerate the conditions, during Portland's wet, rainy winter.

2. What type of solutions did the Portland Bureau of Transportation already propose in regard to the bad road quality in Portland?

The department of transportation only uses hot asphalt to patch roads, and has simply asked people to tolerate the deteriorating road conditions, until weather improves. Meanwhile the potholes get worse and worse, and in many places, will end up requiring fully repaving sections of road, because 15 foot long channels have formed due to the ongoing lack of repair.

3. To what extent are these solutions successful or not and why?

This solution, the only one that the state seems to pursue, has been an abysmal failure. First and foremost because they are not able to implement it in a timely manner. And now that we have had some sunny days, and they have patched some potholes, for whatever reason, many of their patches are already deteriorating. Potholes have reopened on a number of main thoroughfares that they focussed for their first several sunny days.

4. What problems regarding road quality of local roads in Portland does the PARC group try to solve? (e.g. what is the mission of the PARC group?)

Our primary focus is to mobilize people to reclaim public spaces for the community. Part of this means taking responsibility for the conditions of the roads. We do not have the numbers or funding that the state has, but we mobilize people to cold patch potholes, as well as to make them more visible, with paint, in order to make them easier to avoid, especially for cyclists, especially at night.

5. What is the opinion of the PARC group on implementing a participatory sensor network application as described above in order to discover road anomalies and maintain an overall road quality indication?

Interestingly enough, we have also been developing an app, nothing high tech like yours, but an app for crowdsourcing information about potholes, that is shared between everyone with the app, you simply mark the pothole on a map, with its severity, etc.

Having an automated sensor based app sounds like a powerful use of technology, but we do have concerns based on the work we do. The data that is collected, to any central database is of particular concern. Any sort of identifying information collected or permissions to access features of the phone are troublesome. Especially here in the US, where our federal government just passed a law allowing all of that information to be sold.

6. To what extent does the PARC group think the participatory sensor network application described above is able to provide support to the PARC group's mission?

Like we said, we have developed an app to serve a similar purpose, but without the high tech input. Having the information is very helpful. It is a question of how the information is collected and what kind of sacrifices are made to get it.

7. How can the web application aspect of the participatory sensor network application described above be improved in order to provide better support to the PARC group's mission?

Like we said, focussing on privacy. Like only collecting the sensor and location data, and not any identifying information from the phone. Also, the ability for people to go in and mark work they have done at any location. Mark a pothole fixed. Etc.

8. Any comments, questions, requests etc regarding the participatory sensor network application that come up can be posted here

Not really. Thanks for getting in touch. Sorry it has taken so long to respond to you.

9. In case I have more questions what is the best way to keep in touch?

This email is best. Thank you.

E - Stakeholder interview scheme 2 results

1. What is the current situation regarding road quality of local roads in Portland according to the Portland Bureau of Transportation (PBOT)? (in case of good road quality go to question 4)

The majority of our Arterial and Collector streets are in Fair to Poor condition
The majority of our Local streets are in Poor to Very Poor condition

Condition definitions are available in our 2015 Status and Conditions Report (p 74-75) available online here: https://www.portlandoregon.gov/transportation/62871

2. What type of solutions did the PBOT already propose in regard to the bad road quality in Portland?

Our approach is for preventative maintenance rather than 'worst first'. We try to reinvest in our assets before they degrade to the point where they are more costly to treat. We also prioritize our collectors and arterials over local streets

3. To what extent are these solutions successful or not and why?

Our pavement prioritization method uses data to track degradation over time (we use Street Saver to track condition and to develop preliminary paving lists). We feel confident in our approach. Unfortunately our resources limit our ability to address early treatment needs citywide.

4. How many percent of the annual PBOT budget is spent in order to maintain a reasonable level of road quality? (if this data is not publicly available is there other statistical information publicly available that sheds a light on the volume of budget required in order to maintain a reasonable level of road quality?)

~5% of our budget is spent on maintaining our pavement system. Please note that PBOT's total budget used in this calculation includes our Tram, Streetcar, Sewer Repair, and Parking programs.

5. What is the opinion of the PBOT on implementing a participatory sensor network application as described above in order to discover road anomalies and maintain an overall road quality indication?

We are currently investigating the value of crowdsourced data on potholes, road condition from Waze

For automatically-sensed data, we had talked about the introduction of ‘smart’ vehicles or autonomous vehicles in the future.

6. Any comments, questions, requests etc regarding the participatory sensor network application that come up can be posted here

7. In case I have more questions what is the best way to keep in touch?

You can e-mail me

F - Stakeholder interview scheme 3 results

1. How many research groups around the globe are conducting research in use of participatory sensing systems and smartphone sensor systems in order to detect road anomalies and IRI computation?

Giving a number is quite hard but there are quite a lot of research groups doing research with smartphone sensor applications for road quality data acquisition. In terms of monitoring the data acquired from those systems little to no research has been done yet. In itself participatory sensing is nothing new. However, PS thinks that research into a participatory sensing for road quality data acquisition is quite new.

2. What are the main reported downsides to smartphone sensor systems and participatory sensing systems currently used in research by research groups around the globe and why?

Firstly the existence of several dynamical system variables (in essence making the sensor network a dynamical network) such as changing road and road anomaly topologies, changing system coverage due to mobility aspects of the system and changing routing. Secondly privacy aspects involved in data acquisition when creating smartphone sensor systems and participatory sensing systems. Thirdly missing know-how on how to motivate people to participate in a smartphone sensor systems or participatory sensing system either implicit or explicit.

3. How does the PS group envision use of a participatory sensing system prototype to address societal issues?

Firstly we hope to improve the overall safety and security of roads. Secondly we hope to make an economical impact for local and national governments through improving maintenance planning. Thirdly we hope that the system is able to decrease the likelihood of people having to spend money on car damage caused by bad road quality.

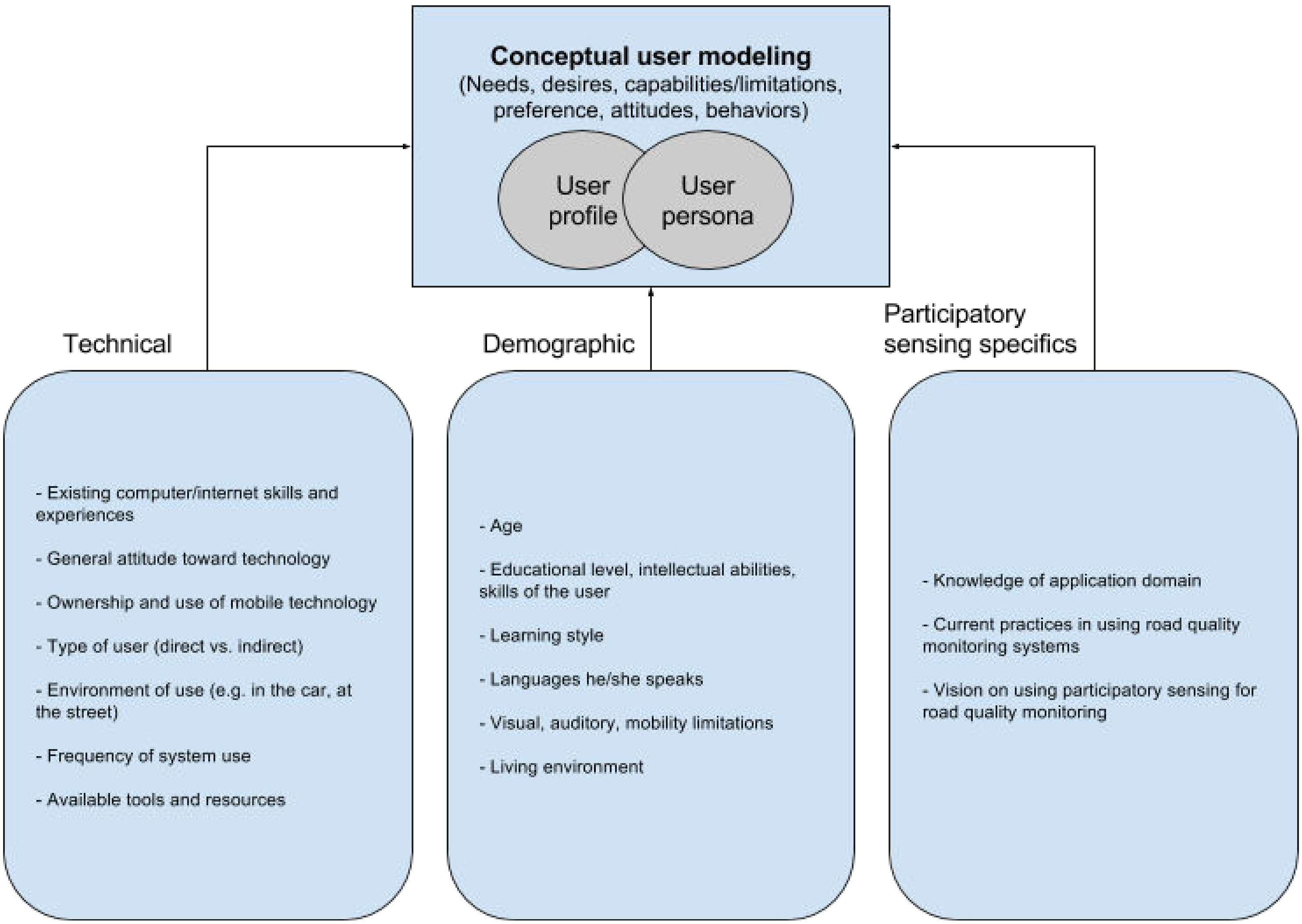
4. How does the PS group envision use of a participatory sensing system prototype in both current and further research?

Firstly the system could be used in research about governmental issues such as road safety and road quality monitoring. Secondly the system could be used in research about obtaining a better insight about infrastructure quality other than just road quality (for example determining defect train track segments based on a quality indication). Thirdly the system could be used in big data (data science) research about heterogeneity of data (to what extent can data from this system be correlated with data from other systems in order to obtain new insights?)

5. Are there dedicated system, smartphone app, database and UI requirements that might positively influence both current and further PS group research?

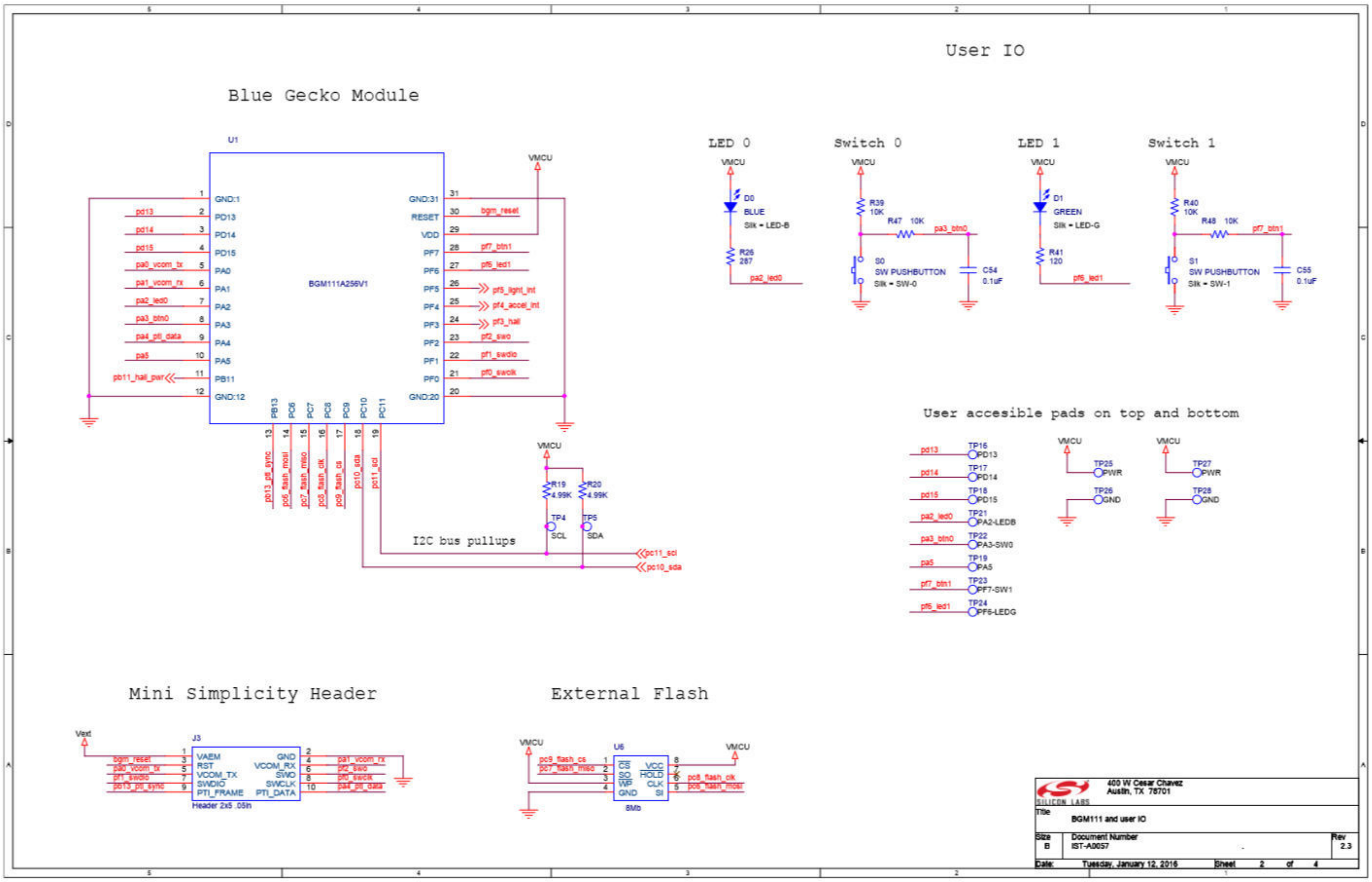
Firstly energy efficiency (making sure the system does not require too much power consumption). Secondly seamless system integration (if no connection can be made between system components data has to be saved locally in a good manner or connection has to be made through other transmission types).

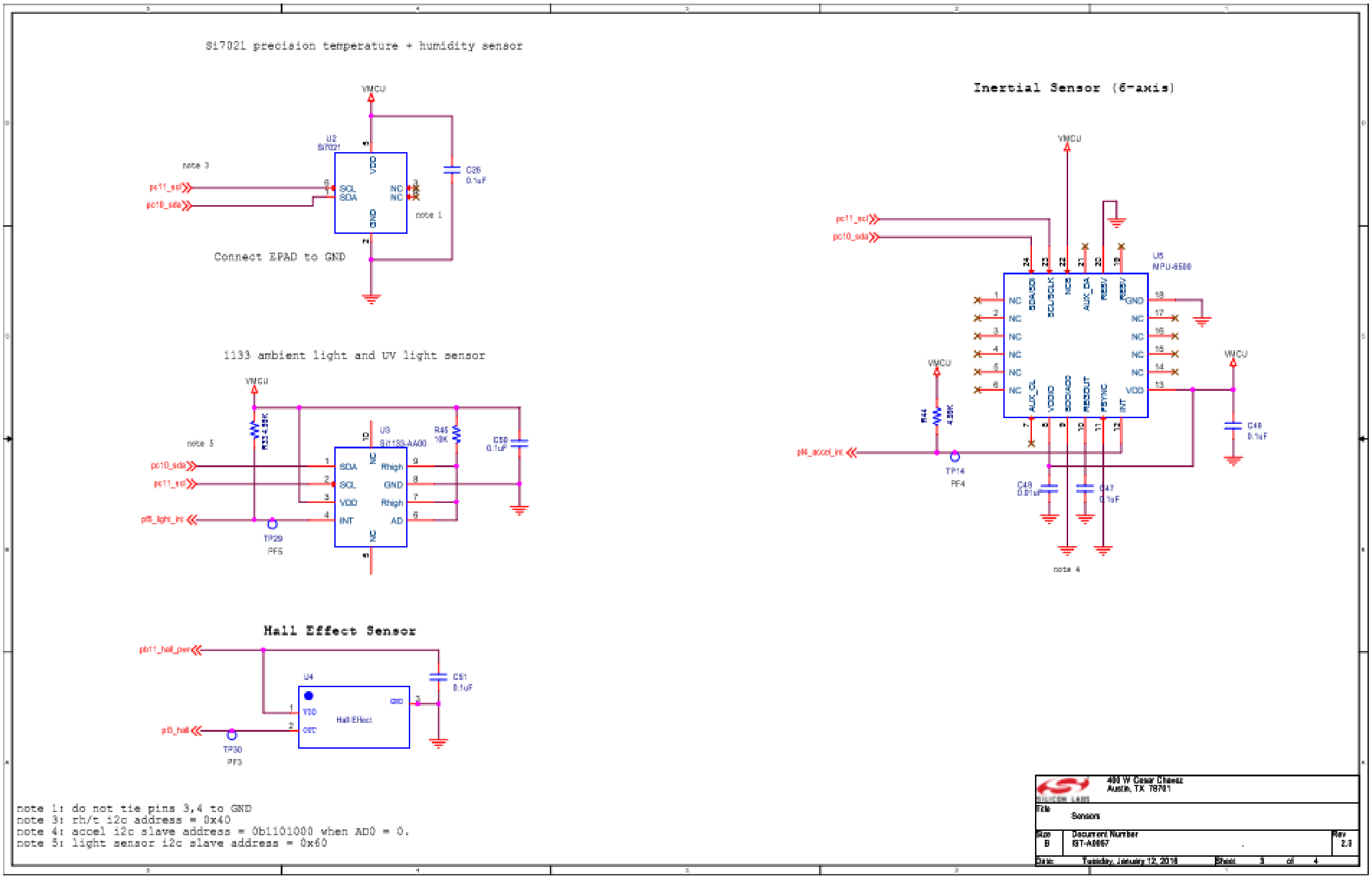
G - Persona framework



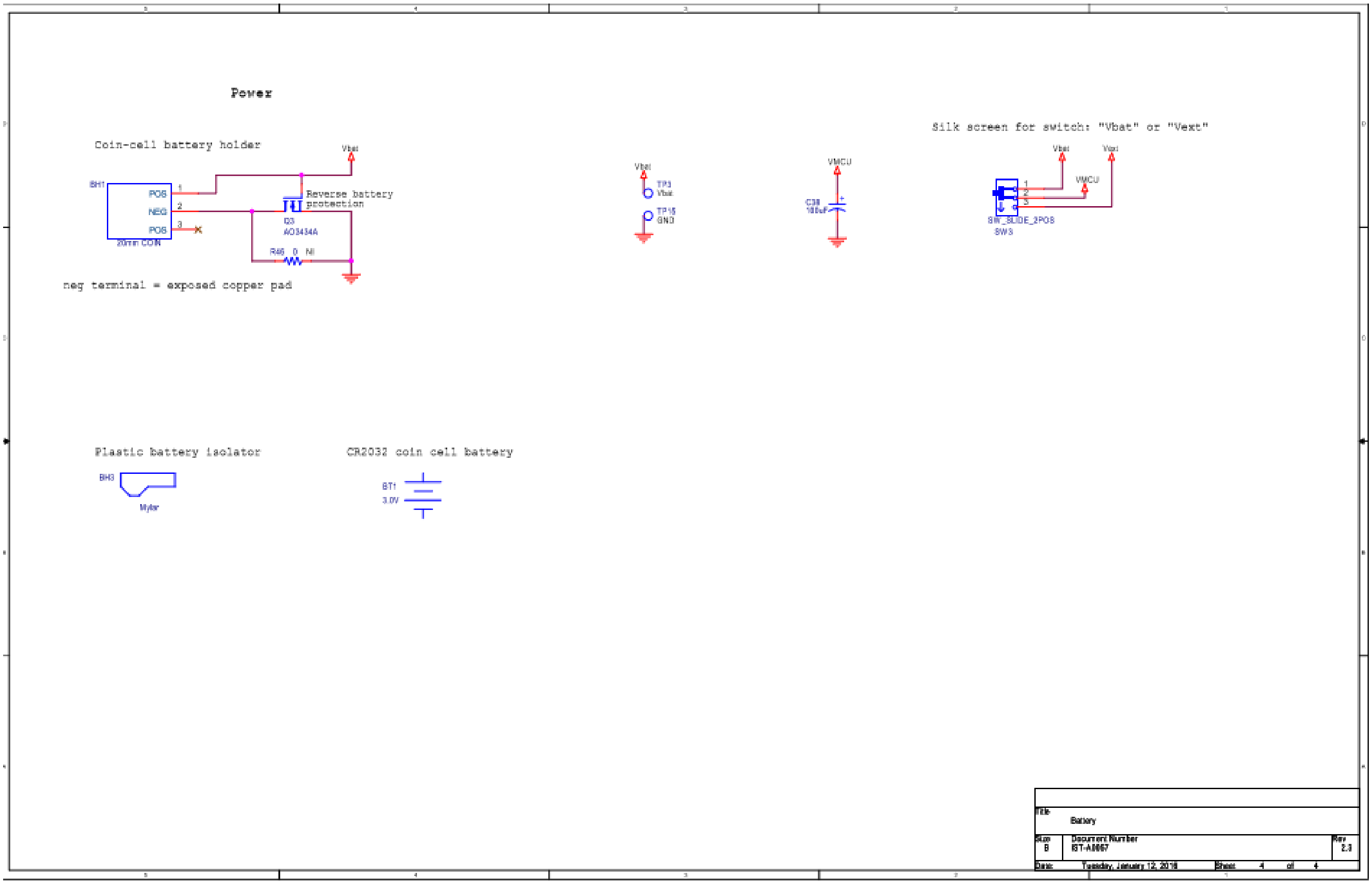
The following framework is an alteration on the framework presented in [91].

H - Thunderboard react schematics [92]





note 1: do not tie pins 3,4 to GND
note 3: rh/t i2c address = 0x40
note 4: accel i2c slave address = 0b1101000 when AD0 = 0.
note 5: light sensor i2c slave address = 0x60


Power
Coin-cell battery holder
BH1
POS
NEG
POS
20mm COIN
Vbat
Reverse battery protection
Q3
AO3434A
R46 0 NI
neg terminal = exposed copper pad
Vbat
TP3
Vbat
TP15
GND
VMCU
C38
100uF
Silk screen for switch: "Vbat" or "Vext"
Vbat
Vext
VMCU
SW_SLIDE_2POS
SW3
Plastic battery isolator
BH3
Mylar
CR2032 coin cell battery
BT1
3.0V
Title
Battery
Size
B
Document Number
IST-A0067
Rev
2.3
Date:
Tuesday, January 12, 2016
Sheet 4 of 4

I - Database handling effectiveness latency graphical results

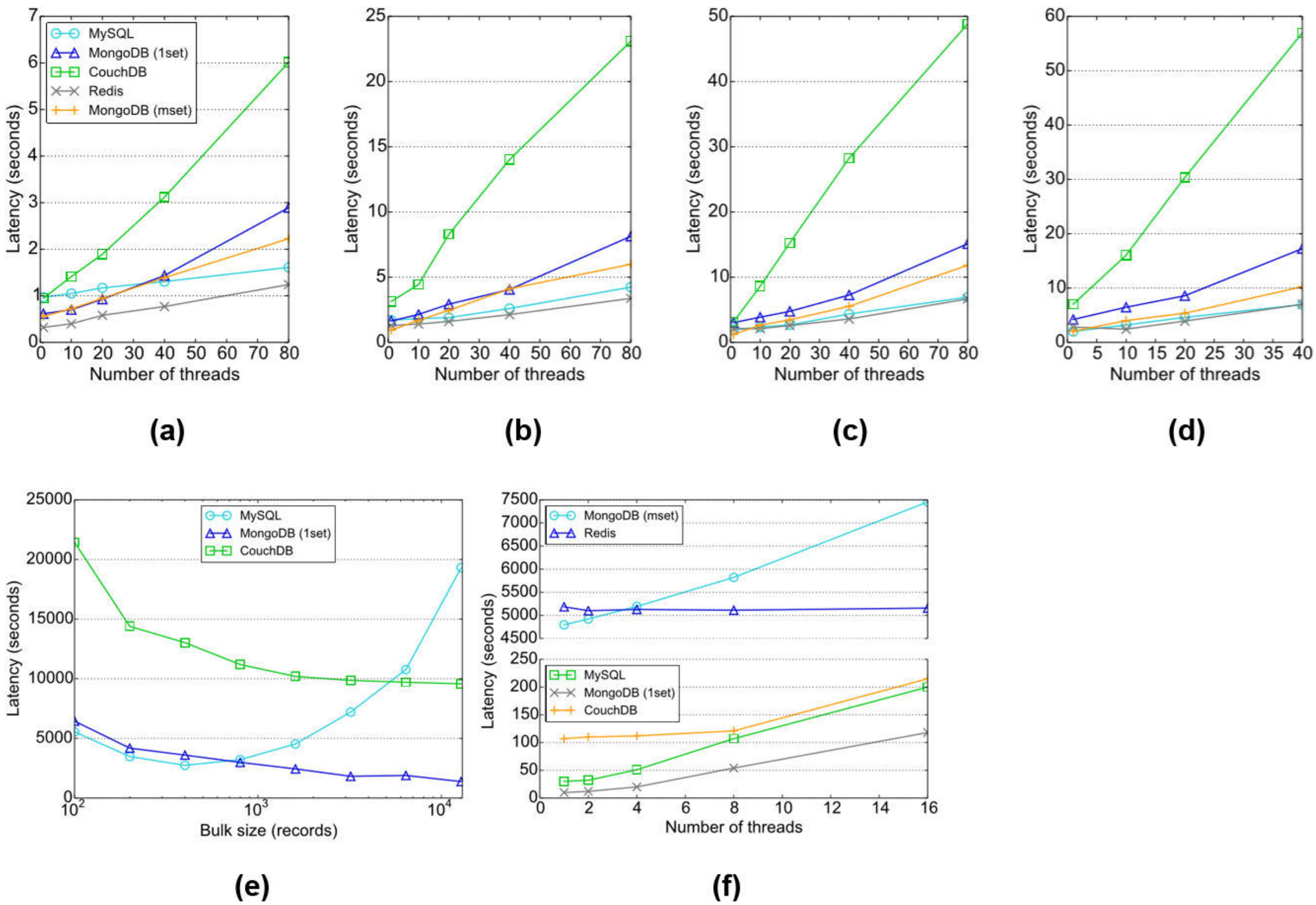


Fig 43. Read latency results in multi-threaded environments for different number of records: (a) 1M (b) 10M (c) 20M (d) 40M. (e) Latency when bulk insert is performed (f) multi-thread write latency. [22]

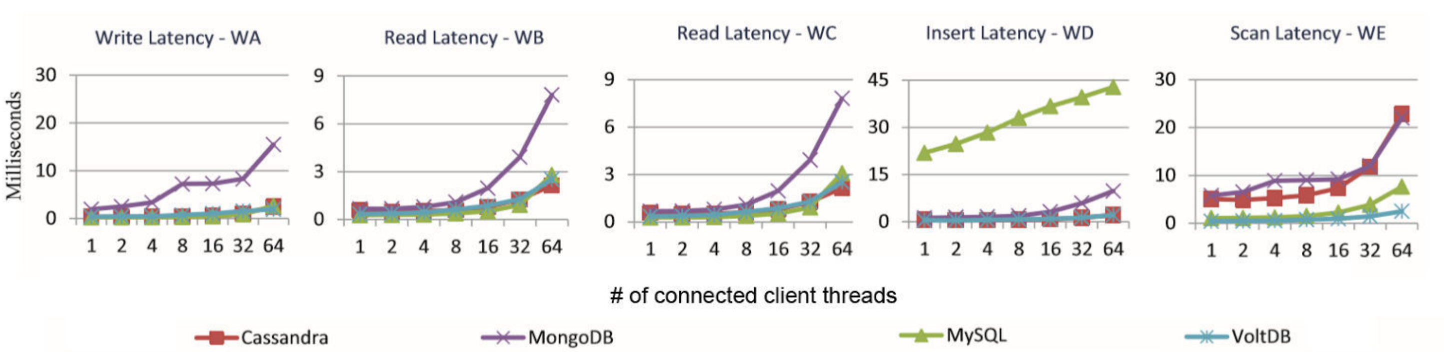


Fig 44. Latency results for different YCSB workloads: A, B, C, D and E [23]

J - Database handling effectiveness throughput graphical results

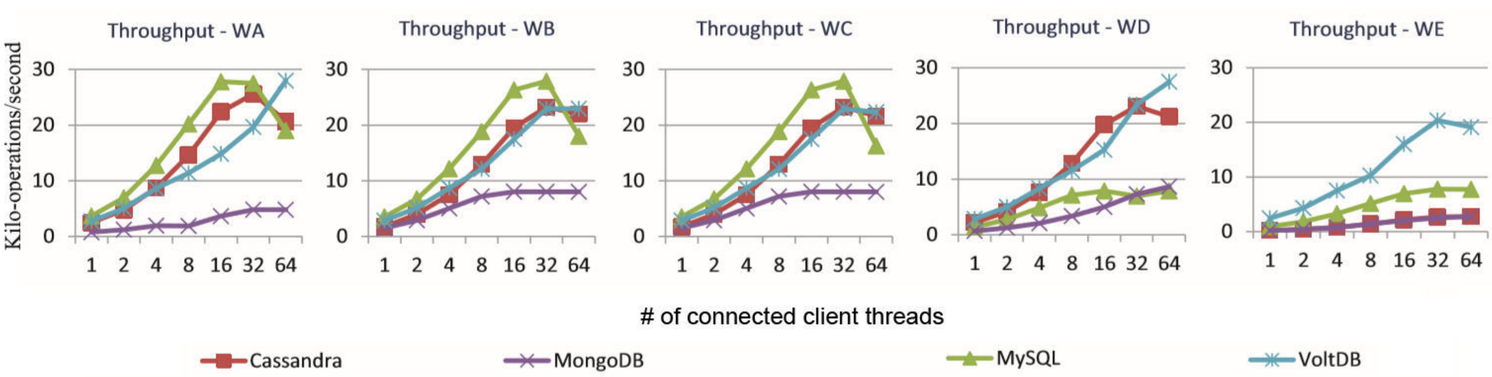


Fig 45. Throughput results for different YCSB workloads: A, B, C, D and E [23]

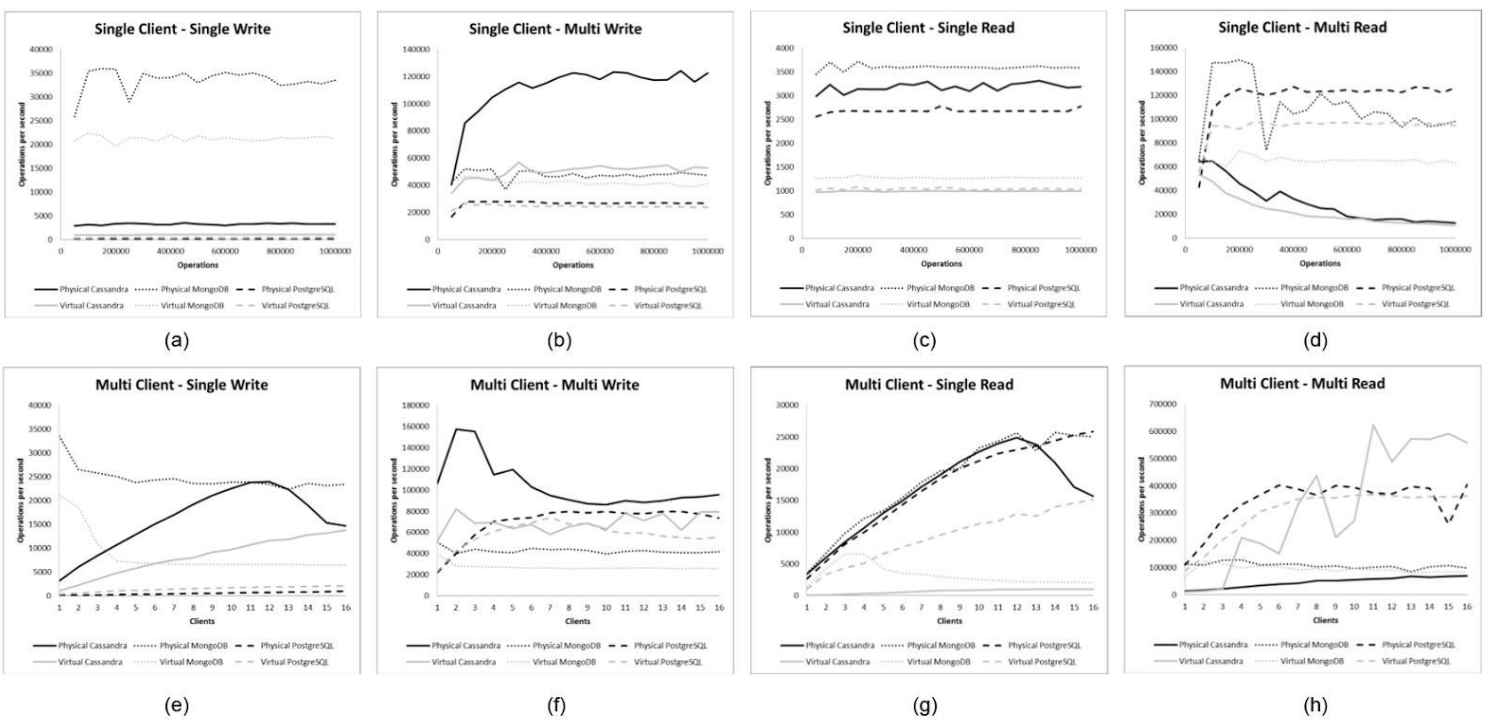


Fig 46. Operations per second for different client and operation situations [24]

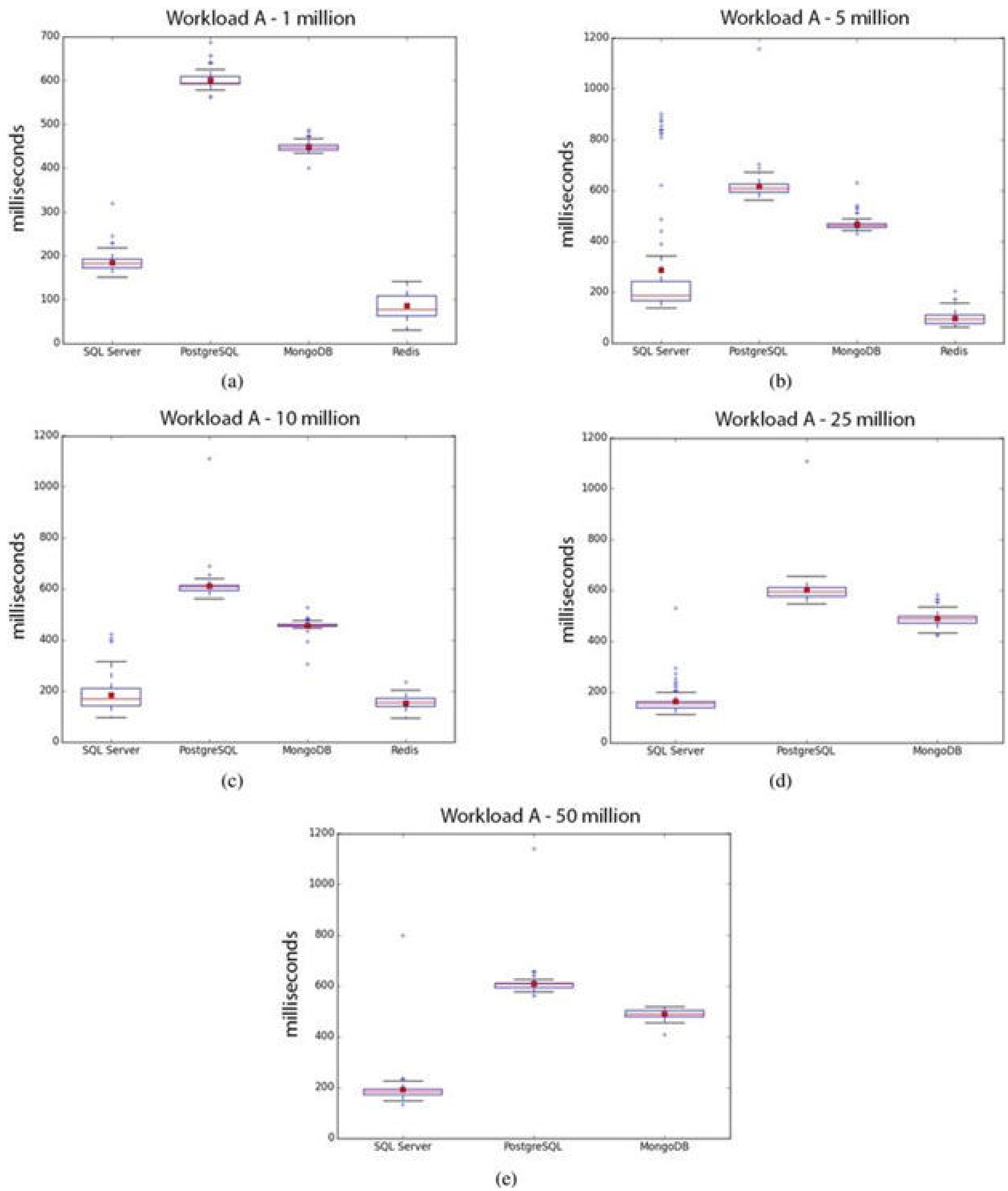


Fig 47. Process time YCSB workload A for different number of records [25]

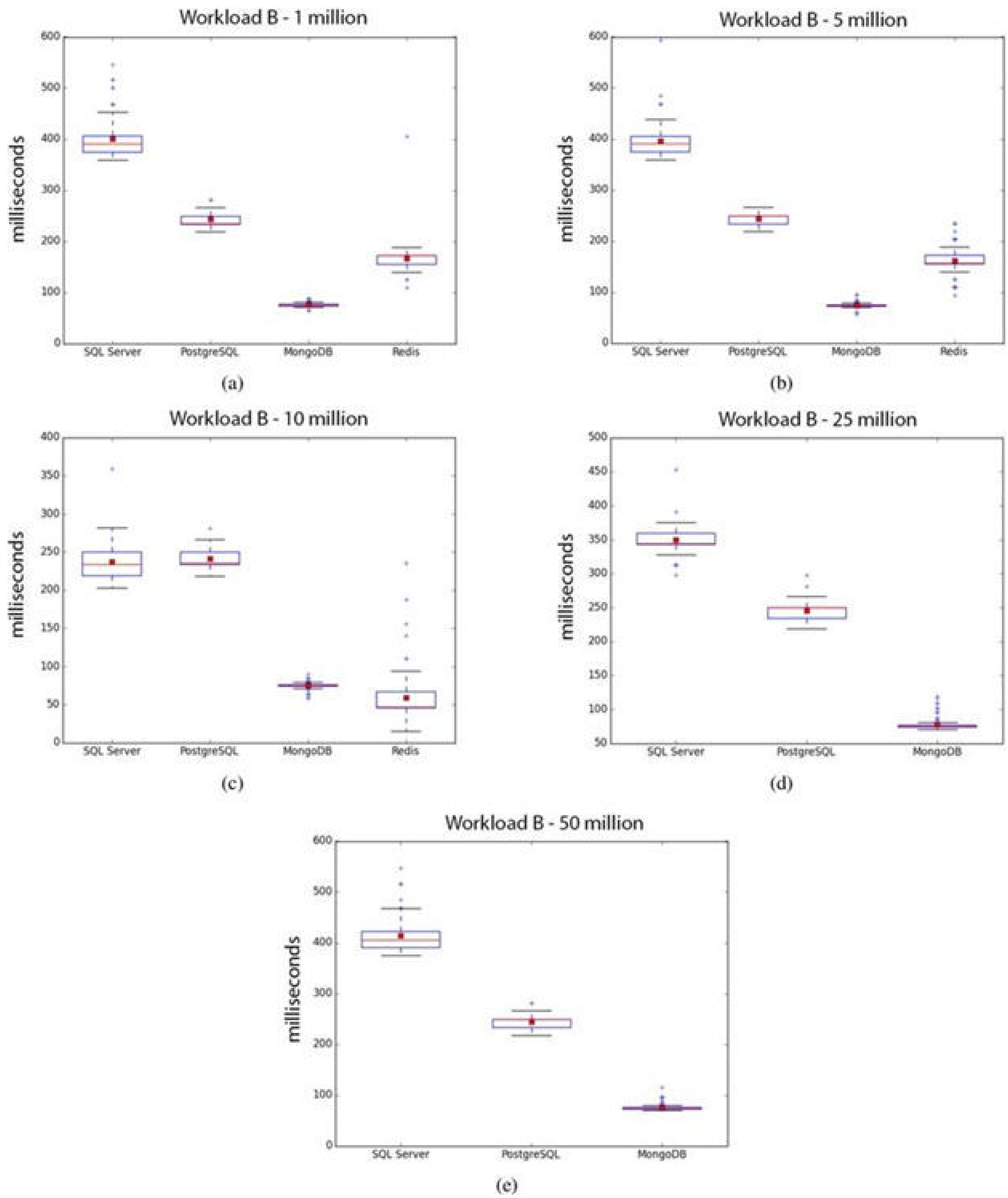


Fig 48. Process time YCSB workload B for different number of records [25]

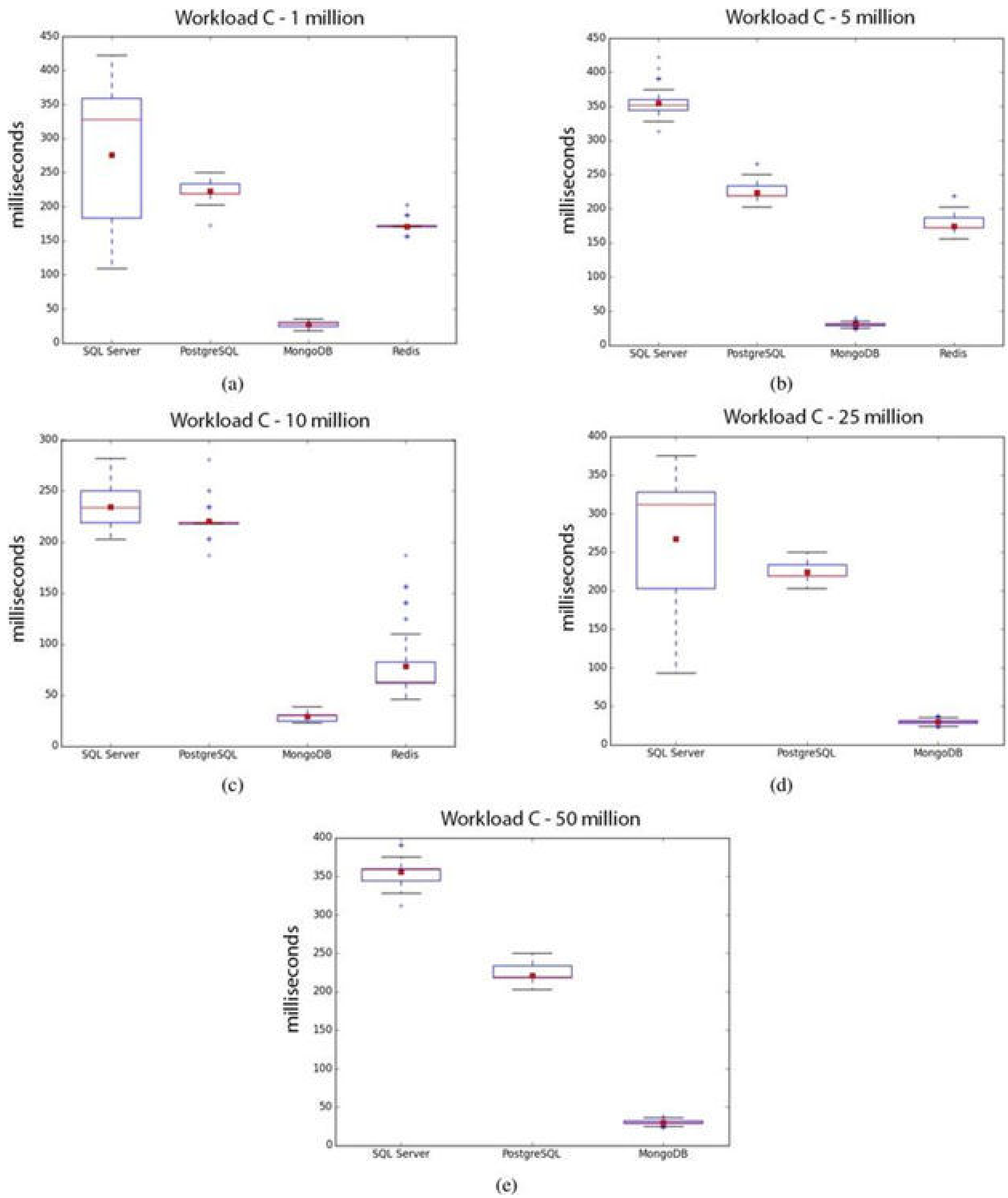


Fig 49. Process time YCSB workload C for different number of records [25]

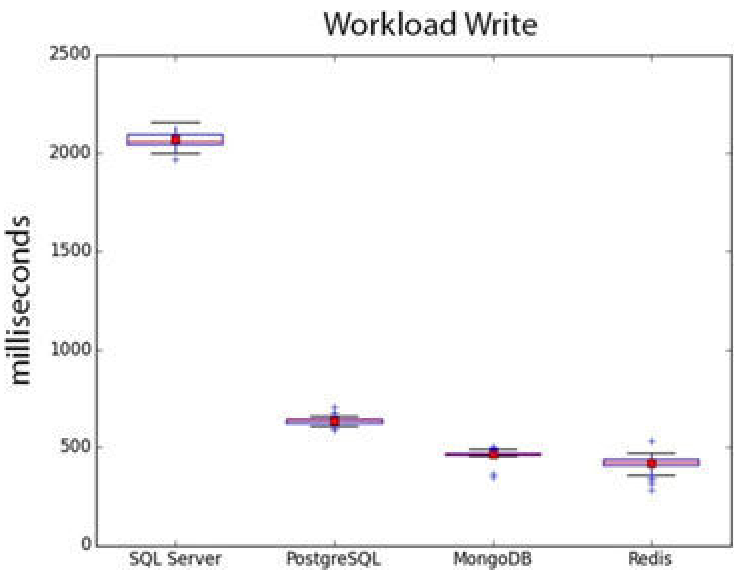


Fig 50. Process time for write heavy workload [25]

K - Database handling effectiveness Query performance graphical results

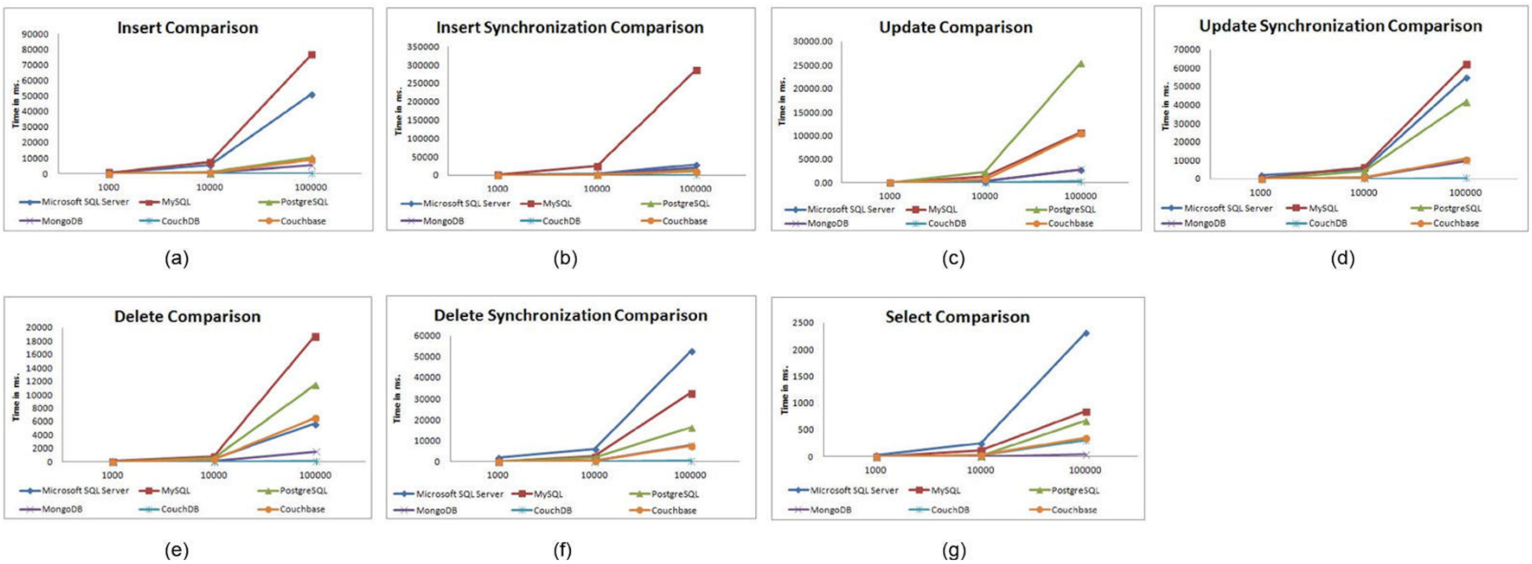


Fig 51. Query performance time (ms) for different CRUD operations in number of records per query (X-axis) [26]. The difference between normal and synchronization comparison is that the synchronization comparison is based on a distributed environment.

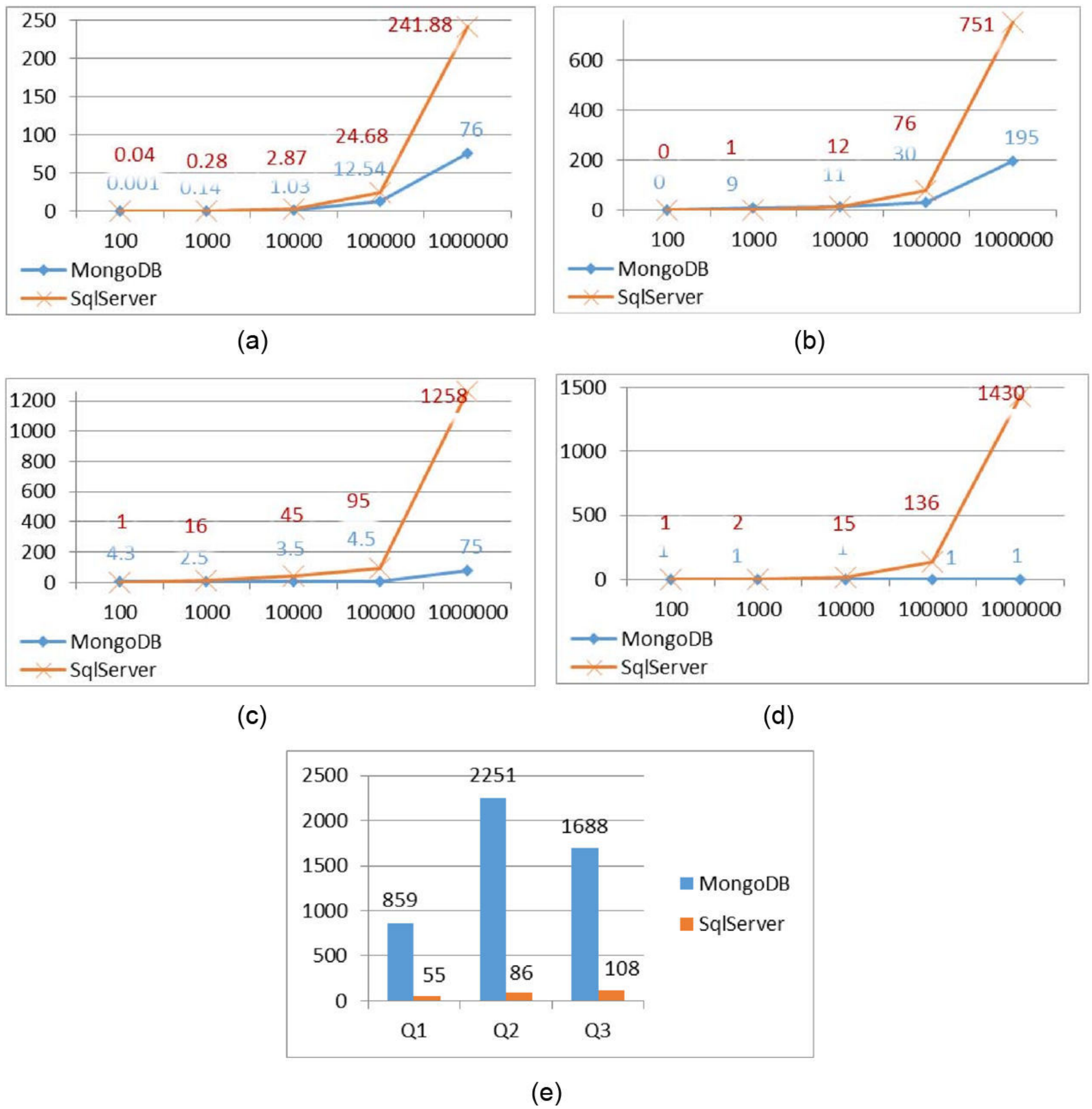


Fig 52. Performance time (ms) (Y-axis) vs. number of records (X-axis) for (a) insert query (b) update query (c) delete query (d) select query. (e) Performance time (ms) (Y-axis) for different aggregate query operations. [27]

L - Dedicated sensor system source code

dedicatedSensorSystem.ino:

```
/*

Dedicated sensor system sketch
Created by: Bram van Berlo

The sketch is created as part of a bachelor end assignment about creating a participatory sensing system for road quality data acquisition. The sketch acts as a basic state machine.

The first state is called the setup state. In this state all of the sensors and serial connections to the gps and ble module are being established and preparations for the calibration state are being taken.
The second state is called the calibration state. In this state the accelerometer and gyroscope are being calibrated in order to make sure they are aligned to the perfect theoretical value as close as possible.
Afterwards the sketch switches to the car_not_driving state. In this state the sensor system checks if the car is driving already or not and validates the gps data and connection to the mpu6050.
If this is the case the car switches to the car_is_driving state.
In the car_is_driving state the sensor system is constantly generating acceleration, orientation and gps data and sending this data through a serial connection to the cc2541 ble chip at the hm-10 module.
If the sensor system notices that the car is not driving anymore it will switch back to the car_not_driving state.

Notes:

- In order for this sketch to work the x and z axis of both 'getMotion6' and 'getAcceleration' functions in MPU6050.cpp have to be switched
- The MPU6050 chip has to be placed in such a way that the x-axis is facing up, y-axis is facing to the right and the z-axis is facing to the front

Created with help of the following tutorials and libraries:

Libraries:
- TinyGPS++ library created by Mikal Hart
- I2Cdev and MPU6050 library created by Jeff Rowberg
- AltSoftSerial library created by Paul Stoffregen
- Statistic library created by Rob Tillaart

Tutorials:
- 'raw to meaningful acceleration and orientation value conversion' by Bence Kaulics and Ajmal Moochingal:
https://electronics.stackexchange.com/questions/39714/how-to-read-a-gyro-accelerometer
- 'MPU-6050 (GY-521) Arduino DMP Tutorial and Calibration' by Tinker Guy: https://www.youtube.com/watch?v=ce3eA8nzInE

*/
```

```cpp
//Library includes
#include <TinyGPS++.h>
#include <I2Cdev.h>
#include <MPU6050.h>
#include <SoftwareSerial.h>
#include <AltSoftSerial.h>
#include <Statistic.h>

// Arduino Wire library is required if I2Cdev I2CDEV_ARDUINO_WIRE implementation
// is used in I2Cdev.h
#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
  #include <Wire.h>
#endif

//Constant definitions
#define CAR_IS_INITIALIZING 0
#define CAR_IS_NOT_DRIVING 1
#define CAR_IS_DRIVING 2

#define TYPE_LOCATION_DATA 1
#define TYPE_ACCELERATION_DATA 2
#define TYPE_ORIENTATION_DATA 3

//Library objects
AltSoftSerial altSerial; //standard pins rx, tx are 8, 9.
SoftwareSerial mySerial2(10, 11); // RX, TX

TinyGPSPlus gps;
MPU6050 accelgyro;

Statistic accXvalues;
Statistic accYvalues;
Statistic accZvalues;
Statistic gyrXvalues;
Statistic gyrYvalues;
Statistic gyrZvalues;

//System variables
double speedThreshold = 4.04;
//double speedThreshold = 0; //Speed threshold used for testing purposes
int currentDriveState = CAR_IS_INITIALIZING;

bool startValueStream = false;
bool stopValueStream = false;

unsigned long currentTime;
unsigned long previousTime;

void setup() {
  //Initialize serial(UART) ports
  altSerial.begin(9600);
  mySerial2.begin(230400);
  Serial.begin(19200);

  //MPU6050 setup section
  // join I2C bus (I2Cdev library doesn't do this automatically)
  #if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
      Wire.begin();
  #elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE
      Fastwire::setup(400, true);
  #endif
  accelgyro.initialize();

  //Clear data buffers needed for MPU6050 calibration
  accXvalues.clear();
  accYvalues.clear();
  accZvalues.clear();
  gyrXvalues.clear();
  gyrYvalues.clear();
  gyrZvalues.clear();
}
```

```
void loop() {
  //Gps object data feed
  while (altSerial.available() > 0)
  gps.encode(altSerial.read());

  //Initialize timer
  currentTime = millis();

  //ValueStream control logic
  if (startValueStream == true) {
    mySerial2.println("start");
    //These serial prints are used as a bug fix in order to slow down serial communication between Arduino and cc2541 (hm-10), this counts for all of the hardware serial prints
    Serial.println("&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&");
    startValueStream = false;
  }
  else if (stopValueStream == true) {
    mySerial2.println("stop");
    Serial.println("%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%");
    stopValueStream = false;
  }

  //State machine system
  switch (currentDriveState) {
    case CAR_IS_INITIALIZING:
      {
        sensorCalibration();
      }
      break;
    case CAR_IS_NOT_DRIVING:
      {
        if (gps.speed.isValid() && gps.location.isValid() && accelgyro.testConnection()) {
          double currentSpeed = gps.speed.kmph();
          if (currentSpeed > speedThreshold) {
            startValueStream = true;
            currentDriveState = CAR_IS_DRIVING;
          }
        }
      }
      break;
    case CAR_IS_DRIVING:
      {
        if (gps.speed.isValid() && gps.location.isValid() && accelgyro.testConnection()) {
          locationData();
          accData();
          gyrData();
          double currentSpeed = gps.speed.kmph();
          if (currentSpeed < speedThreshold) {
            stopValueStream = true;
            currentDriveState = CAR_IS_NOT_DRIVING;
          }
        }
      }
      break;
    default:
      break;
  }
}
```

```
void sensorCalibration() {
  const int calibrationSampleSize = 2000; //Since MPU6050 works at a standard sampling rate of 1000Hz this calibration process will take approx. 2 seconds to complete
  const int errorValues = 100; //The amount of first values unreliable due to the MPU6050 not having stabilized yet
  int16_t AccX, AccY, AccZ;
  int16_t GyrX, GyrY, GyrZ;

  if (accelgyro.testConnection()) {
    //Reset current offsets
    accelgyro.setXAccelOffset(0);
    accelgyro.setYAccelOffset(0);
    accelgyro.setZAccelOffset(0);
    accelgyro.setXGyroOffset(0);
    accelgyro.setYGyroOffset(0);
    accelgyro.setZGyroOffset(0);

    //Accumulate MPU6050 data
    for (int i=0; i<calibrationSampleSize; i++) {
      accelgyro.getMotion6(&AccX, &AccY, &AccZ, &GyrX, &GyrY, &GyrZ);
      if (i<errorValues) {
        continue;
      }
      accXvalues.add(AccX);
      accYvalues.add(AccY);
      accZvalues.add(AccZ);
      gyrXvalues.add(GyrX);
      gyrYvalues.add(GyrY);
      gyrZvalues.add(GyrZ);
    }

    //Orientation of the MPU6050: Xaxis up/down, Yaxis left/right and Zaxis front/back
    int accXoffset = (16384-accZvalues.average())/8;
    int accYoffset = -accYvalues.average()/8;
    int accZoffset = -accYvalues.average()/8;
    int gyrXoffset = -gyrXvalues.average()/4;
    int gyrYoffset = -gyrYvalues.average()/4;
    int gyrZoffset = -gyrZvalues.average()/4;

    //Set offsets
    accelgyro.setXAccelOffset(accXoffset);
    accelgyro.setYAccelOffset(accYoffset);
    accelgyro.setZAccelOffset(accZoffset);
    accelgyro.setXGyroOffset(gyrXoffset);
    accelgyro.setYGyroOffset(gyrYoffset);
    accelgyro.setZGyroOffset(gyrZoffset);

    Serial.println("done calibrating");
    currentDriveState = CAR_IS_NOT_DRIVING;
  }
}


void locationData() {
  if (gps.location.isUpdated() && gps.location.isValid() && gps.speed.isUpdated() && gps.speed.isValid()) {
    long interval = 600;
    if (currentTime - previousTime > interval) {
      Serial.println("gpsPlsssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss");
      mySerial2.println("gpsPls");
      previousTime = currentTime;
    }
  }
}

void accData() {
  int16_t AccX, AccY, AccZ;
  accelgyro.getAcceleration(&AccZ, &AccY, &AccX);

  float AccXadjusted = ((float)AccX / 16384.0)*9.80665; //Acceleration conversion from raw values to m/s2
  float AccYadjusted = ((float)AccY / 16384.0)*9.80665;
  float AccZadjusted = (((float)AccZ / 16384.0)*9.80665)+9.80665;

  String dataWindowString = String(String(TYPE_ACCELERATION_DATA)+","+String(AccXadjusted)+","+String(AccYadjusted)+","+String(AccZadjusted));

  Serial.println(dataWindowString);
  mySerial2.println(dataWindowString);
}

void gyrData() {
  int16_t GyrX, GyrY, GyrZ;
  accelgyro.getRotation(&GyrX, &GyrY, &GyrZ);

  float GyrXadjusted = (float)GyrX / (131.0*57.29577951); //Orientation conversion from raw values to rad/s
  float GyrYadjusted = (float)GyrY / (131.0*57.29577951);
  float GyrZadjusted = (float)GyrZ / (131.0*57.29577951);

  String dataWindowString = String(String(TYPE_ORIENTATION_DATA)+","+String(GyrXadjusted)+","+String(GyrYadjusted)+","+String(GyrZadjusted));

  Serial.println(dataWindowString);
  mySerial2.println(dataWindowString);
}
```

M - Android smartphone application source code

MainActivity.java:

```java
package com.example.bramvberlo.dedicatedsensorsystemtodatabase;

import android.app.Activity;
import android.bluetooth.BluetoothGatt;
import android.bluetooth.BluetoothGattCallback;
import android.bluetooth.BluetoothGattCharacteristic;
import android.bluetooth.BluetoothGattDescriptor;
import android.bluetooth.BluetoothGattService;
import android.bluetooth.BluetoothProfile;
import android.bluetooth.le.BluetoothLeScanner;
import android.bluetooth.le.ScanCallback;
import android.bluetooth.le.ScanResult;
import android.content.ComponentName;
import android.content.Intent;
import android.content.ServiceConnection;
import android.content.pm.PackageManager;
import android.os.Handler;
import android.os.IBinder;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.bluetooth.BluetoothManager;
import android.content.Context;
import android.bluetooth.BluetoothAdapter;
import android.util.SparseArray;
import android.bluetooth.BluetoothDevice;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import android.widget.ViewFlipper;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import java.util.UUID;
import com.example.bramvberlo.dedicatedsensorsystemtodatabase.GpsTimeService.gpsBinder;
import com.example.bramvberlo.dedicatedsensorsystemtodatabase.noSQLdatabaseService.localNoSQLbinder;

/*
    Developer: Bram van Berlo
    Context: CreaTe bachelor graduation project titled 'a participatory sensing system for road quality data acquisition'
    Description: the following class represents the main activity of the complete android app.
    The class controls the UI and bluetooth low energy mechanism to find and connect to the dedicated sensor system.
    Afterwards it will read incoming sensor data and either add it to the noSQL database,
    connect to a new document when the value stream is stopped or retrieve data from the gps service.

    The class has been made with help of the following libraries and tutorials:

    Tutorials:
    - https://www.youtube.com/watch?v=x1y4tEHDwk0&t=988s, Developing Bluetooth Smart Applications for Android Tutorial by Dave Smith
    - https://www.youtube.com/watch?v=0c4jRCm353c, Android App Development for Beginners - 43 - Bound Services by theNewBoston
    - https://www.youtube.com/watch?v=zCj5Qzzex_A, Android App Development for Beginners - 44 - Bound Services Example by theNewBoston
*/
```

```java
public class MainActivity extends AppCompatActivity {

    //Bluetooth low energy variables
    private static final UUID SERIAL_SERVICE = UUID.fromString("0000ffe1-0000-1000-8000-00805f9b34fb");
    private static final UUID SERIAL_SERVICE_CHAR_CONFIG = UUID.fromString("00002902-0000-1000-8000-00805f9b34fb");
    private final int BLUETOOTH_REQUEST = 1;

    //Bluetooth low energy items
    private BluetoothManager bleManager;
    private BluetoothAdapter bleAdapter;
    private BluetoothLeScanner scanner;
    private BluetoothGatt connectedGatt;
    private SparseArray<BluetoothDevice> locatedDevices;
    private ArrayList<BLEobject> bleItems;

    //Ui flipper
    private ViewFlipper UIchanger;

    //Items from first view
    private Handler handler;
    private customListViewAdapter customAdapter;
    private Button scanInitiated;
    private ListView foundDevices;

    //Items from second view
    private Button disconnectFromDevice;
    private TextView dataIncoming;

    //Gps service
    GpsTimeService gps;
    boolean isGpsBound = false;

    //noSQLdatabase service
    noSQLdatabaseService noSQLdatabase;
    boolean isNoSQLdatabaseBound = false;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        //Initializations

        //Bluetooth low energy items
        bleManager = (BluetoothManager)getSystemService(Context.BLUETOOTH_SERVICE);
        bleAdapter = bleManager.getAdapter();
        scanner = bleAdapter.getBluetoothLeScanner();
        locatedDevices = new SparseArray<BluetoothDevice>();
        bleItems = new ArrayList<BLEobject>();

        //Ui flipper
        UIchanger = (ViewFlipper) findViewById(R.id.UIflipper);

        //Items from first view
        scanInitiated = (Button) findViewById(R.id.initiateDeviceScan);
        foundDevices = (ListView) findViewById(R.id.discoveredBleDevices);
        handler = new Handler();
        customAdapter = new customListViewAdapter(this, bleItems);
        foundDevices.setAdapter(customAdapter);

        //Items from second view
        disconnectFromDevice = (Button) findViewById(R.id.connectionClose);
        dataIncoming = (TextView) findViewById(R.id.incomingData);

        //Gps service
        Intent gpsIntent = new Intent(this, GpsTimeService.class);
        bindService(gpsIntent, gpsConnection, Context.BIND_AUTO_CREATE);

        //NoSQL database service
        Intent noSQLdatabaseIntent = new Intent(this, noSQLdatabaseService.class);
        bindService(noSQLdatabaseIntent, noSQLdatabaseConnection, Context.BIND_AUTO_CREATE);
    }
```

```java
@Override
protected void onResume() {
    super.onResume();

    //Check if bluetooth is enabled and enable if necessary
    if (bleAdapter == null || !bleAdapter.isEnabled()) {
        Intent enableBluetooth = new Intent(bleAdapter.ACTION_REQUEST_ENABLE);
        startActivityForResult(enableBluetooth, BlUETOOTH_REQUEST);
        return;
    }

    //Check for BLE support (used for debugging purposes)
    if (!getPackageManager().hasSystemFeature(PackageManager.FEATURE_BLUETOOTH_LE)) {
        Toast.makeText(this, "This smartphone does not support BLE. The app has been terminated", Toast.LENGTH_SHORT).show();
        finish();
        return;
    }

    //Button click listeners

    //Scan for nearby devices and put them in listView when user clicks on 'Scan for devices' button
    scanInitiated.setOnClickListener(new View.OnClickListener(){
        public void onClick(View v){
            locatedDevices.clear();
            customAdapter.clear();
            if (scanner != null) {
                startBleScan();
            }
        }
    });

    //Disconnect gatt connection when user presses 'Disconnect from device' button
    disconnectFromDevice.setOnClickListener(new View.OnClickListener(){
        public void onClick(View v){
            if (connectedGatt != null && bleAdapter != null) {
                connectedGatt.disconnect();
            }
        }
    });

    //ListView item click listener

    //Find BLE device to connect with when user clicks on found BLE device in listView
    foundDevices.setOnItemClickListener(new AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
            int hash = Integer.parseInt(((TextView) view.findViewById(R.id.hashCode)).getText().toString());
            BluetoothDevice chosenDevice = locatedDevices.get(hash);
            if (chosenDevice != null) {
                connectedGatt = chosenDevice.connectGatt(MainActivity.this, false, gattCallBack);
            }
        }
    });
}

//Current ble connection is disconnected when the app is terminated
protected void onStop() {
    super.onStop();
    if (connectedGatt != null) {
        connectedGatt.disconnect();
    }
}

//Runnable used by handler in order to stop ble scan
private Runnable stopRunnable = new Runnable() {
    @Override
    public void run() {
        stopBleScan();
    }
};

private void startBleScan() {
    Toast.makeText(getApplicationContext(), "Searching for ble devices...", Toast.LENGTH_SHORT).show();
    scanner.startScan(scanCallback);
    handler.postDelayed(stopRunnable, 5000); //The program will stop scanning for devices after 5 seconds in order to limit power consumption
}

private void stopBleScan() {
    Toast.makeText(getApplicationContext(), "Search has been stopped", Toast.LENGTH_SHORT).show();
    scanner.stopScan(scanCallback);
}
```

```java
//Callback that handles enable bluetooth request
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (scanner == null) {
        scanner = bleAdapter.getBluetoothLeScanner();
    }

    if (requestCode == BlUETOOTH_REQUEST && resultCode == RESULT_OK) {
        locatedDevices.clear();
        customAdapter.clear();
        if (scanner != null) {
            startBleScan();
        }
    }
    else if (requestCode == BlUETOOTH_REQUEST && resultCode == RESULT_CANCELED) {
        Toast.makeText(getApplicationContext(), "This app requires bluetooth to work. The app has been terminated", Toast.LENGTH_SHORT).show();
        finish();
    }
}

//Callback that fills custom ListView once a ble scan result has been acquired
private final ScanCallback scanCallback = new ScanCallback() {
    @Override
    public void onScanResult(int callbackType, ScanResult result) {
        BluetoothDevice foundDevice = result.getDevice();

        boolean deviceAlreadyFound = false;
        for (int i=0; i<locatedDevices.size(); i++) {
            if (locatedDevices.valueAt(i).getAddress().equals(foundDevice.getAddress())) {
                deviceAlreadyFound = true;
            }
        }

        if (foundDevice != null && !deviceAlreadyFound) {
            locatedDevices.append(foundDevice.hashCode(),foundDevice);
            BLEobject deviceFound = new BLEobject(foundDevice.getName(), foundDevice.getAddress(), foundDevice.hashCode(), result.getRssi());
            customAdapter.add(deviceFound);
        }
    }
};
```

```java
//Bluetooth Gatt handler
private BluetoothGattCallback gattCallBack = new BluetoothGattCallback() {

    Activity mActivity = MainActivity.this;

    //Bug fix for garbage collector not collecting unused variables quick enough
    private Handler handler = new Handler();
    final private int dataSendInterval = 60000; //send interval in milliseconds
    private Boolean handlerIsRunning = false;
    final Runnable runnable = new Runnable() {
        @Override
        public void run() {
            handlerIsRunning = true;
            suggestMemoryClean();
            handler.postDelayed(this, dataSendInterval);
        }
    };

    //Connection state callback
    @Override
    public void onConnectionStateChange(BluetoothGatt gatt, int status, int newState) {
        if (status == BluetoothGatt.GATT_SUCCESS && newState == BluetoothProfile.STATE_CONNECTED) {
            mActivity.runOnUiThread(new Runnable() {
                public void run() {
                    Toast.makeText(getApplicationContext(), "Device connection established, discovering services...", Toast.LENGTH_SHORT).show();
                    UIchanger.showNext();
                }
            });
            gatt.requestConnectionPriority(BluetoothGatt.CONNECTION_PRIORITY_HIGH);
            gatt.discoverServices();
            handler.postDelayed(runnable, dataSendInterval);
        }
        else if (status == BluetoothGatt.GATT_SUCCESS && newState == BluetoothProfile.STATE_DISCONNECTED) {
            try {
                if (connectedGatt != null) {
                    connectedGatt.close();
                    connectedGatt = null;
                }
            }
            catch (Exception e) {
                e.printStackTrace();
            }
            handler.removeCallbacks(runnable);
            mActivity.runOnUiThread(new Runnable() {
                public void run() {
                    Toast.makeText(getApplicationContext(), "Disconnected from device", Toast.LENGTH_SHORT).show();
                    UIchanger.showPrevious();
                }
            });
        }
        else if (status != BluetoothGatt.GATT_SUCCESS) {
            if (handlerIsRunning) {
                handler.removeCallbacks(runnable);
                handlerIsRunning = false;
            }
            mActivity.runOnUiThread(new Runnable() {
                public void run() {
                    Toast.makeText(getApplicationContext(), "A connection error has occurred", Toast.LENGTH_SHORT).show();
                    if (UIchanger.getDisplayedChild() == 1) {
                        UIchanger.showPrevious();
                    }
                }
            });
        }
    }
```

```java
    //Callback initiated when a connection is established and services have been discovered
    @Override
    public void onServicesDiscovered(BluetoothGatt gatt, int status) {
        List<BluetoothGattService> services = gatt.getServices();
        System.out.println(services);
        for (BluetoothGattService service : services) {
            System.out.println(service.toString());
            List<BluetoothGattCharacteristic> characteristics = service.getCharacteristics();
            for (BluetoothGattCharacteristic characteristic : characteristics) {
                UUID uuid = characteristic.getUuid();
                if (uuid.equals(SERIAL_SERVICE)) {
                    gatt.setCharacteristicNotification(characteristic, true);
                    BluetoothGattDescriptor descriptor = characteristic.getDescriptor(SERIAL_SERVICE_CHAR_CONFIG);
                    descriptor.setValue(BluetoothGattDescriptor.ENABLE_NOTIFICATION_VALUE);
                    gatt.writeDescriptor(descriptor);
                    mActivity.runOnUiThread(new Runnable() {
                        public void run() {
                            Toast.makeText(getApplicationContext(), "Service connection established, listening to incoming data...", Toast.LENGTH_SHORT).show();
                        }
                    });
                }
            }
        }
    }

    //Callback that handles update notifications with associated updated sensor value from the decicated sensor system
    @Override
    public void onCharacteristicChanged(BluetoothGatt gatt, BluetoothGattCharacteristic characteristic) {
        super.onCharacteristicChanged(gatt, characteristic);
        if (connectedGatt != null) {
            String timeStamp = String.valueOf(System.currentTimeMillis() + gps.returnGpsTimeOffset());
            String incomingData = new String(characteristic.getValue());

            String[] splitResult = incomingData.split(",");
            List<String> finalResults = new ArrayList<String>();
            finalResults.addAll(Arrays.asList(splitResult));
            finalResults.add(timeStamp);

            switch (finalResults.get(0).trim()) {
                case "gpsPls":
                    if (finalResults.size() == 2 && noSQLdatabase.accessCurrentUsedDocumentId() != null) {
                        String[] gpsData = gps.returnGpsData().split(",");
                        noSQLdatabase.addGpsData(gpsData[0], gpsData[1], gpsData[2]);
                    }
                    break;
                case "2":
                    if (finalResults.size() == 5 && noSQLdatabase.accessCurrentUsedDocumentId() != null) {
                        noSQLdatabase.addAccData(finalResults.get(1).trim(), finalResults.get(2).trim(), finalResults.get(3).trim(), finalResults.get(4).trim());
                    }
                    break;
                case "3":
                    if (finalResults.size() == 5 && noSQLdatabase.accessCurrentUsedDocumentId() != null) {
                        noSQLdatabase.addGyrData(finalResults.get(1).trim(), finalResults.get(2).trim(), finalResults.get(3).trim(), finalResults.get(4).trim());
                    }
                    break;
                case "start":
                    if (finalResults.size() == 2) {
                        noSQLdatabase.startDataStream();
                    }
                    break;
                case "stop":
                    if (finalResults.size() == 2) {
                        noSQLdatabase.stopDataStream();
                    }
                    break;
                default:
                    break;
            }
        }
    }
};

private void suggestMemoryClean() {
    System.gc();
}
```

```java
    //Service connection handlers

    private ServiceConnection gpsConnection = new ServiceConnection() {
        @Override
        public void onServiceConnected(ComponentName name, IBinder service) {
            gpsBinder binder = (gpsBinder) service;
            gps = binder.getService();
            isGpsBound = true;
        }

        @Override
        public void onServiceDisconnected(ComponentName name) {
            isGpsBound = false;
        }
    };

    private ServiceConnection noSQLdatabaseConnection = new ServiceConnection() {
        @Override
        public void onServiceConnected(ComponentName name, IBinder service) {
            localNoSQLbinder binder = (localNoSQLbinder) service;
            noSQLdatabase = binder.getService();
            isNoSQLdatabaseBound = true;
        }

        @Override
        public void onServiceDisconnected(ComponentName name) {
            isNoSQLdatabaseBound = false;
        }
    };
}
```

BLEobject.java:

```java
package com.example.bramvberlo.dedicatedsensorsystemtodatabase;

/*
    Developer: Bram van Berlo
    Context: CreaTe bachelor graduation project titled 'a participatory sensing system for road quality data acquisition'
    Description: The following class implements a custom ListView object that is used to fill the bluetooth low energy result ListView with

    This class has been created with help of the following tutorial:
    - https://github.com/codepath/android_guides/wiki/Using-an-ArrayAdapter-with-ListView
*/

public class BLEobject {

    public String name, addr;
    public int hashcode, rssi;

    public BLEobject(String name, String addr, int hashcode, int rssi) {
        this.name = name;
        this.addr = addr;
        this.hashcode = hashcode;
        this.rssi = rssi;
    }
}
```

customListViewAdapter.java:

```java
package com.example.bramvberlo.dedicatedsensorsystemtodatabase;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.TextView;
import java.util.ArrayList;

/*
    Developer: Bram van Berlo
    Context: CreaTe bachelor graduation project titled 'a participatory sensing system for road quality data acquisition'
    Description: the following class implements a custom ListView adapter that is used to fill a ListView with ble scan results

    This class has been created with help of the following tutorials:
    - https://github.com/codepath/android_guides/wiki/Using-an-ArrayAdapter-with-ListView
    - https://www.youtube.com/watch?v=nOdSARCVYic
*/

public class customListViewAdapter extends ArrayAdapter<BLEobject>{

    public customListViewAdapter(Context context, ArrayList<BLEobject> bleObjects) {
        super(context, R.layout.bledeviceslayout ,bleObjects);
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {

        BLEobject currentObject = getItem(position);

        LayoutInflater inflater = LayoutInflater.from(getContext());
        View view = inflater.inflate(R.layout.bledeviceslayout, parent, false);

        TextView title = (TextView) view.findViewById(R.id.deviceTitle);
        TextView addr = (TextView) view.findViewById(R.id.deviceAddr);
        TextView rssi = (TextView) view.findViewById(R.id.rssiInformation);
        TextView hashcode = (TextView) view.findViewById(R.id.hashCode);

        title.setText(currentObject.name);
        addr.setText(currentObject.addr);
        rssi.setText("Rssi: "+String.valueOf(currentObject.rssi));
        hashcode.setText(String.valueOf(currentObject.hashcode));

        return view;
    }
}
```

GpsTimeService.java:

```java
package com.example.bramvberlo.dedicatedsensorsystemtodatabase;

import android.app.Service;
import android.content.Context;
import android.content.Intent;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.os.Handler;
import android.os.IBinder;
import android.os.Binder;
import android.os.Looper;
import android.widget.Toast;

/*
    Developer: Bram van Berlo
    Context: CreaTe bachelor graduation project titled 'a participatory sensing system for road quality data acquisition'
    Description: The following service class retrieves the gps location and time data from the internal gps chip in the smartphone.
    This data in turn can then be used by the main activity in order to save gps data when the sensor system requests gps data to be saved.

    The class has been made with help of the following libraries and tutorials:

    Tutorials:
    - https://www.youtube.com/watch?v=0c4jRCm353c, Android App Development for Beginners - 43 - Bound Services by theNewBoston
    - https://www.youtube.com/watch?v=zCj5Qzzex_A, Android App Development for Beginners - 44 - Bound Services Example by theNewBoston
    - https://www.youtube.com/watch?v=QNb_3QKSmMk, How To Get GPS Location In Android by Filip Vujovic
*/
```

```java
public class GpsTimeService extends Service {

    private final IBinder gpsClientBinder = new gpsBinder();

    public GpsTimeService() {

    }

    @Override
    public IBinder onBind(Intent intent) {
        return gpsClientBinder;
    }

    private long currentOffset;
    private String timeStamp;
    private String latitude;
    private String longitude;


    Context context;
    Handler handler;
    private LocationManager gpsManager;
    private LocationListener gpsListener;

    public void onCreate() {
        currentOffset = 0;
        context = getBaseContext();
        handler = new Handler(Looper.getMainLooper());

        gpsManager = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
        gpsListener = new LocationListener() {
            @Override
            public void onLocationChanged(Location location) {
                currentOffset = location.getTime()-System.currentTimeMillis();
                timeStamp = String.valueOf(System.currentTimeMillis() + currentOffset);
                latitude = String.valueOf(location.getLatitude());
                longitude = String.valueOf(location.getLongitude());
            }

            @Override
            public void onStatusChanged(String provider, int status, Bundle extras) {

            }

            @Override
            public void onProviderEnabled(String provider) {

            }

            @Override
            public void onProviderDisabled(String provider) {
                if (context != null) {
                    handler.post(new Runnable() {
                        public void run() {
                            Toast.makeText(getApplicationContext(), "GPS services disabled, please enable gps", Toast.LENGTH_SHORT).show();
                        }
                    });
                }
                Intent gpsOptionsIntent = new Intent(android.provider.Settings.ACTION_LOCATION_SOURCE_SETTINGS);
                startActivity(gpsOptionsIntent);
            }
        };
        gpsManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 1, 0, gpsListener);
    }

    long returnGpsTimeOffset() {
        return currentOffset;
    }

    String returnGpsData() {
        return latitude.concat(",").concat(longitude).concat(",").concat(timeStamp);
    }

    public class gpsBinder extends Binder {
        GpsTimeService getService() {
            return GpsTimeService.this;
        }
    }
}
```

noSQLdatabaseService.java:

```java
package com.example.bramvberlo.dedicatedsensorsystemtodatabase;

import android.app.Service;
import android.content.Context;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.Binder;
import android.os.Handler;
import android.os.IBinder;
import com.couchbase.lite.CouchbaseLiteException;
import com.couchbase.lite.Database;
import com.couchbase.lite.Document;
import com.couchbase.lite.Manager;
import com.couchbase.lite.Query;
import com.couchbase.lite.QueryEnumerator;
import com.couchbase.lite.QueryRow;
import com.couchbase.lite.UnsavedRevision;
import com.couchbase.lite.android.AndroidContext;
import com.google.gson.Gson;
import com.google.gson.JsonArray;
import com.google.gson.JsonObject;
import com.google.gson.JsonParser;
import java.io.IOException;
import java.util.Iterator;
import java.util.Map;
import java.util.UUID;
import java.util.concurrent.ConcurrentHashMap;
import okhttp3.Call;
import okhttp3.Callback;
import okhttp3.MediaType;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.RequestBody;
import okhttp3.Response;

/*
 Developer: Bram van Berlo
 Context: CreaTe bachelor graduation project titled 'a participatory sensing system for road quality data acquisition'
 Description: The following service class implements a Couchbase Lite NoSQL embedded database for Android that is used to buffer incoming sensor data through Bluetooth 4.0 when a user
 is driving around and does not have an active Wi-Fi connection. The service class also implements a listener that checks if it is possible to send the saved sensor data through
 a HTTP POST request to the main server once there is an active Wi-Fi connection.

 The class has been made with help of the following libraries and tutorials:

 Libraries:
 - https://developer.couchbase.com/documentation/mobile/current/guides/couchbase-lite/index.html, Couchbase Lite by Couchbase
 - http://square.github.io/okhttp/, OkHttp by Square
 - https://github.com/google/gson, Gson by Google

 Tutorials:
 - https://www.youtube.com/watch?v=mQZuHDUzGMI, Creation of a HTTP POST function by Techval
 - https://www.youtube.com/watch?v=0c4jRCm353c, Android App Development for Beginners - 43 - Bound Services by theNewBoston
 - https://www.youtube.com/watch?v=zCj5Qzzex_A, Android App Development for Beginners - 44 - Bound Services Example by theNewBoston
*/
```

```java
public class noSQLdatabaseService extends Service {

    private final IBinder noSQLbinder = new localNoSQLbinder();

    public noSQLdatabaseService() {

    }

    @Override
    public IBinder onBind(Intent intent) {
        return noSQLbinder;
    }

    Manager databaseManager = null;
    Database noSQLdatabase = null;
    String currentUsedDocument = null;

    private Handler handler;
    final private int dataSendInterval = 5000; //send interval in milliseconds

    OkHttpClient httpClient;
    MediaType json;

    Gson gson;

    public void onCreate() {
        try {
            databaseManager = new Manager(new AndroidContext(getApplicationContext()), Manager.DEFAULT_OPTIONS);
        }
        catch (IOException e) {
            e.printStackTrace();
        }
        try {
            noSQLdatabase = databaseManager.getDatabase("sensor-data-buffer");
        }
        catch (CouchbaseLiteException e) {
            e.printStackTrace();
        }

        httpClient = new OkHttpClient();
        json = MediaType.parse("application/json; charset=utf-8");

        gson = new Gson();

        handler = new Handler();
        final Runnable runnable = new Runnable() {
            @Override
            public void run() {
                sendData();
                handler.postDelayed(this, dataSendInterval);
            }
        };
        handler.postDelayed(runnable, dataSendInterval);
    }

    //Database functions

    void startDataStream() {
        System.out.println("Data stream started");
        Map<String, Object> properties = new ConcurrentHashMap<String, Object>();
        properties.put("gps-data", new ConcurrentHashMap<>());
        properties.put("acceleration-data", new ConcurrentHashMap<String, Map<String,Object>>());
        properties.put("orientation-data", new ConcurrentHashMap<String, Map<String,Object>>());
        Document newDocument = noSQLdatabase.createDocument();
        try {
            newDocument.putProperties(properties);
        }
        catch (CouchbaseLiteException e) {
            e.printStackTrace();
        }
        currentUsedDocument = newDocument.getId();
    }

    void stopDataStream() {
        currentUsedDocument = null;
    }
```

```java
void sendData() {
    if (noSQLdatabase.getDocumentCount() != 0 && currentUsedDocument == null && isWifiConnected()) {
        Query query = noSQLdatabase.createAllDocumentsQuery();
        try {
            QueryEnumerator result = query.run();
            for (Iterator<QueryRow> it = result; it.hasNext();) {
                QueryRow nextElement = it.next();
                Document currentDocument = nextElement.getDocument();
                Map<String, Object> properties = currentDocument.getProperties();

                /*
                 Since Couchbase Lite android does not support the required JSON framework a correct JSON object model is created
                 down below and afterwards send through a HTTP POST request to the Node.js server application connected to the main
                 MongoDB database
                */

                Map<String, Map<String, Object>> accData = (Map<String, Map<String, Object>>) properties.get("acceleration-data");
                JsonArray parsedAccData = new JsonArray();
                Iterator accit = accData.entrySet().iterator();
                while (accit.hasNext()) {
                    Map.Entry entry = (Map.Entry) accit.next();
                    Map<String, Object> currentValue = (Map<String, Object>) entry.getValue();
                    String jsonString = gson.toJson(currentValue);
                    JsonObject jsonElement = new JsonParser().parse(jsonString).getAsJsonObject();
                    parsedAccData.add(jsonElement);
                    accit.remove();
                }

                Map<String, Map<String, Object>> oriData = (Map<String, Map<String, Object>>) properties.get("orientation-data");
                JsonArray parsedOriData = new JsonArray();
                Iterator oriit = oriData.entrySet().iterator();
                while (oriit.hasNext()) {
                    Map.Entry entry = (Map.Entry) oriit.next();
                    Map<String, Object> currentValue = (Map<String, Object>) entry.getValue();
                    String jsonString = gson.toJson(currentValue);
                    JsonObject jsonElement = new JsonParser().parse(jsonString).getAsJsonObject();
                    parsedOriData.add(jsonElement);
                    oriit.remove();
                }

                Map<String, Map<String, Object>> gpsData = (Map<String, Map<String, Object>>) properties.get("gps-data");
                JsonArray parsedGpsData = new JsonArray();
                Iterator gpsit = gpsData.entrySet().iterator();
                while (gpsit.hasNext()) {
                    Map.Entry entry = (Map.Entry) gpsit.next();
                    Map<String, Object> currentValue = (Map<String, Object>) entry.getValue();
                    String jsonString = gson.toJson(currentValue);
                    JsonObject jsonElement = new JsonParser().parse(jsonString).getAsJsonObject();
                    parsedGpsData.add(jsonElement);
                    gpsit.remove();
                }

                JsonObject parsedDocument = new JsonObject();
                parsedDocument.add("accdata", parsedAccData);
                parsedDocument.add("oridata", parsedOriData);
                parsedDocument.add("gpsdata", parsedGpsData);
                String mapToJson = gson.toJson(parsedDocument);
                transferJsonToMainDatabase(mapToJson, "http://5.199.152.8:8083"); //Always recheck this ip and port with node.js application host 'http://5.199.152.8:8083'
                currentDocument.purge();
            }
        }
        catch (CouchbaseLiteException e) {
            e.printStackTrace();
        }
    }
}
```

```java
void addAccData(String AccX, String AccY, String AccZ, String timestamp) {
    final Map<String, Object> dataBundle = new ConcurrentHashMap<String, Object>();
    dataBundle.put("ax", AccX);
    dataBundle.put("ay", AccY);
    dataBundle.put("az", AccZ);
    dataBundle.put("ts", timestamp);
    Document doc = noSQLdatabase.getDocument(currentUsedDocument);
    try {
        doc.update(new Document.DocumentUpdater() {
            @Override
            public boolean update(UnsavedRevision newRevision) {
                Map<String, Object> properties = newRevision.getProperties();
                Map<String, Map<String, Object>> currentSavedItems = (Map<String, Map<String, Object>>) properties.get("acceleration-data");
                currentSavedItems.put(UUID.randomUUID().toString(), dataBundle);
                properties.put("acceleration-data", currentSavedItems);
                newRevision.setProperties(properties);
                return true;
            }
        });
    }
    catch (CouchbaseLiteException e) {
        e.printStackTrace();
    }
}

void addGyrData(String GyrX, String GyrY, String GyrZ, String timestamp) {
    final Map<String, Object> dataBundle = new ConcurrentHashMap<String, Object>();
    dataBundle.put("ox", GyrX);
    dataBundle.put("oy", GyrY);
    dataBundle.put("oz", GyrZ);
    dataBundle.put("ts", timestamp);
    Document doc = noSQLdatabase.getDocument(currentUsedDocument);
    try {
        doc.update(new Document.DocumentUpdater() {
            @Override
            public boolean update(UnsavedRevision newRevision) {
                Map<String, Object> properties = newRevision.getProperties();
                Map<String, Map<String, Object>> currentSavedItems = (Map<String, Map<String, Object>>) properties.get("orientation-data");
                currentSavedItems.put(UUID.randomUUID().toString(), dataBundle);
                properties.put("orientation-data", currentSavedItems);
                newRevision.setProperties(properties);
                return true;
            }
        });
    }
    catch (CouchbaseLiteException e) {
        e.printStackTrace();
    }
}

void addGpsData(String Lat, String Lng, String timestamp) {
    final Map<String, Object> dataBundle = new ConcurrentHashMap<String, Object>();
    dataBundle.put("lat", Lat);
    dataBundle.put("long", Lng);
    dataBundle.put("ts", timestamp);
    Document doc = noSQLdatabase.getDocument(currentUsedDocument);
    try {
        doc.update(new Document.DocumentUpdater() {
            @Override
            public boolean update(UnsavedRevision newRevision) {
                Map<String, Object> properties = newRevision.getProperties();
                Map<String, Map<String, Object>> currentSavedItems = (Map<String, Map<String, Object>>) properties.get("gps-data");
                currentSavedItems.put(UUID.randomUUID().toString(), dataBundle);
                properties.put("gps-data", currentSavedItems);
                newRevision.setProperties(properties);
                return true;
            }
        });
    }
    catch (CouchbaseLiteException e) {
        e.printStackTrace();
    }
}

String accessCurrentUsedDocumentId () {
    return currentUsedDocument;
}
```

```java
    //Utility function that can be used to empty out database at start of application if necessary

    void emptyDatabase() {
        if (noSQLdatabase.getDocumentCount() != 0) {
            Query query = noSQLdatabase.createAllDocumentsQuery();
            try {
                QueryEnumerator result = query.run();
                for (Iterator<QueryRow> it = result; it.hasNext();) {
                    QueryRow nextElement = it.next();
                    Document currentDocument = nextElement.getDocument();
                    currentDocument.purge();
                }
            }
            catch (CouchbaseLiteException e) {
                e.printStackTrace();
            }
            System.out.println("Current stored documents have been removed");
        }
        System.out.println("No documents were found");
    }

    //Network functions

    boolean isWifiConnected() {
        ConnectivityManager connectionManager = (ConnectivityManager) getSystemService(Context.CONNECTIVITY_SERVICE);
        NetworkInfo currentConnections = connectionManager.getActiveNetworkInfo();
        if (currentConnections != null && currentConnections.isConnected() && currentConnections.getType() == ConnectivityManager.TYPE_WIFI) {
            return true;
        }
        else {
            return false;
        }
    }

    void transferJsonToMainDatabase(String jsonString, String destination) {
        RequestBody dataToBeSend = RequestBody.create(json, jsonString);
        Request request = new Request.Builder()
                .url(destination)
                .post(dataToBeSend)
                .addHeader("content-type", "application/json; charset=utf-8")
                .build();

        httpClient.newCall(request).enqueue(new Callback() {
            @Override
            public void onFailure(Call call, IOException e) {
                System.out.println(call.request().body().toString());
            }

            @Override
            public void onResponse(Call call, Response response) throws IOException {
                System.out.println(response.body().string());
            }
        });
    }

    public class localNoSQLbinder extends Binder {
        noSQLdatabaseService getService() {
            return noSQLdatabaseService.this;
        }
    }
}
```

N - Main database application source code

centralEventController.js:

```
/*
    Developer: Bram van Berlo
    Context: CreaTe bachelor graduation project titled 'a participatory sensing system for road quality data acquisition'
    Description: the following program runs a Node.js web server on a certain ip and port. If a post request is coming in
    and it contains viable sensor and gps data it will be saved in a MongoDB database according to a sensor data model.
    Lastly the web server will return either an ok or server error response.

    The following Node.js application has been created with help of the following tutorials:
    - Node.js HTTP: https://nodejs.org/en/docs/guides/anatomy-of-an-http-transaction/
*/

//Dependencies
var http = require('http');
var mongoose = require('mongoose');
var sensorDataDocument = require('./sensorDataModel.js');


//Application functions divided per category

//http
//callback function when http requests are incoming
function requestReceived(request, response) {
    switch(request.method) {
        case 'GET':
            //If get request is incoming it should just return a blank request since the required request is a post request
            console.log("GET request received successfully! Doing nothing with it.");
            postReturn(response, 200, "Ok");
            break;
        case 'POST':
            var bodyData = [];

            request
                .on('error', function(err) {
                    console.error(err);
                })
                .on('data', function(chunk) {
                    bodyData.push(chunk);
                })
                .on('end', function() {
                    //When all incoming data has been pushed into the bodyData array a new BSON document is created that is inserted into the MongoDB database
                    //If this is successfull an ok response is sent back, else a server error is sent back

                    console.log("POST request received successfully!");
                    var jsonString = Buffer.concat(bodyData).toString();
                    var newJsonDocument = JSON.parse(jsonString);
                    var newDocument = new sensorDataDocument();

                    newDocument.accData = newJsonDocument.accdata;
                    newDocument.oriData = newJsonDocument.oridata;
                    newDocument.gpsData = newJsonDocument.gpsdata;

                    newDocument.save(function(err, savedDocument){
                        if (err) {
                            console.error(err);
                            postReturn(response, 500, "Data was not saved successfully on the database");
                        }
                        else {
                            console.log("Document from POST request succesfully saved on MongoDB server.");
                            postReturn(response, 200, "Ok");
                        }
                    });
                });
            break;
        default:
    }
}
```

```
//Response constructor function
function postReturn(response, statusCode, text) {
    response.on('error', function(err) {
        console.error(err);
    });

    //Replying to the incoming request
    response.writeHead(statusCode, {"Content-Type":"text/plain"});
    response.write(text);
    response.end();
}

function getReturn() {

}

//Inits
mongoose.connect('localhost:27017/sensordata');
console.log("MongoDB database connected succesfully");

http.createServer(requestReceived).listen(8083);
console.log("HTTP server initiated and currently listening to port 8083...");
```

sensorDataModel.js:

```
/*
    Developer: Bram van Berlo
    Context: CreaTe bachelor graduation project titled 'a participatory sensing system for road quality data acquisition'
    Description: the following sketch sketches a BSON data model that is used by a Node.js web server in order to save
    sensor and gps data in a MongoDB database.

    This data model has been created with help of the following tutorials:
    - Mongoose documentation: http://mongoosejs.com/docs/guide.html
*/

//Dependencies
var mongoose = require('mongoose');

//Objects
var Schema = mongoose.Schema;
var sensorDataSchema = new Schema({
    accData: [{}],
    oriData: [{}],
    gpsData: [{}]
});

//Export
module.exports = mongoose.model('sensorDataDocument', sensorDataSchema);
```

startApi.bat:

```
::internal ip check to see if forwarded port is still correct.
::If not forward a port and update this port in both the Node.js web server code and android smartphone application.
start cmd.exe /k "ipconfig"

::start MongoDB database -> directories should be the directories where the MongoDB database is stored and a place where you want to store the sensor data
start cmd.exe /k "cd /d D:\Programmas\MongoDB\bin && mongod --dbpath D:\Programmas\MongoDB\data\db"

::forced wait for the MongoDB database to initialize
timeout /t 4 /nobreak > NUL

::start Node.js HTTP web server -> directory should be the directory where the web server source code is stored
start cmd.exe /k "cd /d D:\BachelorGPprototype\mongodb-api && node centralEventController"
```

editMongoDB.js:

```
//After all road quality data has been gathered execute the code below in the MongoDB shell to:
//Turn string values into integer and float values and to order every array based on the timestamp of every object

use sensordata
db.sensordatadocuments.find().forEach(function(currentDocument){
    currentDocument.accData.forEach(function(object){
            object.ax = parseFloat(object.ax);
            object.ay = parseFloat(object.ay);
            object.az = parseFloat(object.az);
            object.ts = parseFloat(object.ts);
        });
    currentDocument.accData.sort(function(a,b){
        return a.ts - b.ts;
    });

    currentDocument.oriData.forEach(function(object){
        object.ox = parseFloat(object.ox);
        object.oy = parseFloat(object.oy);
        object.oz = parseFloat(object.oz);
        object.ts = parseFloat(object.ts);
    });
    currentDocument.oriData.sort(function(a,b){
        return a.ts - b.ts;
    });

    currentDocument.gpsData.forEach(function(object){
        object.lat = parseFloat(object.lat);
        object.long = parseFloat(object.long);
        object.ts = parseFloat(object.ts);
    });
    currentDocument.gpsData.sort(function(a,b){
        return a.ts - b.ts;
    });

    db.sensordatadocuments.save(currentDocument);
});
```

O - User interface source code

Index.html:

```
<!doctype html>
<html>
    <head>
        <meta charset="utf-8" />
        <meta name="title" content="Road quality user interface">
        <meta name="description" content="The following user interface displays road quality in terms of an IRI number for different road segments on a map.">

        <title>Road quality user interface</title>

        <link rel="stylesheet" href="index.css">
        <link rel="stylesheet" href="libraries/magnific-popup.css">

        <script type="text/javascript" src="libraries/jquery-1.11.1.min.js"></script>
        <script type="text/javascript" src="libraries/jquery.magnific-popup.js"></script>
        <script type="text/javascript" src="libraries/color.min.js"></script>
        <script type="text/javascript" src="libraries/download.js"></script>

    </head>
    <body>
        <!-- The user interface control -->
        <div id="controlUi">
            <div id="controlUiContentWrapper">
                <div id="extractButton" class="buttonStyle">
                    <p id="extractButtonText" class="textStyle">Extract road quality data</p>
                </div>
                <div style="border-right: 1px solid #c9c9c9;" id="aboutButton" class="buttonStyle">
                    <p id="extractButtonText" class="textStyle">About</p>
                </div>
            </div>
        </div>


        <!-- The google map -->
        <div id="googleMap"></div>
        <div id="mapLegend">
            <h4 id="legendTitle">IRI (International</h4>
            <h4 id="legendTitle" style="margin-bottom: 15px;">Roughness Index):</h4>
            <div id="gradientShape">
                <hr style="margin-top: 0px;">
                <hr>
                <hr>
                <hr>
            </div>
            <p style="margin-top: 0px;" class="yoloText">10.0</p>
            <p class="yoloText">6.7</p>
            <p class="yoloText">3.3</p>
            <p class="yoloText">0.0</p>
        </div>
        <div id="dataExtractionMessage"></div>
        <div id="dataExtractionPopup" class="small-dialog zoom-anim-dialog mfp-hide">
            <form id="dataExtractionForm" method="post">
                File name:
                <input id="filename" type="text" name="filename"><br><br>
                File extension:
                <input class="radioButtons" type="radio" name="fileextension" value="json"> JSON
                <input class="radioButtons" type="radio" name="fileextension" value="csv"> CSV<br><br>
                <input id="submitButton" type="submit" value="Generate road quality data file">
            </form>
        </div>
        <div id="aboutPopup" class="small-dialog zoom-anim-dialog mfp-hide">
            <h2 class="aboutQuestionTitle">What is the road quality user interface?</h2>
            <p class="aboutQuestionText">
                The road quality user interface lets you inspect the overall road quality
                in terms of an International Roughness Index (IRI for short) for different road segments.
            </p>
            <p class="aboutQuestionText">
                If you are interested in data that is part of a certain area you can initiate a data extraction,
                mark the area of interest and generate a dataset that can be used for your own liking.
            </p>
            <h2 class="aboutQuestionTitle">Are there any known bugs that I need to account for?</h2>
            <p class="aboutQuestionText">
                If you initiate a data extraction please adjust the size of the area selection a little bit
                in order for the data to be extracted properly.
            </p>
        </div>
        <a id="initiatePopup" class="popup-with-zoom-anim" href="#dataExtractionPopup"></a>
        <a id="initiatePopup2" class="popup-with-zoom-anim" href="#aboutPopup"></a>

        <!-- Scripts that need to be defined at the end of the body element for the webpage to load properly -->
        <script src="https://maps.googleapis.com/maps/api/js?key=AIzaSyB_7-II1Njq2qS8i7VFnxffuBmDs21NXRU"></script>
        <script type="text/javascript" src="mapStyle.js"></script>
        <script type="text/javascript" src="index.js"></script>
    </body>
</html>
```

index.css:

```css
* {
    margin: 0;
    padding: 0;
    text-decoration: none;
}

html, body {
    height: 100%;
    background-color: #000;
}

#googleMap {
    height: 100%;
    width: 100%;
}

#mapLegend {
    background-color: #fff;
    font: normal 12px Arial;
    padding: 10px;
    margin: 10px;
    height: 400px;
    border-style: solid;
    border-width: 1px;
    border-color: #dbdbdb;
    box-shadow: 0px 0px 5px #c9c9c9;
}

#gradientShape {
    width: 50px;
    height: 87%;
    background: linear-gradient(hsl(0, 100%, 50%), hsl(60, 100%, 50%), hsl(120, 100%, 50%));
    float: left;
}

hr {
    width: 140%;
    height: 1px;
    border-bottom: none;
    border-right: none;
    border-left: none;
    margin-top: 115px;
}

.yoloText {
    margin-bottom: 97px;
    margin-left: 80px;
}
```

```css
#controlUi {
    position: fixed;
    top: 0;
    right: 0;
    left: 0;
    z-index: 1000;
    height: 60px;
    background-color: #dbdbdb;
    box-shadow: 0px 2px 0px #c9c9c9;
}

#controlUiContentWrapper {
    margin: auto;
    width: 400px;
    height: 60px;
    display: flex;
}

div.buttonStyle {
    background-color: inherit;
    height: inherit;
    width: 200px;
    border-left: 1px solid #c9c9c9;
}

div.buttonStyle:hover {
    background-color: #d0d0d0;
    height: inherit;
    width: 200px;
    transition-duration: 0.3s;
}

p.textStyle {
    font: normal 16px Arial;
    text-align: center;
    padding-top: 19px;
    color: #656565;
}

#dataExtractionForm {
    font: normal 16px Arial;
}

#filename {
    width: 156px;
    margin-left: 68px;
}

.radioButtons {
    margin-left: 40px;
}

#submitButton {
    width: 305px;
    height: 35px;
}
```

```css
.aboutQuestionTitle {
    font: normal 22px Arial;
    margin-bottom: 15px;
}

.aboutQuestionText {
    font: normal 14px Arial;
    margin-bottom: 15px;
}

/*
The code below has been copied from a W3Schools tutorial about
creating Android Toasts in the browser with CSS and Javascript:
https://www.w3schools.com/howto/howto_js_snackbar.asp
*/

#dataExtractionMessage {
    visibility: hidden;
    min-width: 250px;
    height: 29px;
    margin-left: -335px;
    background-color: #fff;
    color: #000;
    text-align: center;
    border-radius: 2px;
    padding: 16px;
    position: fixed;
    z-index: 1;
    left: 50%;
    bottom: 30px;
    font: normal 16px Arial;
    box-shadow: 0px 2px 0px #c9c9c9;
}

#dataExtractionMessage.show {
    visibility: visible;
    -webkit-animation: fadein 0.5s, fadeout 0.5s 2.5s;
    animation: fadein 0.5s, fadeout 0.5s 2.5s;
}

@-webkit-keyframes fadein {
    from {bottom: 0; opacity: 0;}
    to {bottom: 30px; opacity: 1;}
}

@keyframes fadein {
    from {bottom: 0; opacity: 0;}
    to {bottom: 30px; opacity: 1;}
}

@-webkit-keyframes fadeout {
    from {bottom: 30px; opacity: 1;}
    to {bottom: 0; opacity: 0;}
}

@keyframes fadeout {
    from {bottom: 30px; opacity: 1;}
    to {bottom: 0; opacity: 0;}
}
```

index.js:

```
/*
Author: Bram van Berlo
Context: CreaTe bachelor graduation project titled 'a participatory sensing system for road quality data acquisition'
Description: the following program creates a customized google map with IRI data plotted on it.
The user can extract data from the map based on a selected area.

The program has been created with help of the following code tutorials and samples:
- Javascript value map function by Alnitak: https://stackoverflow.com/questions/5649803/remap-or-map-function-in-javascript
- Color.js Javascript library by Matt Wiebe: https://github.com/Automattic/Color.js
- Javascript toast/snackbar tutorial by w3schools: https://www.w3schools.com/howto/howto_js_snackbar.asp
- Magnific popup library by Dmitry Semenov: http://dimsemenov.com/plugins/magnific-popup/
- Google maps javascript API by Google: https://developers.google.com/maps/documentation/javascript/
*/

//Global variables
var map;
var mapData = [];

//jQuery code to execute once page has been loaded
$(document).ready(function() {
    //Insert json data set into array
    $.getJSON("data/roadQualityData.json", function(data) {
        if (typeof google === 'object' && typeof google.maps === 'object') {
            for (x in data.qualityData) {
                mapData.push(data.qualityData[x]);
            }
        }
    });

    //Android toast style popup initialization
    $('.popup-with-zoom-anim').magnificPopup({
        type: 'inline',

        fixedContentPos: false,
        fixedBgPos: true,

        overflowY: 'auto',

        closeBtnInside: true,
        preloader: false,

        midClick: true,
        removalDelay: 300,
        mainClass: 'my-mfp-zoom-in'
    });
});

//Init the Google map when the rest of the page is loaded
google.maps.event.addDomListener(window, 'load', initGoogleMap);
```

```
//Google map init function
function initGoogleMap() {
    //Init google map
    var startLocation = {
        lat: 51.45,
        lng: 5.47
    };
    map = new google.maps.Map(document.getElementById('googleMap'), {
        zoom: 12,
        center: startLocation,
        disableDefaultUI: true,
        mapTypeControlOptions: {
            mapTypeIds: ['roadmap', 'satellite', 'hybrid', 'terrain', 'styled_map']
        }
    });

    //Set map style. Note: style has been generated with the Google Maps API Styling Wizard
    map.mapTypes.set('styled_map', styledMapType);
    map.setMapTypeId('styled_map');

    //Set map legend
    var maplegend = document.getElementById('mapLegend');
    map.controls[google.maps.ControlPosition.RIGHT_CENTER].push(maplegend);

    //Add data to map
    for (x=0; x<mapData.length; x++) {
        addDataPoint(mapData[x]);
    }

    //UI listeners for button click events
    var extractButton = document.getElementById("extractButton");
    google.maps.event.addDomListener(extractButton, 'click', initDataExtraction);

    var aboutButton = document.getElementById("aboutButton");
    google.maps.event.addDomListener(aboutButton, 'click', showInterfaceInformation);
}

//Function takes iri data and associated geo location and for each data point plots
//a circle on the map with a color depending on the iri number
function addDataPoint(dataInput) {
    var latlng = new google.maps.LatLng(dataInput.lat, dataInput.lng);
    var dataMapPlot = new google.maps.Circle({
        strokeColor: iriToHexColor(dataInput.iri),
        strokeOpacity: 0.8,
        strokeWeight: 2.0,
        fillColor: iriToHexColor(dataInput.iri),
        fillOpacity: 1.0,
        map: map,
        center: latlng,
        radius: 3,
        editable: false,
        draggable: false,
        clickable: false
    });
}


//The reason why this function works is that when scaling from red to green only the H(Hue)
//parameter is changed from 0 to 120 degrees, S(Saturation) and V(Value) are always 100%
function iriToHexColor(currentIri) {
    function mapToOtherValueRange(value, low1, high1, low2, high2) {
        return low2 + (high2 - low2) * (value - low1) / (high1 - low1);
    }

    var Hue = mapToOtherValueRange(currentIri, 0, 10, 120, 0);
    var hsv = {
        h: Hue,
        s: 100,
        v: 100
    };

    return Color().fromHsv(hsv).toString();
}

//Function displays a android toast style message on the screen for a certain amount of time
function showToast(message, duration) {
    var extractionMessageId = document.getElementById("dataExtractionMessage");
    extractionMessageId.innerHTML = message;
    extractionMessageId.className = "show";
    setTimeout(function(){ extractionMessageId.className = extractionMessageId.className.replace("show", ""); }, duration);
}
```

```
//This function handles the data selection and data download process
//once a data extraction request has been initiated by the user
function initDataExtraction() {
    showToast("Please put the rectangle on the area of interest. Right click the rectangle when you are done.", 3000);

    //Rectangle that can be used in order to select a preferred
    //area for which data has to be downloaded
    var currentBounds = {
        north: 51.264,
        south: 51.333,
        east: 5.561,
        west: 5.397
    };

    var rectangle = new google.maps.Rectangle({
        map: map,
        bounds: currentBounds,
        editable: true,
        draggable: true,
        clickable: true
    });

    map.panToBounds(currentBounds);


    //When the user right clicks the rectangle a popup is shown so that the user
    //can pick a data set file type and name
    rectangle.addListener('rightclick', function() {
        var currentSelectedData = [];
        var selectedArea = rectangle.getBounds();

        //Retrieve selected data from road quality data array
        for (x=0; x<mapData.length; x++) {
            var dataLocation = new google.maps.LatLng({
                lat: mapData[x].lat,
                lng: mapData[x].lng
            });

            if (selectedArea.contains(dataLocation)) {
                currentSelectedData.push(mapData[x]);
            }
        }
        rectangle.setMap(null);

        if (currentSelectedData.length == 0) {
            showToast("The selected area did not return any valid data. Please try again.", 3000);
        }
        else {
            $('#initiatePopup')[0].click();
            $("#submitButton").click(function() {
                //When the file type and name have been chosen a request is send to php files
                //that will generate the data set file
                //Afterwards a download sequence is started so that the user can download the data
                var filename = document.getElementById('filename').value;
                var fileextension = $('input[name="fileextension"]:checked').val();

                $.ajax({
                    type: "POST",
                    url: "generateDataFile.php",
                    data: {
                        'selectedData': currentSelectedData,
                        'filename': filename,
                        'fileextension': fileextension
                    }
                })
                .success(function(res) {
                    if (downloadFile(res)) {
                        $.post("deleteDataFile.php", {'fileToDelete': res});
                    }
                    showToast("Data file was generated and saved successfully", 3000);
                })
                .error(function(xhr, textStatus) {
                    var http_response = xhr.status;
                    showToast(http_response.toString()+" "+textStatus, 3000);
                })
                return false;
            });
        }
    });
}

function showInterfaceInformation() {
    $('#initiatePopup2')[0].click();
}
```

generateDataFile.php:

```php
<?php
    /*
    Author: Bram van Berlo
    Context: CreaTe bachelor graduation project titled 'a participatory sensing system for road quality data acquisition'
    Description: the following code receives json data through http requests, is turned into either a .json or .csv data set file
    and afterwards the link from where the file can be downloaded is send back through a http response.

    Created with help of the following tutorials:
    - https://www.youtube.com/watch?v=D7KndoW1Tj8, PHP Security: XSS (Cross-site Scripting) by Codecourse
    */

    if ($_SERVER['REQUEST_METHOD'] === 'POST') {
        if (isset($_POST['selectedData'], $_POST['filename'], $_POST['fileextension'])) {

            $selectedData = $_POST['selectedData'];
            $filename = xssEscape($_POST['filename']);
            $fileextension = xssEscape($_POST['fileextension']);
            $baseUrl = ""; //This url needs to be changed for every server this code will be placed on

            switch ($fileextension) {
                case "json":

                    $json_content = json_encode($selectedData);
                    $complete_name = $filename . ".json";
                    $newfile = fopen($complete_name, 'w');
                    fwrite($newfile, $json_content);
                    fclose($newfile);
                    echo $baseUrl . $complete_name;
                    die();

                    break;
                case "csv":

                    $complete_name = $filename . ".csv";
                    $newfile = fopen($complete_name, 'w');
                    $csvheaders = array('Lat','Long','Iri');
                    fputcsv($newfile, $csvheaders);
                    foreach ($selectedData as $currentObject) {
                        $csvdataelements = array();
                        foreach ($currentObject as $currentValue) {
                            array_push($csvdataelements, $currentValue);
                        }
                        fputcsv($newfile, $csvdataelements);
                    }
                    fclose($newfile);
                    echo $baseUrl . $complete_name;
                    die();

                    break;
            }

        }
        else {
            http_response_code(400);
            die();
        }
    }

    else {
        http_response_code(405);
        die();
    }

    //XSS protection
    function xssEscape($value) {
        return htmlspecialchars($value, ENT_QUOTES, 'UTF-8');
    }
?>
```

deleteDataFile.php:

```php
<?php
    /*
    Author: Bram van Berlo
    Context: CreaTe bachelor graduation project titled 'a participatory sensing system for road quality data acquisition'
    Description: the following program receives a link from where a data set file was downloaded and deletes it
    in order to make sure that the server does not get overflown with data set files requested in a road quality user interface instance.

    Created with help of the following tutorials:
    - https://www.youtube.com/watch?v=D7KndoW1Tj8, PHP Security: XSS (Cross-site Scripting) by Codecourse
    */

    if ($_SERVER['REQUEST_METHOD'] === 'POST') {
        if (isset($_POST['fileToDelete'])) {

            $filePath = xssEscape($_POST['fileToDelete']);
            if(unlink($filePath)) {
                die();
            }
            else {
                http_response_code(400);
                die();
            }

        }
        else {
            http_response_code(400);
            die();
        }
    }
    else {
        http_response_code(405);
        die();
    }

    //XSS protection
    function xssEscape($value) {
        return htmlspecialchars($value, ENT_QUOTES, 'UTF-8');
    }
?>
```

P - Dedicated sensor system user acceptance test

Dedicated Sensor System User Acceptance Test

The following user acceptance test is part of a bachelor project about the development of a participatory sensing system for road quality data acquisition. The user acceptance test tries to obtain an answer to the following question: how does a particular dedicated sensor system enclosure shape play a role in the willingness of users to place the dedicated sensor system inside their car? Please fill out the complete test. Do not forget to submit your answers after you are done.

* Required

Are you currently in the possession of a valid driver's license? *

- Yes
- No

How often do you use your car for transportation purposes? (e.g. doing groceries or driving to work) *

- A few times per year
- A few times per month
- A few times per week
- A few times per day

What brand and model car are you currently driving? *

Short textual answer

In what year was your car model produced? *

Short textual answer

Please have a look at the image below. If you are done please move on to the next set of questions.

If you were invited to participate in a participatory sensing system pilot and asked to place one of the two dedicated sensor system enclosures from the image above inside the car cigarette socket in your car, which one would you pick? (imagine both enclosures have the capability to be placed inside a car cigarette socket) *

- 1.
- 2.
- Neither one
- I do not mind using either one of them

Please explain your answer *

Long textual answer

Thank you for filling out this survey!

References

[1] “Highway Specialists Collect Thousands of Lane Kilometres of Road Data,” *Fugro*. [Online]. Available: https://www.fugro.com/about-fugro/our-expertise/our-work/highway-specialists-collect-thousands-of-lane-kilometres-of-road-data. [Accessed: 11-Apr-2017].

[2] “Home,” *Miramap*. [Online]. Available: http://www.miramap.com/. [Accessed: 11-Apr-2017].

[3] “Products,” *StreetSaver*. [Online]. Available: https://www.streetsaveronline.com/products/streetsaver. [Accessed: 11-Apr-2017].

[4] “Waze,” *Wikipedia*, 29-Mar-2017. [Online]. Available: https://en.wikipedia.org/wiki/Waze. [Accessed: 11-Apr-2017].

[5] F. Seraj, B. J. van der Zwaag, A. Dilo, T. Luarasi, and P. Havinga, “RoADS: A Road Pavement Monitoring System for Anomaly Detection Using Smart Phones,” in *Big Data Analytics in the Social and Ubiquitous Context*, vol. 9546, M. Atzmueller, A. Chin, F. Janssen, I. Schweizer, and C. Trattner, Eds. Cham: Springer International Publishing, 2016, pp. 128–146.

[6] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, and H. Balakrishnan, “The Pothole Patrol: Using a Mobile Sensor Network for Road Surface Monitoring,” in *Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services*, Breckenridge, Colorado, USA, 2008, pp. 29–39.

[7] F. Ackermann and C. Eden, “Strategic Management of Stakeholders: Theory and Practice,” *Long Range Plann.*, vol. 44, no. 3, pp. 179–196, Jun. 2011.

[8] “Thunderboard React Kit | Silicon Labs.” [Online]. Available: http://www.silabs.com/products/development-tools/wireless/bluetooth/thunderboard-react-kit-sensor-cloud-connectivity. [Accessed: 10-Mar-2017].

[9] eZ Systems, “nRF52 Preview DK / Bluetooth low energy / Products / Home - Ultra Low Power Wireless Solutions from NORDIC SEMICONDUCTOR.” [Online]. Available: https://www.nordicsemi.com/eng/Products/Bluetooth-low-energy/nRF52-Preview-DK. [Accessed: 11-Mar-2017].

[10] “IMUduino Wireless 3D motion, BLE, 10 DoF IMU, HTML5, Arduino,” *Kickstarter*. [Online]. Available: https://www.kickstarter.com/projects/1265095814/imuduino-wireless-3d-motion-html-js-apps-arduino-p. [Accessed: 11-Mar-2017].

[11] “Generic Attributes (GATT) and the Generic Attribute Profile,” *Bluetooth*. [Online]. Available: https://www.bluetooth.com/specifications/generic-attributes-overview. [Accessed: 19-Jun-2017].

[12] JGraph Ltd., *draw.io*. 2017.

[13] C. W. Bytheway, *FAST Creativity and Innovation: Rapidly Improving Processes, Product Development and Solving Complex Problems*. J. Ross Publishing, 2007.

[14] D. Kieras, “Model-based evaluation,” in *The Human - Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications*, Third., J. Jacko, Ed. Boca Raton, Florida: CRC Press, Taylor and Francis, 2012, pp. 1299–1318.

[15] G. Booch, J. Rumbaugh, and I. Jacobson, *The Unified Modeling Language User Guide*. Addison-Wesley Professional, 2005.

[16] u-blox AG GNSS Software Research and Development Team, *ublox u-center Windows*.

2017.
[17] MathWorks, *MATLAB R2016a*. 2016.
[18] R Foundation for Statistical Computing, *R*. 2015.
[19] J. T. McClave and T. Sincich, *Statistics*. Pearson, 2011.
[20] Google, *Android Studio*. 2017.
[21] A. Kahlkopf, “iconmonstr - Free simple icons for your next project,” *iconmonstr*. [Online]. Available: https://iconmonstr.com/. [Accessed: 02-Jul-2017].
[22] T. A. M. Phan, J. K. Nurminen, and M. Di Francesco, “Cloud Databases for Internet-of-Things Data,” in *2014 IEEE International Conference on Internet of Things(iThings), and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom)*, Taipei, Taiwan, pp. 117–124.
[23] R. Panda, C. Erb, M. LeBeane, J. H. Ryoo, and L. K. John, “Performance Characterization of Modern Databases on Out-of-Order CPUs,” in *2015 27th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)*, Florianopolis, Brazil, pp. 114–121.
[24] J. S. van der Veen, B. van der Waaij, and R. J. Meijer, “Sensor Data Storage Performance: SQL or NoSQL, Physical or Virtual,” in *2012 IEEE Fifth International Conference on Cloud Computing*, Honolulu, HI, USA, pp. 431–438.
[25] F. Leung and B. Zhou, “Performance evaluation of Twitter datasets on SQL and NoSQL DBMS,” *WEB*, vol. 14, no. 4, pp. 275–286.
[26] C.-O. Truica, F. Radulescu, A. Boicea, and I. Bucur, “Performance Evaluation for CRUD Operations in Asynchronously Replicated Document Oriented Database,” in *2015 20th International Conference on Control Systems and Computer Science*, Bucharest, Romania, pp. 191–196.
[27] S. H. Aboutorabi, M. Rezapour, M. Moradi, and N. Ghadiri, “Performance evaluation of SQL and MongoDB databases for big e-commerce data,” in *2015 International Symposium on Computer Science and Software Engineering (CSSE)*, Tabriz, Iran, pp. 1–7.
[28] D. Clegg and R. Barker, *CASE Method Fast-track: A RAD Approach*. Addison-Wesley, 1994.
[29] S. Pugh, “Concept selection: a method that works,” in *Review of design methodology*, Rome, 1981, pp. 497–506.
[30] P. Múčka, “International Roughness Index specifications around the world,” *Road Materials and Pavement Design*, pp. 1–37, Jun. 2016.
[31] P. Mohan, V. N. Padmanabhan, and R. Ramjee, “Nericell: rich monitoring of road and traffic conditions using mobile smartphones,” in *Proceedings of the 6th ACM conference on Embedded network sensor systems - SenSys ’08*, Raleigh, NC, USA, 2008, p. 323.
[32] G. Klunder, W. van Aalst, D. Willemsen, D. van Vliet, G. Leegwater, and P. Ruijs, “Auto als Sensor – Quick-scan naar detectiemogelijkheden van wegdekschade,” Rijkswaterstaat DVS , Aug. 2010.
[33] “ARAN - Automatic Road Analyzer,” *Furgo*. [Online]. Available: https://www.fugro.com/about-fugro/our-expertise/technology/aran-automatic-road-analyzer. [Accessed: 11-Apr-2017].
[34] “Home,” *Wegdekscan*. [Online]. Available: http://wegdekscan.nl/. [Accessed: 11-Apr-2017].
[35] A. Consulting, *StreetSaver Online Overview*. USA: StreetSaverOnline, 2011.
[36] “About us,” *Waze*. [Online]. Available: https://www.waze.com/about. [Accessed: 11-Apr-2017].
[37] “Waze Connected Citizens Program Fact Sheet.” Waze, 11-Feb-2016.
[38] K. D. Atherton, “Israeli Students Spoof Waze App With Fake Traffic Jam,” *Popular Science*,

31-Mar-2014. [Online]. Available: http://www.popsci.com/article/gadgets/israeli-students-spoof-waze-app-fake-traffic-jam. [Accessed: 11-Apr-2017].
[39] M. Fleeman, “Traffic app maker Waze rejects criticism by L.A. police chief,” *Investing.com*, 27-Jan-2015. [Online]. Available: https://www.investing.com/news/technology-news/traffic-app-maker-waze-rejects-criticism-by-l.a.-police-chief-325374. [Accessed: 11-Apr-2017].
[40] K. Roose, “Did Google Just Buy a Dangerous Driving App?,” *New York Magazine*, 14-Jun-2013. [Online]. Available: http://nymag.com/daily/intelligencer/2013/06/did-google-just-buy-a-dangerous-driving-app.html. [Accessed: 11-Apr-2017].
[41] D. L. Strayer, J. M. Cooper, J. Turrill, J. Coleman, N. MedeirosWard, and F. Biondi, “Measuring Cognitive Distraction in the Automobile,” University of Utah, Jun. 2013.
[42] S. Miles, “Stakeholder: Essentially Contested or Just Confused?,” *J. Bus. Ethics*, vol. 108, no. 3, pp. 285–298, Jul. 2012.
[43] R. K. Mitchell, B. R. Agle, and D. J. Wood, “TOWARD A THEORY OF STAKEHOLDER IDENTIFICATION AND SALIENCE: DEFINING THE PRINCIPLE OF WHO AND WHAT REALLY COUNTS,” *Acad. Manage. Rev.*, vol. 22, no. 4, pp. 853–886, 1997.
[44] S. Miles, “Stakeholder Definitions: Profusion and Confusion,” *EIASM 1st Interdisciplinary conference on stakeholders, resources and value creation, IESE Business School, University of Navarra, Barcelona*, June 7-8, 2011.
[45] R. E. Freeman, *Strategic management: a stakeholder approach*. 1984.
[46] “Chapter 7, section 8: Identifying and Analyzing Stakeholders and Their Interests,” *Community Toolbox*, 2014. [Online]. Available: http://ctb.ku.edu/en/table-of-contents/participation/encouraging-involvement/identify-stakeholders/main. [Accessed: 04-Apr-2017].
[47] T. H. Byers, R. C. Dorf, and A. J. Nelson, *Technology Ventures: From Idea to Enterprise*. McGraw-Hill Education, 2014.
[48] P. Edwards, *How streets, roads, and avenues are different*. Vox, 2016.
[49] “Wegbeheer in Nederland,” *Rijkswaterstaat, Ministerie van Infrastructuur en Milieu*. [Online]. Available: https://www.rijkswaterstaat.nl/wegen/wegbeheer/wegbeheer-in-nederland. [Accessed: 05-Apr-2017].
[50] “Wegbeheer in Nederland,” *Rijkswaterstaat, Ministerie van Infrastructuur en Milieu*. [Online]. Available: https://www.rijkswaterstaat.nl/wegen/wegbeheer/wegbeheer-in-nederland. [Accessed: 05-Apr-2017].
[51] P. Edwards, *How streets, roads, and avenues are different*. Vox, 2016.
[52] T. H. Byers, R. C. Dorf, and A. J. Nelson, *Technology Ventures: From Idea to Enterprise*. McGraw-Hill Education, 2014.
[53] *Pexels*. [Online]. Available: https://www.pexels.com. [Accessed: 05-Apr-2017].
[54] K. H, “What is your intelligence type? – the 9 types explained,” *Psych2go*, 10-Feb-2017. [Online]. Available: https://www.psych2go.net/intelligence-types-explained/. [Accessed: 05-Apr-2017].
[55] *Pexels*. [Online]. Available: https://www.pexels.com. [Accessed: 05-Apr-2017].
[56] “UG164: ThunderboardTM React (RD-0057-0201) User’s Guide,” Silicon Labs.
[57] eZ Systems, “nRF52 DK / Bluetooth low energy / Products / Home - Ultra Low Power Wireless Solutions from NORDIC SEMICONDUCTOR.” [Online]. Available: https://www.nordicsemi.com/eng/Products/Bluetooth-low-energy/nRF52-DK. [Accessed: 10-Mar-2017].

[58] “specifications,” *Bluetooth*. [Online]. Available: https://www.bluetooth.com/specifications. [Accessed: 19-Jun-2017].
[59] K. Aviles, *Bluetooth Low Energy App Development: The Basics*. 2016.
[60] K. Roukounaki, “Five popular databases for mobile,” *Developer Economics*, 10-Sep-2014. [Online]. Available: https://www.developereconomics.com/five-popular-databases-for-mobile. [Accessed: 19-Jun-2017].
[61] Windows Apps Team and J. Carnahan, “When to use a HTTP call instead of a WebSocket (or HTTP 2.0),” *Windows blogs*, 14-Mar-2016. [Online]. Available: https://blogs.windows.com/buildingapps/2016/03/14/when-to-use-a-http-call-instead-of-a-websocket-or-http-2-0/. [Accessed: 19-Jun-2017].
[62] “MPU-6500 Product Specification Revision 1.1,” InvenSense Inc., May 2014.
[63] “NEO-6 u-blox 6 GPS Modules,” U-blox, 2011.
[64] C. Wodehouse, “SQL vs. NoSQL Databases: What’s the Difference?,” *Upwork*, 09-May-2016. [Online]. Available: https://www.upwork.com/hiring/data/sql-vs-nosql-databases-whats-the-difference/. [Accessed: 11-Apr-2017].
[65] J. Berrington, “Databases,” *Anaesthesia & Intensive Care Medicine*, vol. 18, no. 3, pp. 155–157, 2017.
[66] G. Harrison, *Next Generation Databases: NoSQLand Big Data*. Apress, 2015.
[67] P. P. Srivastava, S. Goyal, and A. Kumar, “Analysis of various NoSql database,” in *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*, Greater Noida, Delhi, India, pp. 539–544.
[68] C. JMTauro, S. Aravindh, and S. A.B, “Comparative Study of the New Generation, Agile, Scalable, High Performance NOSQL Databases,” *IJCAI* , vol. 48, no. 20, pp. 1–4, Jun. 2012.
[69] A. Corbellini, C. Mateos, A. Zunino, D. Godoy, and S. Schiaffino, “Persisting big-data: The NoSQL landscape,” *Inf. Syst.*, vol. 63, pp. 1–23, Jan. 2017.
[70] G. C. Deka, “A Survey of Cloud Database Systems,” *IT Prof.*, vol. 16, no. 2, pp. 50–57.
[71] J. Bhogal and I. Choksi, “Handling Big Data Using NoSQL,” in *2015 IEEE 29th International Conference on Advanced Information Networking and Applications Workshops*, Gwangiu, South Korea, pp. 393–398.
[72] “Computer performance,” *Wikipedia*, 22-Feb-2017. [Online]. Available: https://en.wikipedia.org/wiki/Computer_performance. [Accessed: 11-Apr-2017].
[73] S. Parikh and K. Stirman, “Schema Design for Time Series Data in MongoDB,” *MongoDB*, 30-Oct-2013. [Online]. Available: https://www.mongodb.com/blog/post/schema-design-for-time-series-data-in-mongodb. [Accessed: 11-Apr-2017].
[74] hideki, “Lightweight, embedded, syncable NoSQL database engine for Android,” *GitHub*, 10-Mar-2017. [Online]. Available: https://github.com/couchbase/couchbase-lite-android. [Accessed: 11-Apr-2017].
[75] Anubis, B. Kaulics, and A. Moochingal, “How to read a gyro/accelerometer,” *StackExchange electrical engineering*, 22-Jun-2015. [Online]. Available: https://electronics.stackexchange.com/questions/39714/how-to-read-a-gyro-accelerometer. [Accessed: 22-Jun-2017].
[76] S. S. Shapiro and M. B. Wilk, “An analysis of variance test for normality (complete samples),” *Biometrika*, vol. 52, no. 3–4, pp. 591–611, Dec. 1965.
[77] Tinker Guy, *MPU-6050 (GY-521) Arduino DMP Tutorial and Calibration*. 2015.

[78] E. Njus, “Why Portland anarchists are patching potholed streets,” *The Oregonian / OregonLive*, 15-Mar-2017. [Online]. Available: http://www.oregonlive.com/commuting/index.ssf/2017/03/why_portland_anarchists_are_pa.html. [Accessed: 16-Jun-2017].
[79] R. Anderson *et al.*, “Measuring the Cost of Cybercrime,” in *The Economics of Information Security and Privacy*, R. Böhme, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 265–300.
[80] K. S. Yeo, M. C. Chian, T. C. Wee Ng, and D. A. Tuan, “Internet of Things: Trends, challenges and applications,” in *2014 International Symposium on Integrated Circuits (ISIC)*, Singapore, pp. 568–571.
[81] “What is data linkage?,” *Scottish Government*, 24-Sep-2014. [Online]. Available: http://www.gov.scot/Topics/Statistics/datalinkageframework/Whatdatalinkageis. [Accessed: 16-Jun-2017].
[82] P. Bouckaert, “License to Kill, Philippine Police Killings in Duterte’s ‘War on Drugs,’” *Human Rights Watch*, 01-Mar-2017. [Online]. Available: https://www.hrw.org/report/2017/03/01/license-kill/philippine-police-killings-dutertes-war-drugs. [Accessed: 16-Jun-2017].
[83] G. Dworkin, “Paternalism,” *Stanford Encyclopedia of Philosophy*, 12-Feb-2017. [Online]. Available: https://plato.stanford.edu/entries/paternalism/#HardVsSoftPate. [Accessed: 17-Jun-2017].
[84] V. Matta, M. Di Mauro, and M. Longo, “DDoS Attacks With Randomized Traffic Innovation: Botnet Identification Challenges and Strategies,” *IEEE Trans.Inform.Forensic Secur.*, vol. 12, no. 8, pp. 1844–1859, Aug. 2017.
[85] J. Pijpker and H. Vranken, “The Role of Internet Service Providers in Botnet Mitigation,” in *2016 European Intelligence and Security Informatics Conference (EISIC)*, Uppsala, Sweden, 2016, pp. 24–31.
[86] “Fox-IT | For a more secure society,” *Fox-IT*. [Online]. Available: https://www.fox-it.com/nl/. [Accessed: 17-Jun-2017].
[87] K. Verhoeven, “Het Internet der Dingen | Maak apparaten veilig!,” D66, Nov. 2016.
[88] J. Driver, “The History of Utilitarianism,” *Stanford Encyclopedia of Philosophy*, 22-Sep-2014. [Online]. Available: https://plato.stanford.edu/entries/utilitarianism-history/. [Accessed: 17-Jun-2017].
[89] “SHA-2,” *Wikipedia*, June 16, 2017 (Last edited). [Online]. Available: https://en.wikipedia.org/wiki/SHA-2. [Accessed: 17-Jun-2017].
[90] M. Kemp, “Assessment of the results of national reports in light of the Commission Guidelines on evaluating AES and Defeat Devices,” The Greens | European Free Alliance, Feb. 2017.
[91] C. LeRouge, J. Ma, S. Sneha, and K. Tolle, “User profiles and personas in the design and development of consumer health technologies,” *Int. J. Med. Inform.*, vol. 82, no. 11, pp. e251–68, Nov. 2013.
[92] “ThunderBoard BGM111 Sensor Module schematics,” Silicon Labs, Mar. 2016.