Mälardalen University
School of Innovation Design and Engineering
Västerås, Sweden

Thesis for the Degree of Master of Science in Engineering -
Software Engineering 30.0 credits

Event-Based Messaging Architecture for Vehicular Internet of Things (IoT) Platforms

Meera Aravind
mad15006@student.mdh.se

Examiner: Sasikumar Punnekkat
Mälardalen University, Västerås, Sweden

Supervisor 1: Radu Dobrin
Mälardalen University, Västerås, Sweden

Supervisor 2: Jakob Palmheden,
Scania AB, SE-151 87 Södertälje, Sweden

June 23, 2017

Abstract

Internet of Things (IoT) has revolutionized transportation systems by connecting vehicles consequently enabling their tracking, as well as monitoring of driver activities. The IoT platform for most vehicles typically consists of 1) an on-board system consisting of the communication unit, sensors and a set of ECUs that are interconnected using a CAN network, 2) an off-board system consisting of the applications deployed on the servers (e.g., cloud) that processes the data send by the communication unit over the internet, and 3) mobile devices like a mobile phone or a computer that communicates with the on-board and off-board systems. Such an IoT platform requires a significant amount of data to be send from the on-board system to the off-board servers, contributing to high network usage.

There are two main architectural paradigms for sending data: 1) interval based architecture, in which data is send at regular intervals and 2) event based architecture, in which data is send whenever relevant events occur. Currently, (e.g., at Scania), the data is being send at regular intervals, i.e., using an interval based approach. In this case, data is send even if it is not relevant for reporting leading to a wastage of network resources, e.g., when the data does not change considerably compared to the previously sent value. Sending data in an event-based manner, when the data is relevant for reporting, e.g., changes significantly, reduces the network usage when compared to the interval based approach.

In this thesis, we investigate the possibility of using an event based architecture to send data from the on-board system to the off-board system in order to reduce network usage and improve the accuracy of the data available off-board. We first propose an event based architecture for data transfer in the context of Internet of vehicles. We then implement a simulator to evaluate our proposed architecture for the specific case of position data. Finally, we perform extensive experiments varying different parameters and compare, for example, average message size per minute and average off-board error distance.

The results show that our event based architecture improves the accuracy of data available at the off-board system, by a careful selection of events. Moreover, we found that our event based architecture significantly decreases the frequency of sending messages, particularly during highway driving, leading to reduced average data transfer rates. Our results enable a customer to perform trade-offs between accuracy and data transfer rates.

Future work will aim at implementing the event based architecture on a real platform as well as investigating the possibility of using the event based architecture for more accurate prediction by incorporating additional details such as the final destination of the vehicle and odometer values.

Acknowledgements

I would like to express my sincere gratitude to my supervisor at the University, Radu Dobrin, for his support and valuable feedback that helped me improve this thesis. I would also like to thank my supervisor at Scania, Jakob Palmheden and other members of the REIA team for all their support, encouragement and guidance during the thesis. Last but not the least, I would also like to thank my husband and all my family for their unceasing encouragement.

Table of Contents

1	Introduction	7
1.1	Motivation	8
1.2	Problem Formulation	9
1.3	Thesis Organization	10
2	Background	10
2.1	Internet Of Things	10
2.2	Event-Based Systems	11
2.3	The Global Positioning System (GPS)	12
2.4	Geographic Coordinate System	12
2.5	Interval-Based Architecture	13
3	Research Methodology	14
4	State of the Art	16
5	Proposed Event-Based Architecture for Internet of Vehicles	18
5.1	Event-Based Architecture for Position Data	18
5.2	Estimation Algorithm used in the Estimator	20
5.3	Delaying the off-board vehicle representation	20
5.4	Quality Attributes of the Software Architecture	22
6	Simulator Implementation	23
6.1	Simulator Input file	23
6.2	Input Parameters	24
6.3	Estimator Algorithm	25
6.4	Estimation validation	26
6.5	User Interface	26
6.6	Latency for position updates	28
7	Experimental Results and Analysis	30
7.1	Experimental Goals	30
7.2	Experiments without Latency	31
7.2.1	Variation of Average Off-board Error Distance	32
	Interval-Based:	32
	Event-Based without delay:	32
	Event-Based with delay:	33
7.2.2	Variation of Number of Estimated Incorrect Routes	34
	Interval-Based:	34
	Event-Based without delay:	34
	Event-Based with delay:	35
7.2.3	Variation of Average Message Size (Bytes)/minute	36
	Interval-Based:	36
	Event-Based without delay:	37

	Event-Based with delay:	37
7.2.4	Variation of Average Message Transfer Interval	37
	Interval-Based:	38
	Event-Based without delay:	38
	Event-Based with delay:	38
7.2.5	Variation of Total Number of Messages	39
	Interval-Based:	40
	Event-Based without delay:	40
	Event-Based with delay:	40
7.3	Experiments with Latency	42
7.3.1	Variation of Average Off-board Error distance	42
	Interval-Based:	42
	Event-Based without delay:	42
	Event-Based with delay:	42
7.3.2	Variation of Number of Estimated Incorrect Routes	44
	Interval-Based:	44
	Event-Based without delay:	44
	Event-Based with delay:	44
7.3.3	Variation of Average Message Size (Bytes)/minute	46
	Interval-Based:	46
	Event-Based without delay:	46
	Event-Based with delay:	46
7.3.4	Variation of Average Message Transfer Interval	46
	Interval-Based:	47
	Event-Based without delay:	47
	Event-Based with delay:	47
7.3.5	Variation of Total Number of Messages	47
	Interval-Based:	47
	Event-Based without delay:	49
	Event-Based with delay:	49
7.4	Discussion	49
8	Conclusions	53
8.1	Future work	53
8.2	Threats to validity	55
	References	57

List of Figures

1	The extended vehicle.	8
2	Trace position sampling.	13
3	Architecture of the interval-based system.	14
4	Overview of the research method followed.	15
5	Variants of dead-reckoning protocol.	17
6	The proposed architecture of event based on-board and off-board system.	18
7	The proposed architecture of the event based system for position data.	19
8	On-board and off-board position at time t_2	21
9	On-board and off-board position at time t_3	21
10	On-board and off-board position at time t_4	21
11	The simulator user interface	29
12	The latency distribution curve.	30
13	Average off-board error distance variation with changing interval in interval-based architecture.	32
14	Average off-board error distance variation with changing maximum allowed error distance in event-based architecture.	33
15	Average off-board error distance variation with changing maximum allowed error distance and delay in event-based architecture.	34
16	Number of estimated incorrect routes variation with changing maximum allowed error distance in event-based architecture.	35
17	Number of estimated incorrect routes variation with changing maximum allowed error distance and delay in event-based architecture.	35
18	Average message size in bytes/min variation with changing interval in interval-based architecture.	36
19	Average message size in bytes/minute variation with changing maximum allowed error distance in event-based architecture.	37
20	Average message size variation with changing maximum allowed error distance and delay in event-based architecture.	38
21	Average message transfer interval variation with changing maximum allowed error distance in event-based architecture.	39
22	Average message transfer interval variation with changing maximum allowed error distance and delay in event-based algorithm.	39
23	Number of messages sent with changing interval in interval-based architecture.	40
24	Total number of messages with changing maximum allowed error distance and delay in event-based architecture with latency.	41
25	Number of messages sent with changing maximum allowed error distance and delay in event-based architecture.	41
26	Average error distance variation with changing interval in interval-based architecture with latency.	43
27	Average error distance variation with changing maximum allowed error distance in event-based architecture with out delay and with latency.	43

28	Average off-board error distance variation with changing maximum allowed error distance and delay in event-based algorithm with latency.	44
29	Number of estimated incorrect routes variation with changing maximum allowed error distance in event-based architecture with out delay and with latency.	45
30	The difference in number of incorrect routes with changing maximum allowed error distance and delay in event-based algorithm with latency.	45
31	Average message size in bytes/minute variation with changing maximum allowed error distance in event-based architecture with out delay and with latency.	46
32	Average message size in bytes/minute variation with changing maximum allowed error distance and delay in event-based algorithm with latency.	47
33	Average message transfer interval variation with changing maximum allowed error distance in event-based architecture with out delay and with latency.	48
34	Average message transfer interval variation with changing maximum allowed error distance and delay in event-based architecture with latency.	48
35	Total number of messages sent with changing interval in interval-based architecture with latency.	49
36	Total number of messages with changing maximum allowed error distance in event-based architecture with latency.	50
37	Total number of messages with changing maximum allowed error distance and delay in event-based architecture with latency.	50
38	Relative average off-board error distance vs relative data consumption	51
39	Comparison of the average message size/minute in interval based and event based approaches	52
40	Messages sent while the vehicle is on the highway.	54
41	Messages sent while the vehicle is on roundabout.	54

List of Tables

1	The simulator input parameters.	25
3	The latency values used in experiment.	28
2	Error messages and reasons.	28

1 Introduction

The advent of technology and wide availability of Internet has resulted in an increased number of connected devices via the Internet. The interconnected network of physical smart devices, such as sensors, actuators or other devices embedded with electronics is referred to as Internet of Things (IoT). IoT presents a wide range of opportunities to improve human life, in addition to reducing the environmental impact of new technologies. This is achieved by centralized/distributed control of these interconnected devices with the aim of improving efficiency and reducing the overall resource consumption.

The advances in IoT has further motivated the connection of critical infrastructures such as transportation systems to the Internet. This enables fleets of interconnected vehicles that can, for example, reduce fuel consumption by efficient platooning. Platooning can be defined as grouping the vehicles so that they can travel close to each other in an efficient, safe and coordinated manner reducing fuel consumption and carbon dioxide emission [1]. These connected vehicles are equipped with Internet access or wireless local area network, and can communicate with the environment around them by exchanging information with other vehicles and devices. They have the potential to significantly improve road safety, traffic management and environmental implications. Moreover, the connected vehicles supply relevant information to the driver enabling them to take more informed decisions. In order analyze the vehicle related data for better transport efficiency, they can be send from the vehicle and stored externally.

IoT has revolutionized the transport industry making it safer, greener and smarter [2]. The IoT platform for most vehicles typically consists of 1) an on-board system, 2) an off-board system and 3) and mobile devices that communicates with the on-board and off-board systems. The on-board system consists of the communication unit, sensors and a set of ECUs that are interconnected using a CAN network [3]. The communication unit installed (in Scania vehicles today) communicates using 2G and 3G mobile networks, in addition to Bluetooth or Wifi for short range communications. The off-board system consists of the applications deployed on the servers (e.g., cloud) that processes the data send by the communication unit over the internet. Mobile devices include mobile phones or computers that are used to interact with the vehicle (on-board and off-board systems). The off-board and on-board systems together are called as *the extended vehicle* as shown in figure 1 [4].

The extended vehicle concept represents the coherent entirety of the road vehicle, off-board systems, external interfaces and the data communication between the vehicle and off-board systems [4]. The shortcomings of the telecom network can be covered-up by the off-board unit making the data available to data consumers even in case of network coverage problems. The sensors in the on-board system measure driver behaviors and the state of vehicle. Cameras, radar sensor, speed sensors, ultrasound range sensors, heat sensors, steering angle sensor, the throttle position or breaking pedal force sensors are some examples of sensors present in the vehicle. The communication unit transfers the different data collected continuously during the operation of the vehicle to the off-board system. The current state-of-practice at Scania is to send the data at pre-defined regular intervals. In the rest of the thesis,

Figure 1: The extended vehicle.

this is referred to as interval based messaging.

This master thesis was performed at the Connected Services - Architecture and Analytics team (REIA) at Scania Research and development in Södertälje, Sweden. Scania is one of the globally leading manufacturers of heavy vehicles (trucks and buses) and operates in more than 100 countries with around 46,000 employees. The head office of Scania is located in Södertälje, Sweden [5]. The REIA group is responsible for descriptions and design of the systems architecture for parts of the Scania IoT platform. REIA work in cooperation with architects, development engineers and system developers throughout the entire Scania. The thesis was done in collaboration with Gustav Wiklander from Uppsala University. My specific contributions to the thesis include designing of the software architecture together with Gustav, framing and implementing initial version of the estimation algorithm, understanding and implementing the current algorithm used at Scania in the simulator, executing parts of the simulation runs and preparation of the graphs. Moreover, I conducted a separate literature survey, as well as wrote this thesis document entirely.

1.1 Motivation

One challenge of using telecom networks to send data to the off-board system is the scarce availability of network resources such as bandwidth and network coverage. Moreover, sending data over the telecom network incurs data usage charges, which can increase with the amount of data, e.g., particularly when the vehicle is roaming. Consequently, sending data periodically may imply that redundant data is sent, which increases network usage (e.g., when the data remains the same over a long time interval).

Sending data in an event-based manner when the data is relevant for reporting (for example, deviates from estimated value or change considerably compared to previously sent value) instead of fixed intervals is expected to reduce the network

usage. If the amount of data transferred is lowered, it will also substantially reduce the cost of sending data. It can also potentially improve the accuracy of the data since the data is transmitted whenever there is a change. Therefore, the aim of this thesis is to design an event based messaging architecture for connected vehicles and evaluate it with respect to already existing interval based messaging system (in the context of Scania).

In order to limit the scope of this thesis, we consider only the *position data*. Position data of the vehicle is a special case which is obtained from GPS and send to off-board system at regular intervals. It consists of vehicle id, latitude, longitude, speed, heading, altitude and timestamp. This data is used, e.g., for live tracking of the vehicle and to analyze the behavior of the vehicle in different road conditions. The amount of position data sent can be reduced by using suitable dead reckoning algorithms to estimate the position of vehicle. A *dead reckoning algorithm* estimates the current position of a moving object using previously known position and calculates the progress made using other known or estimated values such as speed, direction of travel etc. Executing such an algorithm in both on-board and off-board devices, and sending data only when the estimated value deviates from the actual position is expected to reduce the amount of position data that is send to the data-center.

Therefore, in this thesis we focus on improving the accuracy of the position data available at the off-board system and reducing the cost of its transmission by developing an event based architecture.

1.2 Problem Formulation

The goal is to focus on the following specific problems.

Problem 1: How can an event-based architecture be designed for sending data from the on-board system to the off-board system in vehicular IoT platforms?

Problem 2: What methods and algorithms are necessary for such an event-based messaging platform?

Problem 3: How much accuracy improvement and data reduction can be obtained from the proposed event-based messaging platform, in the context of Scania's vehicular IoT, when compared to the existing interval-based architecture?

We again remind the reader that, in order to have a concrete use case, we focus only on the position data.

Solutions to the first two problems were achieved by first conducting literature surveys and understanding existing system design and available algorithms. Then a suitable event-based architecture and estimation algorithm was designed. For the third question, a simulation environment was developed to compare the existing interval based system with the proposed event based system. Different metrics were defined for evaluating the accuracy improvement and data reduction.

1.3 Thesis Organization

The remaining part of the thesis report is organized into the following sections: Section 2 provides a general overview of the concepts that the reader has to familiarize with. Section 3 discusses the methodology adopted to complete the thesis systematically. Some of the research already conducted in the field is presented in section 4. The proposed event based architecture and estimation algorithm elucidated in section 5. The simulator, designed to evaluate the architecture and algorithm, and its implementation is explained in section 6. Section 7 discusses different experiments conducted, their outcomes and evaluation of the results. The thesis report concludes with section 8 and it consists of a summary of the experiments conducted, suggestions for future work and the factors that can affect the validity of the results.

2 Background

In this section we briefly discuss some of the relevant concepts for the thesis.

2.1 Internet Of Things

The Internet of Things is expected to transform existing industrial systems into a whole new level by offering advanced connectivity of various devices. A commonly used definition of IoT from [6] is stated below:

Definition 1. A dynamic global network infrastructure with self-configuring capabilities based on standard and interoperable communication protocols where physical and virtual ‘Things’ have identities, physical attributes, and virtual personalities and use intelligent interfaces, and are seamlessly integrated into the information network. [6]

Xu et. al proposes a four layered Service Oriented Architecture (SOA) for IoT. It consists of the following layers [7]:

1. Sensing Layer: This layer can sense and acquire data from the physical world and consists of sensors, RFID (Radio-frequency identification) etc.
2. Networking Layer: This layer connects different devices together and provides basic networking support to enable data transfer between them.
3. Service Layer: This layer creates, supports and provides various services according to application requirements and user needs.
4. Interface Layer: This layer supports the interaction with users and other applications on the network.

Industrial applications of IoT have been evolving rapidly. Currently, IoT plays a major role in transportation industry enabling tracking of vehicle location, monitoring its movement and predicting future position. Security and privacy issues are the major concerns of adopting IoT in transport industry as it can lead to leakage of private data.

2.2 Event-Based Systems

A discrete system can be categorized as time based or event based systems. In time-based system, also known as interval based system, samples are generated periodically leading to possibly high number of computations that provide no new information [8]. However, in an event based system, computations are triggered only when certain events take place that has the potential to change the state of the system. This will in turn improve the resource utilization efficiency.

Chandy [9] defines event as a significant change in the state of the universe. Event-Based systems are also known as event driven systems or sense respond systems. In [10], Michelson defines event as a notable thing that occurs inside or outside the system and may imply a problem, a threshold, an opportunity or a deviation. Each event occurrence can have an event header and an event body. The event header describes the event occurrence with the help of event specification id, event time stamp etc. The event body describes what has happened.

Events can be categorized as periodic, stochastic or sporadic depending on their arrival pattern. Periodic events occur at regular intervals of time and therefore are predictable about their occurrence. Events are stochastic if they occur according to some probabilistic distribution pattern. Sporadic events are unpredictable and do not follow any pattern [11].

An event based architecture includes production of events, detection of events and response to events. Such architectures are extremely loose coupled and highly distributed. The producers of the event have no knowledge about the consumers of the event or how they are processed. The event driven architecture is built of four logical layers [10]:

1. Event Generators: The event generators are the source of events and could be an application, data store, sensor etc.
2. Event Channel: The event channel transports the events between event generators, event processing engines, and downstream subscribers.
3. Event Processing: The event processing layer evaluates the events against certain processing rules, and actions are initiated if necessary.
4. Downstream Event Driven Activity: This includes the activity that is initiated as a consequence of the event.

According to Kolios et al., the basic event based system architecture for IoT applications consists of a local host and remote host. The local host represents the embedded device that can monitor and control the physical system. The remote host can be the servers, also other devices that manages or tracks the system. Both the local and remote host is assumed to have the model of the system behavior. The local host can detect any deviation from the expected behaviour and inform the remote host on updated system behavior [12].

In this thesis, we focus on the position data and the data used for verification, in order to detect events, is obtained from the on-board GPS. A detailed description of how the events are created follows in section 5.

2.3 The Global Positioning System (GPS)

GPS is a satellite-based navigation system that is used to locate positions on the earth's surface. GPS is owned by United States government and falls under the category Global Navigation Satellite Systems (GNSS). The GPS system consists of the GPS satellites, the control system operated by the U.S. military, and the GPS equipment used by users.

There are 32 GPS satellites orbiting around earth as of February 2016 [13]. The original GPS design had 24 GPS satellites orbiting the earth at an altitude of about 11,000 nautical miles, with four satellites in each of the 6 orbital planes. Each GPS satellite had 4 atomic clocks out of which only one is used at a time. The GPS satellites continuously broadcast radio signals with satellite position and timing data. The GPS receiver in the user equipment measures the time taken by the signals to travel from satellite to the receiver and calculates the distance to each of the satellites. Three spheres are defined with each having a radii equal to the distance from the receiver to each of the three satellites and centre as the position of each satellite. The three spheres intersect at 2 common points which could be the possible locations of the GPS receiver. In most cases, one of the points can be discarded as it will be nowhere near the earth. The measurement from fourth satellite is required to correct the clock error. A GPS receiver needs signals from atleast two satellites to be able to estimate its location [14]. The location of the receiver is represented using geographic coordinate system discussed in section 2.4.

Accuracy of GPS can be affected by a number of possible sources of errors. The speed of GPS signals can be altered by the ionosphere and troposphere layers affecting its travel time. Electrical interference or inherent errors in the GPS receiver can cause distortion of the GPS signals. Incorrect information about satellite orbits (ephemeris data) can affect the computations. Satellite clock drifts can lead to large computation errors. Bouncing off of the satellite signals from reflective surfaces can cause multipath effects [14].

2.4 Geographic Coordinate System

A geographical coordinate system considers earth as a three dimensional spherical surface and every point on the earths surface can be represented using latitudes and longitudes. Longitude and latitude are measured as angles from the center of the earth to a point on the surface of the earth. Any point on the surface of the earth can be represented using combination of these two components. Graticule is the grid formed by lines of latitude and longitude.

Definition 2. The *latitude* of a point is defined as the angle between the equatorial plane and the line joining the point with the centre of the earth.

Definition 3. The *longitude* of a point is the angle between the prime meridian to the meridian that passes through the point on the earth's surface.

The latitude values can range from from -90° at the south pole to $+90^\circ$ at the north pole. The longitude value can range from -180° to $+180^\circ$ when travelling to west and east respectively

Figure 2: Trace position sampling.

2.5 Interval-Based Architecture

The current architecture adopted for the state of practice at Scania for sending messages from on-board to off-board system at Scania is interval based. Position messages are sent to the off-board system at pre-defined intervals as requested by the customers. There are two different algorithms for collecting the position samples:

1. **Scheduled positioning algorithm:** If the vehicle key has been switched on for the specified interval, e.g., 60 seconds, a position message is sent to the off-board system and that is considered as the commencement of interval. A position sample is collected periodically according to the scheduled period and sent immediately to the off-board system along with the previously collected samples.
2. **Trace positioning algorithm:** This sampling algorithm is dependent on the speed of the vehicle as shown in figure 2. If the vehicle speed is between 6 and 30 km/h, then GPS is sampled every minute. If the speed goes above 30 km/h, then samples are created at an interval in seconds calculated as 2 times the speed of the vehicle. If the speed of vehicle goes below 2 km/h, then a position sample is created immediately with tag “vehicle stopped”. If the speed again goes above 6 km/h, then another position sample is created with tag “vehicle not stopped”. No samples are created between these. Note that the first algorithm may create a sample if its timer expires.

Figure 3 shows the architecture diagram of current interval based system at Scania. Trace position sampler collects position samples according to the trace positioning algorithm and the scheduled position sampler works according to the scheduled positioning algorithm. The collected samples are stored in the output buffer and sent to the datacenter at the completion of the interval. Position samples are also sent to the off-board system in cases when the positioning buffer is full

Figure 3: Architecture of the interval-based system.

(maximum limit is 60). Hence, a position message can contain upto 60 different position samples. The latest one is referred to as the position head and the rest constitute the position tail. A position sample consists of location coordinates, speed, heading, altitude and timestamp.

All position messages are stored in the datacenter and used by different applications for providing various services to the users.

3 Research Methodology

As described by Shaw in [15] the research approach for proposing a new technique or method in software engineering is to:

"Invent new ways to do some tasks, including procedures and implementation techniques. Develop a technique to choose among alternatives."[15]

In this thesis, we are proposing new event based architecture and algorithms for updating location information from the on-board to the off-board system. Hence, it falls into the above category of classification of research product by Shaw [15]. The research method followed in this thesis as illustrated in the figure 4 is based on the systematic steps described by Dodog-Crnkovic [16].

The primary step of the thesis was to understand the problem. This step included performing literature studies both internal and external to Scania. Then the research

Figure 4: Overview of the research method followed.

questions were framed which narrowed down the focus of the thesis. This process was repeated with the purpose of understanding the researches conducted in the field of designing an event based system for position updates. With the gathered information we designed an event based architecture and estimation algorithm for updating position from on-board to off-board system upon event triggers. This was followed by designing and implementing an initial version of the simulation environment for comparing the proposed event based system with existing interval based system. Experiments were performed in the simulation environment and obtained results were recorded. The results were evaluated and presented at Scania. Based on the feedback, further modifications were done on the solution and the simulation environment. Then new set of experiments were conducted and the results were analyzed followed by further fine tuning of the simulation environment. These steps were repeated and the experiments were conducted a number of times. The final results were analyzed and conclusions were drawn. The final results are

presented in this thesis.

4 State of the Art

Vehicular Internet of Things is an emerging field of research that has the potential to provide a wide range of benefits, e.g., improve fuel efficiency by understanding the driver behaviors. Gerla et al., [17] discusses the evolution of urban fleet of vehicles from a collection of sensor platforms to a network of connected vehicles that can exchange their information in order to improve performance and safety with minimum impact on environment.

Remember that, in this thesis, we limit the scope of this thesis to the position data, and focus on designing an event based architecture for sending position data whenever the estimated position changes from the actual value. Several algorithms and techniques have been designed to estimate the position of a vehicle using the input from various sensors. GPS is the most commonly used vehicle positioning system. However, in outdoor environments such as forested areas, near tall buildings or in tunnel environments, the GPS signals may not be reachable.

Li and Xu [18] proposes a novel fusion positioning strategy for estimating position by integrating multiple low-cost sensors such as GPS, MEMS-based (Micro-Electro-Mechanical Systems) Inertial Measurement Unit and a sliding mode observer. Chio et al., [19] presents a dead-reckoning (DR) positioning approach based on inertial-measurement-unit (IMU) and improving the accuracy of the position using Character Recognition algorithm by extracting road names of street signs.

Several researches were conducted to integrate map details along with the sensor data to improve the accuracy of position estimation. Map matching algorithms uses a high resolution spatial road network map along with the various filters (kalman, particle etc) to estimate absolute position of the vehicle with enhanced accuracy. Davidson et al., [20] proposes a particle filter based map matching algorithm and compares it with the position data obtained from GPS. A similar approach of estimating position using low cost sensor equipment and digitally stored map is discussed in [21]. In this paper, the author evaluates both point-mass filter method and sequential Monte Carlo method for estimation and obtain position estimates with accuracy comparable with that of GPS.

All of these algorithms are for estimating position from the vehicle with the help of different sensor values that can be read from the vehicle. However, for this thesis we require an algorithm that can estimate the position from the off-board system as well which will not have access to the latest sensor values.

As noted before, a *dead reckoning algorithm* estimates the current position of a moving object using previously known position and calculates the progress made using other known or estimated values such as speed, direction of travel etc.

Leonhardi et al., [22] discuss different predictor corrector approaches for reducing the number of update messages for transmitting location information of mobile objects. They also proposed a classification of dead-reckoning protocols as illustrated in Figure 5. In the following, we explain this classification:

- *Linear prediction*: This is the simplest protocol and assumes that the vehicle

Figure 5: Variants of dead-reckoning protocol.

is moving along a straight line with last known speed and direction from the last reported position.

- *Higher order prediction function:* This protocol can predict the speed of vehicle considering the acceleration and could capture its movement on a curved road.
- *Map-based dead reckoning/ Model-based dead reckoning:* In this protocol, the position of the vehicle is integrated with available road map with the assumption that the vehicle always moves on the road with the last reported speed. It can also be referred to as model-based dead reckoning where map is used as the model.
- *Map-based with probability information:* The probabilities of users following a certain link of road at an intersection is determined and this information is used to enhance map. Estimation of the position is done assuming that the vehicle follows the link with highest probability.
- *History-based dead-reckoning:* In this protocol, the prediction of the position is done based on the user's past movements. It is assumed that the users will use the same route repeatedly.
- *Dead-reckoning with known route:* The route of the vehicle will be known in advance and the prediction can be done based on its velocity.

In this thesis we use model-based dead reckoning approach for estimation of the position of the vehicle assuming that the vehicle is always on the road.

5 Proposed Event-Based Architecture for Internet of Vehicles

In this thesis we propose a general event based architecture consisting of the following 4 components:

1. Estimator: that can estimate the latest data based on some inputs
2. Error Detector: that can compare the estimated value with actual sensor value
3. Sensor: that are available on the vehicle
4. Data Sender: that sends the collected data from the vehicle to the off-board system

Figure 6: The proposed architecture of event based on-board and off-board system.

The architecture of the on-board system contains all the four components as shown in Figure 6. The arrows in the diagram indicates the direction of data flow. In the on-board system, estimated value from the on-board estimator is compared with the actual sensor data. In case the estimated value varies from actual value obtained from sensor than a pre-set limit, the actual value is send to the off-board system. The off-board system also has an estimator which does the same calculation as the on-board estimator. The off-board system uses the estimator to estimate current value if no data is received from the on-board system.

5.1 Event-Based Architecture for Position Data

In order to limit the scope of the thesis, we consider only the position data. In the following, we explain in detail different architectural components with respect to the position data with the help of figure 7:

1. Estimator: The estimator uses a suitable dead reckoning algorithm to estimate the position of the vehicle using the previous position (previously estimated position or known position from GPS), velocity, heading and a digital map. The algorithm used in the estimator is explained in detail in section 5.2. The

Figure 7: The proposed architecture of the event based system for position data.

same algorithm is executed both in on-board and off-board device. Off-board estimator uses the last available value from the database to start the estimation.

2. **Error Detector:** The error detector compares the estimated value of position from the estimator with the actual value obtained from the GPS. If there is any deviation of estimated position more than the tolerance level then an event is triggered to send the actual value to the off-board system. The tolerance level is user defined. It can be the allowed distance between actual position and estimated position or the allowed heading difference between the actual and estimated value if both actual and estimated position is on the same road.
3. **Data Sender:** Data Sender sends the data to the datacenter in case an event occurs and actual position needs to be send to the off-board system.
4. **Position Database:** Position database stores all the position data received from various vehicles.
5. **Data Consuming Service:** There are different services which uses live data of the truck. One example is the fleet management portal where the customers can track live position of the vehicle. These services use the data provided by the off-board estimator.

5.2 Estimation Algorithm used in the Estimator

The off-board and the on-board estimators employ the same algorithm to calculate the estimated value, in this case, the position data. In the following, we discuss the algorithm that we have developed.

Our algorithm, called POS_ESTIMATE, estimates the position using the last transferred GPS position, speed and heading assuming the vehicle is on the road. The estimation of new position happens in five steps. The estimation begins using the initial position obtained from the vehicle when the vehicle starts. Estimation of latest position is done every second.

Algorithm 1: POS_ESTIMATE: The position estimator algorithm.

- 1: Compute the distance travelled in 1 second time using last known speed.
 - 2: Identify the direction of travel using last known heading.
 - 3: Identify the road on which the last known position is.
 - 4: Trace the calculated distance in the direction of travel on the same road.
 - 5: Identify the coordinates of the newly estimated position.
-

5.3 Delaying the off-board vehicle representation

One disadvantage with this algorithm is that at times the estimation can go wrong and the users may observe jumping of the vehicle to a new position on the map when the actual position is sent from the on-board system. If newly estimated point is on a different road than the actual vehicle (for example, while estimating at the intersections), or if the distance between the actual and estimated position is large, then this jump of vehicle is highly distinguishable. Delay is introduced to reduce the number of cases displaying incorrect routes to the user. The application used by the user executes the estimation algorithm after a certain amount of set delay when compared to the on-board system. However, the on-board system runs the estimation algorithm without any delay. Hence, whenever there is a deviation from the actual route of truck, the latest position is send to the off-board system. The client can use this newly received value which will be ahead of the displayed value on the client. Using this newly sent value from the on-board system, the client can follow the correct route without any estimation until the displayed position of vehicle reaches the newly received point. Actual value needs to be send to the off-board system from the vehicle only if the difference of current time and the time at which last position data was send is greater than or equal to the delay.

The delay can also reduce the number of position points send to the off-board system. In figure 8, let the actual position of vehicle at time t_1 be C. The off-board system will be “delay” seconds behind the actual position as shown in the figure, i.e., at position A such that $t_2 - t_0 = d$. Suppose, position data actPos1 containing details of position C is sent to the off-board system due to some deviation in estimated value from the actual value. For the next “delay” seconds, the off-board system can follow the route without any new estimation till it reaches point C. Now consider figure

Figure 8: On-board and off-board position at time t_2 . Figure 9: On-board and off-board position at time t_3 . Figure 10: On-board and off-board position at time t_4 .

9. Here, the estimated value deviates again from the actual value by not satisfying the required conditions, say at time t_3 , the latest position data actPos2 need not be send to the off-board system. New position data will be send to the off-board system only if the difference between the time at which actPos1 was sent and new position deviation occurred is greater than the delay. This is done to prevent extra messages from being send to the off-board system as the latter will reach actPos1 only after “delay” seconds. Hence, during this time it can follow the route based on actPos1 and prevent additional messages from being sent. In figure 10, the representation of vehicle in off-board system reaches actPos1. Hence, any deviation of estimated value after this point will be send to the off-board system.

5.4 Quality Attributes of the Software Architecture

In this section, we discuss the proposed architecture in terms of the quality attributes it addresses. Evaluating a software architecture doesn’t tell you if the prescribed architecture is good or bad. It highlights the areas where you are at a risk [11]. The design of the architecture in Figure 7 can be motivated with respect to the following quality attributes [11].

1. **Modifiability:** Modifiability is the ability to modify or alter the software, in response to the changes in the environment, requirements or functional specification, cost effectively and in less amount of time [23].

In order to achieve modifiability, the software has been split into a number of small modules each encapsulating certain functionality. Dependency among modules is less, thereby reducing coupling and increasing modifiability of the overall program. For example, change of estimation algorithm will only affect the estimator module.

2. **Performance:** Performance is measured in terms of the number of transactions processed in unit time or the time required to respond to a stimuli [11].

The proposed architecture improves network performance since the number of position messages send in 1 minute will decrease, enabling transmission of more messages. Moreover, the sent message will consists of only one position data and no position tails.

3. **Functionality:** Functionality is the ability of the system to complete the intended task [11].

In this system, the intended task is the availability of the latest position of the vehicle at the off-board system. Since the error detector always checks the estimated value with real value, and sends the real value in case the estimated value is wrong, off-board system will always have the correct position data. Hence, the functionality is not affected.

4. **Security:** Security is the measure of the ability of the system to deny access to unauthorized attempts of usage without affecting the service to legitimate users [11].

With the new design, we are not modifying any existing security features and hence, the security of the system remains unaffected. Encrypted data is sent from on-board to off-board system. Moreover, the estimator of the off-board system which provides the most accurate data is executed upon using the concerned applications by authorized users.

5. **Reliability:** Reliability is the ability of the software to continue with its intended functions without experiencing failures over a specified interval of time [24].

In our case, the estimator always ensures the availability of the approximate position data in the off-board system. Hence, even if the network is unavailable the approximate value of position data is available at the off-board system.

6 Simulator Implementation

In this section, we describe the simulator that was implemented to compare the proposed event based architecture with the interval based architecture that is being used at Scania currently. HereMap has been used for representing the data in simulator. Here is a company owned by Volkswagen, BMW and Daimler, and provides map related data and associated services to its customers [25]. Different APIs provided by HereMap have been used in the simulator for various purposes, for example determining the road on which vehicle is located, finding the point on same road after certain distance etc [26]. The program was written using typescript and C#.

6.1 Simulator Input file

The input file to the simulator consists of GPS data of a truck collected at an interval of 1 second for a duration of one hour. The file is in .txt format. This data is considered as the live GPS data during simulation. The file also contains the speed and heading of the truck at each point.

Input file was prepared from a file containing position coordinates for every second and speed at a resolution of 10ms. Data for every second was calculated by taking average of the speed for each 1 second interval. Heading was calculated based on the direction of travel at each point. For each of the point in the file, next point at a distance greater than 3 metres was identified and bearing was calculated using equation 1. The bearing between two points is the direction relative to North, which if followed from the start point in a straight line along a great-circle arc will reach at the end point [27]. The same value was used as heading if the calculated value was positive. In cases of negative values it was added with 360 degrees and used as heading.

$$\text{Bearing} = \text{atan2}(\sin\Delta\lambda \times \cos\phi_2, \cos\phi_1 \times \sin\phi_2 - \sin\phi_1 \times \cos\phi_2 \times \cos\Delta\lambda) \quad (1)$$

where ϕ_1, ϕ_2 are the latitudes of start and end points, λ_1, λ_2 are the longitudes of start and end points and $\Delta\lambda$ is the difference in longitudes [27].

6.2 Input Parameters

Experiment has been performed using different input parameters. Each of them is explained in table 1. The input parameters were set before executing simulations.

Name	Type	Description	Values Used
maxErrorDistance in metres	number	Represents the maximum allowed error distance between the actual position and estimated position in metres.	50, 75, 100, 125, 250, 500, 1000
delayOn	boolean	Represents if there is any delay between the actual position of the vehicle and their representation on the client. Delay can be set On or Off.	true, false
delay	number	If the delay is set as On, then the required amount of delay is set in seconds.	5, 10, 15, 20, 30, 45, 60
LatencyOn	boolean	Represents if latency has to be added for updating of position data at datacenter compared to the time at which it is created in the on-board system.	true, false
minimumSpeed in km/h	number	Represents the minimum speed below which the vehicle is considered to be stationary.	6
maxHeadDiff (differentRoad) in degrees	number	Represents the maximum allowed heading difference between the estimate and actual vehicle even if the both estimated position and actual position is on the same road.	10

maxHeadDiff (sameRoad) in degrees	number	Represents the maximum allowed heading difference between the estimated and actual value if both the estimated position and actual position is on different road.	120
consecutiveTriggers	number	Determines the number of times the error should occur consecutively before sending the actual position to the off-board system	3
streetMatchQualityThreshold	number	Represents the closeness of the location to the nearest road. Maximum value is 1.	0.65

Table 1: The simulator input parameters.

The values of input parameters maxErrorDistance, delayOn, delay and LatencyOn were varied to perform experiments and the obtained results were recorded in an Excel Sheet.

6.3 Estimator Algorithm

Using the first point from the file the simulator starts estimation. Based on the input position, it estimates the new position for each second and displays the data on a map. Input position can be last estimated position or data from file depending on if actual data has been send to off-board system or not. Implementation of the estimation is as follows:

1. **Compute the distance travelled in 1 second time using last known speed:** Distance travelled is calculated as the product of speed and time.

$$\text{Distance} = \text{Speed} \times \text{Time}$$

2. **Identify the direction of travel using last known heading:** Heading helps in determining the direction of travel. In HereMap, road is divided into many segments. Each segment has a number of points depending on its shape. Using the HereMapAPI, segmentID and segment points of the known position data is obtained. The point on the segment that is nearest to the current position is identified. Bearing between these two points is calculated using equation 1 and compared with the heading to determine the direction of travel.

3. **Identify the road on which the last known position is:** Segment id of current position is obtained using the API. As mentioned earlier, in HereMap roads are represented as collection of segments.
4. **Trace the calculated distance in the direction of travel on the same road:** Using the API it is possible to identify a point at certain distance and bearing. This is done by traversing along the segments in the direction of travel and finding the point that is at the calculated distance from the last known position.
5. **Identify the coordinates of the newly estimated position:** Latitude and longitude of the new position is obtained using the API.

These steps repeats using the newly estimated position in case no update happened from the on-board to off-board system. If the actual position was sent from on-board system, then both on-board and off-board systems uses this newly updated position for estimation.

6.4 Estimation validation

Each estimated value is verified against the location data of the truck in the input file to check for deviations. If the estimated position and actual position differs by a length greater than the allowed limit, then the actual current position, heading and speed is send to the off-board system. The newly sent actual position value is used for estimating next position. In addition to the check on the maximum deviation allowed, there are two other conditions that are checked to ensure that the estimated value is correct. If the road names of the estimated and actual position is different and their headings differ by an angle greater than 10 degrees, or if the road names are same and the headings differ by greater than 120 degrees, in both cases the estimation is considered to be incorrect and the actual GPS value from the vehicle is send to the off-board system. In order to cover up the limitations of HereMap API, the actual data is send only if the error happens consecutively a number of times. The number of times can be set by the user. We have performed the experiments setting the consecutive trigger as 3. Additionally, to cater the cases where the vehicle moves off the road, for example to parking lots, and prevent the estimation from continuing, we also check the closeness of the point from the road. This is done to avoid sending of actual position to off-board system when the vehicle is at parking lot. Moreover, if the speed of vehicle is less than 6km/h, then the vehicle is considered to be stationary.

6.5 User Interface

Figure 11 represents the user interface screen of the simulator. The green, blue and red circles with arrow represents the position of the vehicle as per the estimation

algorithm, interval based algorithm used at Scania and the actual position of the vehicle respectively. Whenever, an actual position is sent, then the point is marked on the map with the help of a green flag. The reason for sending the actual position is also displayed along with the flag. Different error messages and their reasons that can lead to sending of actual position is listed in table 2. The simulator also displays the points at which position data will be send to the off-board system in case of Scania's current algorithm with the help of blue flags. At the end of simulation, the simulator reports the average error distance, total number of incorrect routes and the average interval at which position data was send from the on-board to the off-board system. The latency for updating each position data is also displayed on the simulator screen.

The right bottom corner has two buttons namely "Hide marker text" and "Hide circle". "Hide marker text" can be turned on to hide all the text bubbles displaying error messages. Similarly "Hide circle" can be turned on to hide the green and blue circles surrounding the green and blue circles with arrow that represent the off-board vehicle position as per the estimation algorithm and interval based algorithm respectively. The green circle is drawn with latest estimated position as the centre and the maximum allowed error distance set by the user as the radius in cases with no delay. If a delay is set by the user, then the radius is calculated as the sum of the maximum allowed error distance and the distance that can be travelled with last known speed in delay seconds. Similarly, the blue circle is drawn with latest obtained position from the on-board system as the centre and the distance that can be travelled with the last known speed in time difference between the current time and time at which last position was transferred from the on-board system to the off-board system in case of interval based algorithm as the radius. The speed at which the simulator is executed can be adjusted on the arrows at the bottom of the screen. The calculations are speeded up by a factor selected by the user. There is a limit of increasing the speed as there are lot of asynchronous calls occurring during simulation and the simulator has to wait for the results of the asynchronous functions before continuing.

Error Message	Reason
ErrorDistance: Xm	If the distance between estimated position and the actual position is greater than the set limit. X represents the actual distance between the estimated and actual position of the vehicle in metres.
Leaving nonroad area.	This error is triggered whenever the vehicle leaves the parking lot or load-unload area. This error occurs when : the actual truck position $\text{streetMatchQuality} > \text{streetMatchQualityThreshold}$ and estimated truck position $\text{streetMatchQuality} < \text{streetMatchQualityThreshold}$.
Road and heading mismatch.	If the estimated and actual roads are different and their heading also differs more than the maxHeadDifference for 'different roads'.

Latency (s)	Percentage	Mean Value (s)
<1	(0-2.4)	0.5
1-2	[2.4-61.3)	1.5
2-3	[61.3-87.3)	2.5
3-4	[87.3-91.7)	3.5
4-5	[91.7-93.5)	4.5
5-10	[93.5-96.1)	7.5
10-15	[96.1-96.8)	12.5
15-20	[96.8-97.1)	17.5
20-30	[97.1-97.5)	25
30-60	[97.5-99.1)	45
60-180	[99.1-100)	120

Table 3: The latency values used in experiment.

Opposite direction.	If the estimated and actual roads are same but their heading differs more than the maxHeadDifference for 'same roads'
---------------------	---

Table 2: Error messages and reasons.

6.6 Latency for position updates

In this case latency is defined as the delay in receiving the data at the datacenter after sending the message from the on-board system due to network issues. In order to determine the latency, we extracted existing position data from Scania's database for last two years and compared their message creation time and database update time. The curve in figure 12 was obtained on plotting the data obtained from the database. We ignored values with latency greater than 3 minutes.

The value obtained from database can be grouped into the categories as shown in table 3. For each of the categories, their percentage contribution is given by the second column. In the simulator, a latency value is added to each of the position update by randomly selecting a value between 0 and 100 and picking up the corresponding latency value as specified in the table.

Figure 11: The simulator user interface

Figure 12: The latency distribution curve.

7 Experimental Results and Analysis

The proposed architecture and algorithm for estimation of position using previously known position, heading and speed is evaluated on the basis of 5 metrics described in section 7.1: Average off-board error distance, number of estimated incorrect routes, message size in bytes/minute, average message transfer frequency and total number of messages .

Results obtained from simulation for both interval-based algorithm and event-based algorithm are discussed here. The 1 minute interval values of the interval based architecture is the basis for comparison for all 5 metrics. This shown by a black horizontal line in the appropriate graphs. The experiments were divided into two sections: with latency and without latency. As explained previously, latency is the delay in receiving the data at the datacenter after sending the message from the on-board system due to network issues.

7.1 Experimental Goals

The experiments were performed by keeping the values of minimumSpeed, maxHeadDiff (differentRoad), maxHeadDiff (sameRoad), consecutiveTriggers and streetMatchQualityThreshold fixed. maxErrorDistance and delay were varied to evaluate its effects on the quality of estimation. Our main purpose behind the experiment is to compare and find out the feasibility and advantages of using an event based messaging system. The metrics for evaluation are:

1. **Average off-board error distance:** It is the average of the great circle dis-

tance between the actual position of the vehicle at that time and the position of the vehicle that is displayed on the client. In case of the time-based algorithm that is currently being used in Scania, average error distance is calculated between the actual position of vehicle and the last known position displayed at the off-board system. The distance between two points is measured using Haversine formula.

The Haversine formula is used for calculation great-circle distance between two points on the surface of a sphere given their latitudes and longitudes [27]. The great-circle distance or orthodromic distance can be defined as the shortest distance between two points on the surface of a sphere, measured along its surface.

$$a = \sin^2(\Delta\phi/2) + \cos\phi_1 \times \cos\phi_2 \times \sin^2(\Delta\lambda/2)$$

$$c = 2 \times \text{atan2}(\sqrt{a}, \sqrt{1-a})$$

$$\text{distance} = R \times c$$

where $\Delta\phi$ is the difference between latitudes of the two points in radians, $\Delta\lambda$ is the difference between longitudes of the two points in radians, R is earth's radius (mean radius = 6,371km).

2. **Number of estimated incorrect routes:** The number of times the client displays a position which is not the prospected route of the vehicle. It is the number of times a wrong route was selected by the estimator. In here map each road is divided into many segments. The number of incorrect routes is measured by counting the segments covered by the estimated point which are not traversed by the actual vehicle. This is a drawback with the event-based system as no incorrect route will be displayed in case of interval based system.
3. **Average message size (bytes)/minute:** Average size of the messages transferred in 1 minute in bytes.
4. **Average message transfer interval:** It is the average time difference between the messages containing position data that will be send from the on-board to the off-board system.
5. **Total number of messages:** It is the total number of messages sent from the on-board system to the off-board system.

The experiments were divided into two sets: with latency on and without any latency. Each set consisted of a number of simulation runs by varying the delay and maximum allowed error distance. Maximum allowed error distance is the allowed distance between the actual and estimated points in metres. The delay determines the time (seconds) for which the display on client is behind the actual position of the truck.

7.2 Experiments without Latency

The results of the experiments discussed in this section were conducted without considering the effects of latency.

7.2.1 Variation of Average Off-board Error Distance

In this section, we compare the average off-board error distance of interval-based architecture and the event-based architecture with and without delay.

Interval-Based: Here we explain the effects of changing message sending interval on accuracy in case of interval based architecture currently used at Scania. The intervals used at Scania for sending messages are 1 min, 5 min and 10 minute. In the simulation, we have obtained data for 1-10 minute range by incrementing the interval by a minute. This experiments were done assuming there is no time difference in sending data from on-board system and updating the data at the datacenter.

Figure 13 depicts the changes in the average error distance for different intervals in interval-based messaging system. As can be seen from the graph, average error distance increases with increase in the interval. The minimum error distance is for the 1 minute interval which is approximately 454 metres.

Figure 13: Average off-board error distance variation with changing interval in interval-based architecture.

Event-Based without delay: Now we will discuss the results of using an event-based system for updating location information. We assume that there is no latency for updating the position data in the datacenter and the live vehicle position is displayed on the client without any delay.

Maximum allowed error distance which is a user setting was varied and the differences in average off-board error distance was measured.

The average off-board error distance of the event based architecture shows an increase with the increase in maximum allowed error distance, as can be seen in

Figure 14: Average off-board error distance variation with changing maximum allowed error distance in event-based architecture.

figure 14. However, it can be seen that the average off-board error distance is much lower than the smallest average off-board error distance for the current interval-based algorithm used at Scania, which was approximately 454 metres (shown in figure 13 for the 1 minute interval).

Event-Based with delay: As discussed in section 5.3, the purpose of introducing delay was to reduce the number of incorrect routes displayed on the client so that the most accurate route followed by the vehicle is presented to the user. Introducing delay was also expected to reduce the number of messages with actual position data transferred from the vehicle to the datacenter.

The experiments were performed for different combinations of maximum allowed error distance and delay in order to analyze the variations in average off-board error distance. Here, we compare the minimum average off-board error distance of interval based architecture with those of event-based architecture (with and without delay).

Figure 15 shows different values of average off-board error distance for different combinations of delay and maximum allowed error distance, both set by the user. As can be observed from the graphs, for each of the maximum allowed error distances, average off-board error distances increases with increase in delay. However, for higher values of delay like 45 and 60 seconds, the maximum allowed error distance have negligible effect on the average off-board error distance. The average off-board error distance values are similar for higher values of delay. The black line represents the average off-board error distance value for the interval based algorithm. The average error distance is much below the black line for most of the combinations

Figure 15: Average off-board error distance variation with changing maximum allowed error distance and delay in event-based architecture.

except for high values of delay like 30, 45 and 60 seconds.

7.2.2 Variation of Number of Estimated Incorrect Routes

One drawback with the estimation is the display of wrong routes to the user if the estimation goes wrong. We would like to remind the reader that the number of estimated incorrect routes were measured by counting the number of times the estimated point falls on the road segments that were not covered by the actual vehicle. In the following sections, we discuss how incorrect routes are affected by variation of maximum allowed error distance and delays.

Interval-Based: Since the interval based architecture only sends the actual position of the vehicle and do not perform any estimation, no incorrect routes will be generated.

Event-Based without delay: Different values of maximum allowed error distance were used as the input parameter to understand the effect on the incorrect routes displayed. This does not show a uniform pattern as the incorrect route depends on the nature of the road and points at which position data is obtained from the vehicle. Chances of displaying an incorrect route is more in case of roundabouts, forks, turns etc. Hence, the variation in number of displayed incorrect route as shown in figure 16 is not predictable.

Figure 16: Number of estimated incorrect routes variation with changing maximum allowed error distance in event-based architecture.

Figure 17: Number of estimated incorrect routes variation with changing maximum allowed error distance and delay in event-based architecture.

Event-Based with delay: Here, we compare the variation of the number of estimated incorrect routes generated on adding and varying delay with that of the no delay case. Total number of estimated incorrect routes in the off-board system reduced considerably with the addition of delay compared to the experiment without inducing delay. This is depicted in figure 17. A particular pattern with the reduction of number of incorrect routes cannot be deduced from the figure as it depends to some extent on the nature of route and do not follow a uniform pattern as mentioned in previous section.

7.2.3 Variation of Average Message Size (Bytes)/minute

The message size in bytes was calculated considering the number of position heads, position tails, bytes for encryption and other additional data included in the message. In order to calculate the average message size, the cumulative size of the messages in bytes for the 1 hour journey was calculated. Then the obtained total was divided by 60 to calculate the average message size in bytes/minute.

Interval-Based: Figure 18 shows the variation of average message size in bytes/minute for different message sending intervals. This also includes the size of the position tails send along with the position heads in each message. Average message size per minute also decreases with increase in interval as the number of messages decrease with increase in interval.

Figure 18: Average message size in bytes/min variation with changing interval in interval-based architecture.

Event-Based without delay: With the increase in maximum allowed error distance, the number of messages decreases resulting in the decrease of average message size sent per minute. This is shown in figure 19. For maximum allowed error distances of 100m and above the average message size in bytes per minute is lower than 46 bytes/minute (for interval based architecture). Moreover, the associated average off-board error distance for these simulation runs is much lower than the interval based approach with interval as 1 minute (figure 14). In short, the event based approach reduces the message size while improving the accuracy.

Figure 19: Average message size in bytes/minute variation with changing maximum allowed error distance in event-based architecture.

Event-Based with delay: In figure 20 the black line represents the average message size in bytes/minute for the interval based architecture. The average message size in bytes/min in case of event based system exceeded the interval based value only in 5 cases out of the total 56 cases. With the increase in maximum allowed error distance, the average message size in bytes/minute decreases. Moreover, by adding delay, number of messages sent decreased resulting in decrease of average message size/minute.

7.2.4 Variation of Average Message Transfer Interval

The message transfer interval is fixed in case of interval based architecture. However, in case of event based architecture it was calculated as 3600 seconds divided by total number of messages sent converted to 00:00:00 format.

Figure 20: Average message size variation with changing maximum allowed error distance and delay in event-based architecture.

Interval-Based: The 1 minute interval of the interval based architecture is considered for comparing with the event based architecture.

Event-Based without delay: The rate at which messages with actual position data is sent to the off-board system increases with increase in maximum allowed error distance. This variation is shown in figure 21. For a maximum allowed error distance of 50m, 75m, 100m and 125m more messages are required than the interval based system. The interval based messaging system with an interval of 1 minute sends 60 position messages from on-board to off-board system. The average message transfer interval for the interval based architecture is 00:01:00. For higher values of maximum allowed error distances 250m, 500m and 1000m lesser messages are sent compared to the interval based algorithm with much higher accuracy.

Event-Based with delay: A comparison of the average message transfer interval is given in the figure 22. Black line representing interval based algorithm with an interval of 1 minute is considered as the basis for comparison. Higher the transfer rate, better is the performance of the event based system. For higher values of delay and maximum allowed error distance, lower is the average transfer rate and is lower than 1 minute which is the case of interval based system. By increasing the values of delay or maximum allowed error distance, the average message transfer interval increases. For a maximum allowed error distance of 25m and above all of the simulation runs gave an average message transfer interval greater than 1 minute. Similar is the case with simulation runs having delay greater than 30.

Figure 21: Average message transfer interval variation with changing maximum allowed error distance in event-based architecture.

Figure 22: Average message transfer interval variation with changing maximum allowed error distance and delay in event-based algorithm.

7.2.5 Variation of Total Number of Messages

In this section, we compare the total number of messages that were sent from the on-board system to the off-board system.

Interval-Based: Figure 23 represents the variations in the number of messages sent from the on-board to off-board system in interval based architecture. Each message can contain a position head and a number of position tails. The number of messages sent to the off-board system decreases exponentially with increase in interval.

Figure 23: Number of messages sent with changing interval in interval-based architecture.

Event-Based without delay: The number of messages sent to the off-board system decreases with increase in the maximum allowed error distance set as simulator input. 60 messages are sent in case of interval based architecture for the 1 hour journey. From the figure, it can be seen that for maximum allowed error distance less than or equal to 125 metres, the number of messages were more than 60. The number of messages is below 60 for 250, 500 and 1000 metres of maximum allowed error distance.

Event-Based with delay: Figure 25 shows the actual number of messages containing position data transferred in various simulation runs. Total number of messages transferred in case of interval based system was 60 represented by the black line. The number of messages transferred decreases with increase in delay or increase in maximum allowed error distance.

Figure 24: Total number of messages with changing maximum allowed error distance and delay in event-based architecture with latency.

Figure 25: Number of messages sent with changing maximum allowed error distance and delay in event-based architecture.

7.3 Experiments with Latency

All of the above experiments were repeated by manually adding latency to each of the position messages. In order to understand the latency pattern we analyzed existing data from the database and their latencies. The latency values were selected from the table 3 by randomly generating a number within 0 to 100 and determining the probability category to which it falls (from column 2 of the table). Based on the probability, the values in column “Mean Value” were selected as the latency for that particular position message.

7.3.1 Variation of Average Off-board Error distance

In this section, we repeat the experiments in section 7.2.1 by including latency. Average off-board error distance of the interval based architecture with latency is compared with the event-based architecture with latency.

Interval-Based: The results of the experiment conducted for the interval-based algorithm including latency are similar to the case without latency.

However, a slight increase can be observed in the average off-board error distance compared to the case without latency. This increase in error distance is due to the delay in updating the latest value at datacenter. As a result of which the client continues to display the previous position until the data is updated. Average error distance for various intervals is displayed on figure 26. The shape of the curve is similar to the case without latency with an elevated value of average off-board error distance for each interval.

Event-Based without delay: Even after including latency the average error distance is much lesser than the interval based case. The maximum average off-board error distance is for the simulation run with maximum allowed error distance 1000 metres, i.e., approx. 194m. This is considerably low compared to the minimum value in interval-based system with latency which is around 574 metres. As can be seen in figure 27, the average off-board error distance doesn’t show a uniform increase as the latency values are randomly induced. Average error distance increases if the values of latency is higher.

Event-Based with delay: All the simulation runs were re-executed by adding latency for storing the messages in the datacenter and displaying the data on the client with different delays. Since the latency values are randomly induced, the variations of different metrics is not expected to show a uniform pattern.

Figure 28 shows the variation of average error distance. Average error distance is slightly higher than the case with no latency due to the delayed update of data. Average error distance increases with increase in maximum allowed error distance and delay. For higher values of delay, the effect of maximum allowed error distance is negligible. The average error distance value is similar for delays greater than 20 seconds. In the figure, black line indicates the average error distance value for the interval based architecture including latency. For simulation runs with delay less

Figure 26: Average error distance variation with changing interval in interval-based architecture with latency.

Figure 27: Average error distance variation with changing maximum allowed error distance in event-based architecture with out delay and with latency.

than or equal to 30 seconds, the average error distance of event based architecture

Figure 28: Average off-board error distance variation with changing maximum allowed error distance and delay in event-based algorithm with latency.

is lesser than the interval-based architecture.

7.3.2 Variation of Number of Estimated Incorrect Routes

Changes in the number of estimated incorrect routes after including latency are discussed in this section.

Interval-Based: As mentioned previously, no incorrect routes are produced in case of interval based architecture as there is no estimation.

Event-Based without delay: The number of incorrect routes is more with latency than the no latency case as the updated information reaches the datacenter after certain delay. Hence, more incorrect routes are taken by the estimator. The graph depicting variation in the number of incorrect routes is shown in figure 29.

Event-Based with delay: We compare the incorrect estimated routes generated with no delay with that generated by the simulation runs after adding different values of delay. Incorrect estimated routes decreased on adding delay compared to the simulation runs with no delay. However, the effect of latency considerably increased the incorrect routes. This is because of the delay in receiving the actual data in cases when a message is sent from the vehicle due to latency. The differences in the number of incorrect routes for different delays and maximum allowed error distances compared to the case with no latency is shown in figure 30 in red colour.

Figure 29: Number of estimated incorrect routes variation with changing maximum allowed error distance in event-based architecture with out delay and with latency.

Figure 30: The difference in number of incorrect routes with changing maximum allowed error distance and delay in event-based algorithm with latency.

Figure 31: Average message size in bytes/minute variation with changing maximum allowed error distance in event-based architecture with out delay and with latency.

7.3.3 Variation of Average Message Size (Bytes)/minute

Average bytes/minute used for sending messages from on-board to off-board system is compared between interval based and event based architecture by including the effects of latency.

Interval-Based: The average message size/minute is same as the case without latency. The graph is same as figure 18. This is because the number of messages sent is not varying.

Event-Based without delay: Variation in the message size/minute depends on the number of messages sent. Increase in maximum allowed error distance results in reducing the number of messages and hence the average message size per minute as in figure 31.

Event-Based with delay: With the addition of delay, the number of messages and hence the average message size/minute decreases gradually. However, figure 32 does not show any uniform pattern in the decrease as the latency values are randomly induced and is different for each simulation run.

7.3.4 Variation of Average Message Transfer Interval

Changes in the average message transfer interval after including latency is discussed in this section.

Figure 32: Average message size in bytes/minute variation with changing maximum allowed error distance and delay in event-based algorithm with latency.

Interval-Based: Average message transfer interval is constant for interval based architecture and we consider 1 minute interval as the basis for comparison.

Event-Based without delay: Figure 33 shows a uniform increase in average message transfer interval with the increase in maximum allowed error distance. Higher values of maximum allowed error distance have messages sent at interval greater than one. However, this will lead to reduced accuracy of data available at the off-board system.

Event-Based with delay: Average message transfer interval increases on adding delay as the number of messages decreases. The black line represents the 1 minute interval of interval based architecture. The average message transfer interval goes above 1 minute for higher values of delay or for higher values of maximum allowed error distance.

7.3.5 Variation of Total Number of Messages

Here, we discuss the variation in the number of messages sent from on-board to off-board system both incase of interval based and event based architecture after including latency.

Interval-Based: The number of messages sent and the average message size per minute did not change with latency. The graph is same as figure 23.

Figure 33: Average message transfer interval variation with changing maximum allowed error distance in event-based architecture with out delay and with latency.

Figure 34: Average message transfer interval variation with changing maximum allowed error distance and delay in event-based architecture with latency.

Figure 35: Total number of messages sent with changing interval in interval-based architecture with latency.

Event-Based without delay: As can be seen from the figure 36 number of messages decreases with increase in maximum allowed error distance.

Event-Based with delay: Number of messages reduces on adding delay. However, in the figure 37, reduction of messages on adding delay is not consistent. This is due to the effect of latency which is different for different runs.

7.4 Discussion

Adoption of event based architecture can improve the accuracy of data available at the off-board system. Accuracy of data is inversely proportional to data reduction in case of event based architecture. Suitable values of maximum allowed error distance and delay has to be set depending on the accuracy and data reduction that is to be achieved. The total number of messages directly affect the message size in bytes/minute and the average message transfer interval. By sending approximately 46 bytes/ minute, average off-board error distance obtained in case of interval based architecture is approximately 522 metres. On the other hand, under the event based approach, the average off-board error distance is reduced by 92% (maximum allowed error distance 100m) even though only 42 bytes/minute is sent. The major disadvantage with the estimation is the incorrectly estimated routes which is not present in the case of the interval based architecture. These incorrect estimations can be controlled to some extent by adding delay. With the addition of delay, we were able to further control the number of messages sent. However, delay affects the accuracy of data available at off-board system. Using event-based architecture we

Figure 36: Total number of messages with changing maximum allowed error distance in event-based architecture with latency.

Figure 37: Total number of messages with changing maximum allowed error distance and delay in event-based architecture with latency.

Figure 38: Relative average off-board error distance vs relative data consumption

were able to obtain an average off-board error distance of around 505m by sending 29bytes/minute. In case of interval based architecture, around 46 bytes had to be sent to obtain a similar average-off-board error distance. Hence, similar accuracy was achieved in event based architecture by using only 63% of the data.

Figure 38 shows the relative average off-board error distance plotted against relative data consumption for each of the simulation runs with latency. The average off-board error distance and average message size/minute of interval based architecture with 1 minute interval represents 100 percent of x and y axis respectively. The values obtained from different simulation runs for the event based architecture are represented by different dots. Dots within the red box are the ones having lesser average off-board error distance, i.e., better accuracy, and lesser data consumption compared to the interval based architecture. Most of the values with delay lesser than 45 seconds falls within the box. Therefore, delays of up to 30 seconds are better, on an average, for event based architectures.

We investigated the average message size/minute under event based and interval based approaches for the same average off-board error distance. We varied the maximum allowed error distance and delays of the simulator in order to achieve an average off-board error distance similar to that of the interval based architecture for intervals from 1 to 5 minute. This was obtained for the cases in which the maximum allowed error distance was fixed as 500m and for delay values 30, 60, 80, 120 and 160 seconds. The average message size/minute required to achieve similar accuracy in interval based and event based approach is compared in figure 39. From the figure, it is clear that the event based approach can provide the same level of accuracy with lesser data consumption. The average message transfer interval for each of the

Figure 39: Comparison of the average message size/minute in interval based and event based approaches

above discussed cases is also shown in the figure.

8 Conclusions

In this thesis, we explored the possibility of using an event-based architecture for Internet of Vehicles' messaging platform. The proposed event-based architecture consists of an on-board and off-board estimator executing the same estimation algorithm. The estimated value is verified against the actual sensor value in the on-board system. An event is triggered whenever the estimated value deviates more than a threshold, and eventually the actual sensor data is sent to the off-board system.

We then evaluated the proposed architecture in the specific context of position data, which mainly consists of location coordinates, speed and heading. A map based dead-reckoning algorithm was used for estimating the position, with the assumption that vehicle is always on the road.

The proposed event based architecture together with the dead reckoning algorithm to estimate the position was found to significantly improve the accuracy of the data available for the off-board applications. However, high number of incorrect routes were generated as a result of some estimations being incorrect. Lower values of maximum allowed error distance for the event based approach gave higher accuracy when compared to the interval based approach. Moreover, the average interval between message transmissions were lower than the interval based approach. Application of delay on the data at the off-board system increased the average interval between message transfer. However, it had a contradicting effect on the accuracy and increased the average error distance. The selection of maximum allowed error distance and delay depends on the requirement of the user. If reduction of messages is important for the user, then higher values of maximum allowed error distance or delay needs to be selected. On the other hand, if availability of data with high accuracy is the primary aim of the user then the maximum allowed error distance and delay has to be kept as low as possible.

8.1 Future work

Accuracy of estimation can be further enhanced by adding more details into the position message in addition to the speed and heading. For example, knowledge of odometer value (distance travelled by the vehicle) can help in estimating the future position of the vehicle more precisely.

As can be seen from figure 40, estimation of the new position is fairly accurate on highways. The messages sent to the off-board system due to inaccurate estimation represented by green flags are very few. The number of messages sent is considerably low in comparison with the messages sent in interval based system represented by blue flags. This is because the speed of the vehicle is relatively steady compared to other roads and chances of changing roads is very less. However, in case of road changes or roundabouts high number of messages are sent as the chances of estimation going wrong is high. This is shown in figure 41. If the estimator has access to details of final destination of the vehicle, the probability of an incorrect estimation is less. The most probable route that the vehicle takes can be determined with the help of routing APIs.

Moreover, we have performed experiments only for the position data. Future

Figure 40: Messages sent while the vehicle is on the highway.

Figure 41: Messages sent while the vehicle is on roundabout.

work also includes extending the architecture to include other kinds of data and designing estimation algorithm for them.

8.2 Threats to validity

Due to time constraints, we were not able to implement and test the event based system architecture and estimation algorithm on the actual system used by Scania. Implementing the proposed set up on the real system and its evaluation will provide more reliable results and can open up a set of challenges which can be experienced only in real environment.

The conclusions were made with the evaluation of just one vehicle data for a duration of one hour. Performing same set of experiments for different set of data can provide a better generalized view of the results. Moreover, the validity of the results to a great extent depends on the accuracy of GPS data. Using more accurate GPS like differential GPS would provide more accurate results [28]. For preparing the input file for estimation, headings were calculated based on formulas and not obtained from the GPS. Hence, the calculated headings will be not accurate and can have impact on the quality of results.

References

- [1] C. Bergenheim, S. Shladover, E. Coelingh, C. Englund, and S. Tsugawa, “Overview of platooning systems,” in *Proceedings of the 19th ITS World Congress, Oct 22-26, Vienna, Austria (2012)*, 2012.
- [2] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark, “Connected vehicles: Solutions and challenges,” *IEEE internet of things journal*, vol. 1, no. 4, pp. 289–299, 2014.
- [3] K. H. Johansson, M. Törngren, and L. Nielsen, “Vehicle applications of controller area network,” in *Handbook of networked and embedded control systems*. Springer, 2005, pp. 741–765.
- [4] ISO/DIS 20077-1:2017, *Road Vehicles Extended vehicle (ExVe) methodology*. ISO, Geneva, Switzerland, 2017.
- [5] (2017) Scania in brief. [Online]. Available: <https://www.scania.com/group/en/section/about-scania/scania-in-brief/>
- [6] R. Van Kranenburg, *The Internet of Things: A critique of ambient technology and the all-seeing network of RFID*. Institute of Network Cultures, 2008.
- [7] L. D. Xu, W. He, and S. Li, “Internet of things in industries: A survey,” *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2233–2243, Nov 2014.
- [8] M. S. Mahmoud and M. Sabih, “Networked event-triggered control: an introduction and research trends,” *International Journal of General Systems*, vol. 43, no. 8, pp. 810–827, 2014.
- [9] M. K. Chandy, “Event-driven applications: Costs, benefits and design approaches,” 2006.
- [10] B. M. Michelson, “Event-driven architecture overview,” *Patricia Seybold Group*, vol. 2, 2006.
- [11] L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice*, 3rd ed. Addison-Wesley Professional, 2012.
- [12] P. Kolios, C. Panayiotou, G. Ellinas, and M. Polycarpou, “Data-driven event triggering for iot applications,” *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1146–1158, Dec 2016.
- [13] (2017) Current gps constellation. [Online]. Available: <http://tycho.usno.navy.mil/gpscurr.html>
- [14] (2017) Understanding the global positioning system (gps). [Online]. Available: <http://www.montana.edu/gps/understd.html>

- [15] M. Shaw, "The coming-of-age of software architecture research," in *Proceedings of the 23rd International Conference on Software Engineering*, ser. ICSE '01. Washington, DC, USA: IEEE Computer Society, 2001.
- [16] G. Dodig-Crnkovic, "Scientific methods in computer science," in *Proceedings of the Conference for the Promotion of Research in IT at New Universities and at University Colleges in Sweden, Skövde, Suecia*, 2002, pp. 126–130.
- [17] M. Gerla, E.-K. Lee, G. Pau, and U. Lee, "Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds," in *Internet of Things (WF-IoT), 2014 IEEE World Forum on*. IEEE, 2014, pp. 241–246.
- [18] X. Li and Q. Xu, "A reliable fusion positioning strategy for land vehicles in gps-denied environments based on low-cost sensors," *IEEE Transactions on Industrial Electronics*, vol. PP, no. 99, pp. 1–1, 2016.
- [19] Y. S. Chiou, F. Tsai, and S. C. Yeh, "A dead-reckoning positioning scheme using inertial sensing for location estimation," in *2014 Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, Aug 2014, pp. 377–380.
- [20] P. Davidson, J. Collin, J. Raquet, and J. Takala, "Application of particle filters for vehicle positioning using road maps," in *23rd International Technical Meeting of the Satellite Division of The Institute of Navigation, Portland, OR*, 2010, pp. 1653–1661.
- [21] P. Hall, "A bayesian approach to map-aided vehicle positioning," Master's thesis, Linköping University, Sweden, 2001.
- [22] A. Leonhardi, C. Nicu, and K. Rothemel, "A map-based dead-reckoning protocol for updating location information," in *Proceedings 16th International Parallel and Distributed Processing Symposium*, April 2002, pp. 8 pp–.
- [23] P. Bengtsson, N. Lassing, J. Bosch, and H. v. Vliet, *Analyzing software architectures for modifiability*, 2000.
- [24] "Ieee standard glossary of software engineering terminology," *ANSI/ IEEE Std 729-1983*, pp. 1–40, Feb 1983.
- [25] (2017) Here (company). [Online]. Available: [https://en.wikipedia.org/wiki/Here_\(company\)](https://en.wikipedia.org/wiki/Here_(company))
- [26] (2017) Maps for developers. [Online]. Available: <https://developer.here.com/>
- [27] C. Veness, "Calculate distance and bearing between two latitude/longitude points using haversine formula in javascript," *Movable Type Scripts*, 2011.
- [28] B. W. Parkinson and P. K. Enge, "Differential gps," *Global Positioning System: Theory and applications.*, vol. 2, pp. 3–50, 1996.