Proposal of Traffic Database Management System

Yoshitoshi Murata
Faculty of Software and Information Science
Iwate Prefectural University
Takizawa, Japan
e-mail: y-murata@iwate-pu.ac.jp,

Abstract— **Cooperative intelligent transport systems (ITSs) are attracting much research to sustainably improve traffic environments. The European Telecommunications Standards Institute proposed and standardized the Local Dynamic Map (LDM), which is a conceptual data store embedded in an ITS station. It contains topographical, positional, and status information related to ITS stations within a geographic area surrounding the host station. The number of mobile objects such as pedestrians and automobiles is very big. They often travel around wide areas and sometimes travel across countries. These mean that a distributed real-time database system is needed. This paper presents a "Traffic Database Management System" (TDMS) for managing information related to mobile objects along roads in real time in the world. TDMS performs the same role as LDM but is aimed at the entire world, not a local area. Specifically, the TDMS structures and commands are introduced.**

Keywords- ITS; mobile object; traffic; stream database.

I. Introduction

Managing location and additional information of mobile object, such as pedestrians and vehicles in an integrated way are very useful for avoiding traffic accidents, reducing energy consumption, analyzing international logistics, etc. The number of mobile objects is too big and its coverage area is too wide to manage their location and additional information with a single server.

Cooperative intelligent transport systems (ITSs) are attracting much research interest as a means to sustainably improve traffic environments, such as by providing driving assistance and navigation information. The European Telecommunications Standards Institute (ETSI) proposed and standardized the Local Dynamic Map (LDM) as a cooperative ITS. LDM is a conceptual data store embedded in an ITS station in vehicles, roadside facilities, and ITS centers. It contains topographical, positional, and status information related to ITS stations [1][2]. The coverage area for each LDM server is limited to a surrounding each host station. Therefore, it is difficult for mobile objects to get information from other LDM servers.

The previously proposed concept of a "Cyber Parallel Traffic World" (CPTW) cloud service is aimed to practice driving in destinations before the trip or drive sightseeing virtually at the coming 5G and connected car era [3]. In CPTW, roads, sidewalks, and traffic facilities, such as traffic signals, are basically represented as they are in the real world. Vehicles, pedestrians, and temporary obstacles move synchronously with their real-world counterparts. CPTW is thus a virtual world synchronized with the real world. Its attention area is not local but worldwide. A previous paper introduced the CPTW concept and the technologies used to create 3D polygons of roads and traffic signals from a road database and a rule database. To achieve CPTW, however, a database is also needed for managing information about mobile objects along roads in real time.

This paper presents the "Traffic Database Management System" (TDMS). Its basic role corresponds to that of LDM for type 4 data (highly dynamic). It differs from LDM in that it manages data from all over the world rather than only local data. Although TDMS handles streaming data the same as sensor networks, it is actually a big data system given that the number of traffic-related mobile objects in the world is tremendous. This means that a distributed database system is needed. Many types of distributed database systems have been developed to deal with big data. Google developed the Google File System [4], Bigtable [5], and MapReduce [6] to deal with the large and ever-increasing amount of Web data. In Google's distributed database system, big Web data are assigned to each distributed server. However, vehicles often travel around wide areas and sometimes travel across countries. Therefore, generated stream data should be uploaded real-timely to the server that manages the area where the data was generated. Distributing the stream data for mobile objects to area servers creates several kinds of problems. For example, mobile objects near area boundaries need information stored in neighbor area servers, which is difficult to do. The proposed system would solve such problems.

In this paper, the concept of TDMS, requirements, database structure, and commands are introduced. The nodes in the system are vehicles and pedestrians. The database stores information about their locations, speed, direction, etc., enabling drivers to obtain information about the mobile objects in their vicinity. Obviously, if drivers are continually accessing the database to obtain this information, the access load will be tremendous. Therefore, the data upload and download speeds must be significantly higher than those of existing sensor network systems.

Prior to a discussion of related work in Section III, CPTW and TDMS concepts are reviewed and introduced in

Section II. The system design, including the database schemes and commands, are introduced in Section IV. The key points are summarized and future work is mentioned in Section V.

II. CPTW AND TDMS CONCEPTS

Before explaining TDMS concept, a review of the concept of CPTW in which TDMS resides will be presented. In the CPTW, roads, sidewalks, and traffic facilities, such as traffic signals are basically represented as they are in the real world. Vehicles, pedestrians, and temporary obstacles move synchronously with their real-world counterparts. Virtual vehicles can be driven in the CPTW in the same way that actual vehicles can be driven in the real world. Unlike in the real world, vehicles and pedestrians can communicate with nearby vehicles and pedestrians by pointing not at their ID (address) but at their position. In this virtual world, it would be possible to detect whether a vehicle or pedestrian obeyed the local traffic rules.

The virtual structures are constructed by extracting road data from maps covering the world and by gathering traffic rules for each locality. These data are stored in a roads database and a rules database. The real-time data for mobile objects, such as the locations of vehicles and pedestrians. are stored and managed in TDMS. An application program creates 3D roads using data in the roads database, creates traffic signals and traffic signs from the rules database, and plots mobile objects, as shown in Figure 1. CPTW is not only useful to experience and practice driving in foreign countries but also to drive sightseeing.

The concept of TDMS is very simple: a stream database is used to store and manage data for all traffic-related mobile objects worldwide, as shown in Figure 2. TDMS would be implemented in connected vehicles and pedestrians since most traffic-related mobile objects are expected to be connected to the Internet through 5G mobile networks.

Archived data as well as real-time data would be stored in each TDMS server. There are two key types of terminals to access TDMS:

(1) A navigation terminal (client sensor device) in each traffic-related mobile object for sending and receiving data records containing ID, Type of object, Time, Location, Speed, Direction, etc. to and from traffic-related mobile objects in their vicinity and for displaying the information received in real time. Data records are selectable by attributing profiles.

(2) A PC that can gather the data for a designated mobile object, and data record contents are selectable. This means that many kinds of services using such data would be provided. For example, some service would extract ego-vehicles, and compulsory confiscate driving a vehicle, or alert vehicles to existence of ego-vehicles. Another one would analyze international logistics.

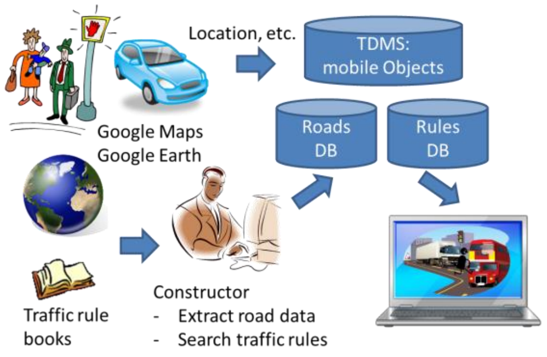
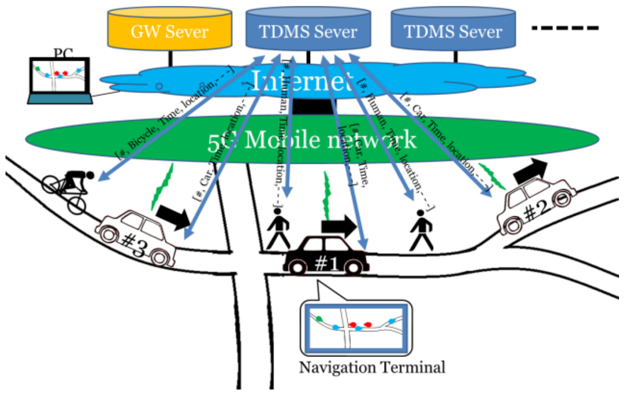


Figure 1. Overview of constructing CPTW


GW Sever
TDMS Sever
TDMS Sever
PC
Internet
5G Mobile network
[#, Bicycle, Time, location,- - -]
[#, Car, Time, location,- - -]
[#, Human, Time, location,- - -]
[#, Car, Time, location,- - -]
[#, Human, Time, location,- - -]
[#, Car, Time, location,- - -]
#3
#1
#2
Navigation Terminal
Figure 2. Image of TDMS (not in 3D)

III. RELATED WORK

The worldwide aspect of TDMS makes it unique. The closest related work is that on LDM type of ITS. In an LDM system, data related to traffic are exchanged between ITS stations through vehicle to vehicle (V2V) or vehicle to infrastructure (V2I) communication and stored in a database at each ITS station. An LDM system is a distributed database system without a central server.

Data related to traffic objects can be categorized into four types:

- Type 1—permanent static data, usually provided by a map data supplier,
- Type 2—transient static data obtained during operation, e.g., changed static speed limits,
- Type 3—transient dynamic data, e.g., weather situation, traffic information,
- Type 4—highly dynamic data, e.g., location of a moving vehicle.

The database tables for each type are related using location information. The roads and rules databases in the CPTW correspond to types 1 and 2. TDMS corresponds to type 4.

LDM systems have been implemented in several ways. NAVTEQ-LDM [7], for example, was implemented using SQLite, which is a relational database management system (RDMS) specialized for light-weight machines, such as smartphones and in-vehicle navigation terminals. The Bosch

Tele Atlas PG-LDM implementation is based on PostgreSQL [7], an open-source RDMS specialized for extensibility and standards compliance. In these systems, data, including highly dynamic data, are stored as RDMS records. Another approach was used by Sato et al. who implemented a LDM as a stream database ("SLDM") [10]. Highly dynamic data are input as stream data and joined with static and/or transient dynamic data stored in tables as RDMS records.

Each LDM in an ITS station covers the local vicinity. In contrast, TDMS is aimed at worldwide coverage, so the LDM approach cannot be applied to TDMS.

IV. SYSTEM DESIGN

In this section, requirements for the TDMS and a system configuration, database structure, and database commands to realize the requirements are introduced.

A. *Requirements and system configuration*

The number of traffic-related mobile objects, such as pedestrians and vehicles, is very big. These objects are distributed worldwide, and some of them, such as vehicles, move quickly and widely. TDMS requires movement information on each mobile object in real time to prevent traffic accidents.

The system must therefore be able to

(1) process big data distributed worldwide,
(2) gather the real-time locations of all traffic-related mobile objects,
(3) provide the location and type of all mobile objects in the vicinity of each mobile object, and
(4) provide a PC with selected data for mobile objects in accordance with the command selected.

To meet these requirements, TDMS must be a distributed database system comprising a gateway (GW) server, area servers, and navigation terminals, as shown in Figure 3.

Each area server needs four tables:

- a road table containing the identification of each road,
- a road reference table containing the relationships among roads crossing an area boundary line,
- an intersection table containing the identification of each intersection, and
- a mobile object (MO) table containing the information needed to estimate the real-time location of each traffic-related mobile object.

Information on the road on which each mobile object is moving is useful for narrowing down the objects that could potentially lead to an accident. The road estimation function in each area server automatically re-defines a road on the basis of global navigation satellite system location data and registers it in the road database. If it detects several mobile objects traveling along a new route, it automatically defines the route as a road. If it does not detect any mobile objects traveling along a road for a certain period of time, it automatically removes the road from the road database. A road is defined on the basis of the intersections, as described in the next sub-section. An intersection is defined on the basis of mobile object movements.

As described in Sub-section C, the road reference database is needed to manage the relationships among roads in contiguous areas.

Each mobile object needs a subset of the road table containing information for other mobile objects in the vicinity to enable its navigation terminal to recognize the road on which it is traveling and to access data for other mobile objects traveling nearby. This subset is similar to an LDM system. The road table, road reference table, and intersection table are implemented in an RDMS because their data do not need to be accessed quickly. In contrast, the MO table is implemented in RAM as a primary memory FIFO buffer because its data must be accessed quickly.

The database structure, design issues, and control schemes are described in the following sub-section.

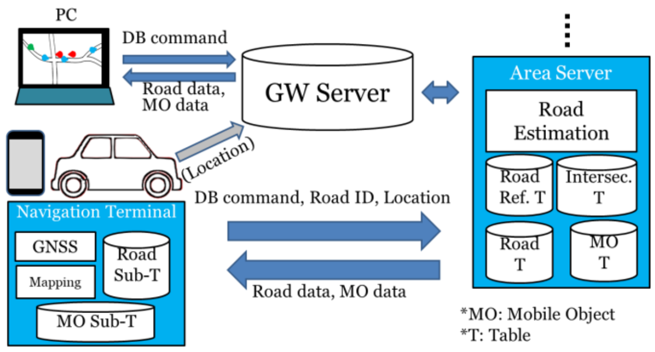


Figure 3. Configuration of TDMS

B. *Database structure*

Management of related information in the database system is described here.

1) *Location format*

A scheme in which each area is defined using latitude and longitude is not suitable because it is difficult to represent areas in which a mobile object moves by using two-dimensional values. Instead, the Military Grid Reference System (MGRS) [9] is used. It is the geocoordinate standard used by NATO militaries for locating points on the earth. MGRS code comprises a grid zone, a 100,000-m square identifier, and easting/northing codes, as shown in Figure 4.

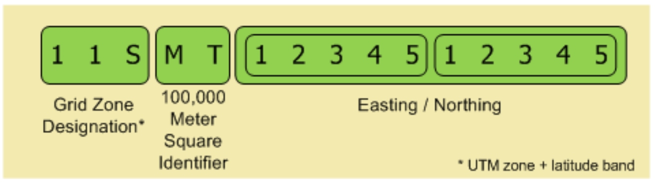


Figure 4. MGRS code [10]

2) *Road identification*

Traffic accidents occur between mobile objects moving along the same road or on an adjacent sidewalk. Traffic accidents rarely occur between objects moving along different roads even if they are close to each other. Although a vehicle could fall from an overpass and cause an accident below, such events are rare and neglected here. Therefore, it is necessary to estimate only the road on which each mobile object is moving. Since vehicles and pedestrians moving in the same direction sometimes take different directions at an intersection, identifying the road between two intersections is useful for managing traffic-related information. This idea is consistent with that of LDM.

Intersections are sometimes identified with a name or a combination of the names of the intersecting roads. Unfortunately, such identification is not universal. Therefore, in TDMS, intersections are identified by a MGRS point on which vehicles travel to some digitized directions. The road estimation function monitors the movements of mobile objects. When it detects a MGRS point on which vehicles travel to some digitized directions, it registers the point in the intersection table. Since the eight digits of the easting and northing codes specify 10-m square areas, the keys of the intersection table comprise the intersection ID (numeric), the grid zone, the 100,000-m square identifier, the four-digit easting code, and the four-digit northing code. The intersection ID is the primary key in the database.

The keys of the road table comprise the road ID (numeric), the start intersection ID, the end intersection ID, and the address of each FIFO buffer (enabling quick access to the buffers). The road ID is the primary key in the database.

3) *Mobile object information*

Real-time location information for each object is needed to manage traffic information. Since it would be difficult for each mobile object to continuously send its location information to the TDMS server, it is sent periodically (1 s would be a realistic period). Therefore, to estimate current location of a mobile object, its speed, direction, and time information are needed in addition to the periodic location information.

Furthermore, since some applications need additional information, such as energy consumption, the mobile object (MO) table contains the original grid zone (the first grid zone generated for a mobile object), the type of object (person, automobile, motorcycle, bicycle, etc.), the object ID (automatically assigned to each mobile object in each grid zone, the actual location (main street or side street), the digital location (MGRS code), the speed, the direction, the time, and the payload for application (energy consumption, etc.).

These data sets are stored in a FIFO buffer memory created for each road. The storage of the mobile object database in the practical memory of a server is described in Sub-section D.

Since the location values in these data sets are MGRS codes, the positional relationship between mobile objects can be grasped by comparing the 5th digits of the easting and northing codes in their MGRS code. The positional relationship between mobile objects can also be grasped by comparing the distances from the start intersection to each mobile object. It is difficult to determine in which lane a mobile object is traveling, and the lane information is needed to estimate the risk of an accident. Therefore, MGRS code is used as the location information.

C. *Area division*

Since the data size of TDMS is very big, and traffic-related mobile objects are distributed worldwide, the server should not be configured as a single server system but as a globally distributed server system, like Google Spanner [8]. Each distributed server is assigned a coverage area, as shown in Figure 5. The gateway server and distributed servers for each area could be implemented on the Internet or directly connected to 5G mobile base stations. The latter approach is better suited for exchanging real-time data. That is why the 5G base stations would be distributed worldwide and not all be operated by one operator. TDMS would thus be established on the Internet.

When a mobile object is registered in TDMS for the first time, its navigation terminal should first access the GW server to be assigned an area server. However, if each mobile object accessed the GW server as soon as it started traveling, a heavy load would be placed on the server. Therefore, when a mobile object starts to travel, it compares its current (recognized) location with the (memorized) location where it will finish traveling. If the recognized location is the same as the memorized one, the mobile object sends its information to the memorized area server. If the recognized location differs from the memorized one, it first accesses the GW server.

In the example shown in Figure 5, Japan is divided into nine areas. Area size is decided in accordance with the processing performance of each distributed server. The processing performance required depends on the number of mobile objects traveling in a management area. The average number of mobile objects in each area usually depends on the number of people living in that area. Since the population may change over time, the area division scheme must be flexible.

Dividing the physical areas on the basis of MGRS grid zones would probably be adequate. However, more investigation by simulation in which the practical number of mobile objects is considered is needed to make this decision.

Dividing the physical area into multiple areas causes problems. For example, if the area servers assign road IDs independently, roads crossing a boundary line are registered differently in the two area servers, making it difficult to determine the relationships of the connecting roads. The road reference table solves this problem;

Road ID assigned by initial area server: primary key

Road ID assigned by next area server

Since the grid zone code for the start intersection in the road ID differs from that for the end intersection for a boundary-crossing road, an area server can extract them. The initial area server sends the grid zone codes for the start and end intersections to the next area server and asks it to find the road for which the grid zone codes for the start and end intersections are the opposite in order to create a road reference database.

For example, as shown in Figure 6, vehicle 1 in area A traveling near the boundary line between areas A and B can obtain data for vehicle 2 but not for vehicle 3 traveling in area B even though vehicle 3 is approaching. A control scheme is needed to solve this problem. The scheme used is for an area server to periodically ask the servers in adjacent areas for data on mobile objects traveling along boundary-crossing roads and on roads that connect to boundary-crossing roads. These mobile objects have a higher probability of colliding with mobile objects in the border area in the near future. Therefore, each area server creates FIFO buffers for such roads and uses to them store data for mobile objects traveling on those roads.

A GW server can access each area server and thereby enable a PC to access an area server through the GW server. A PC can download data using the query commands described in Sub-section D.

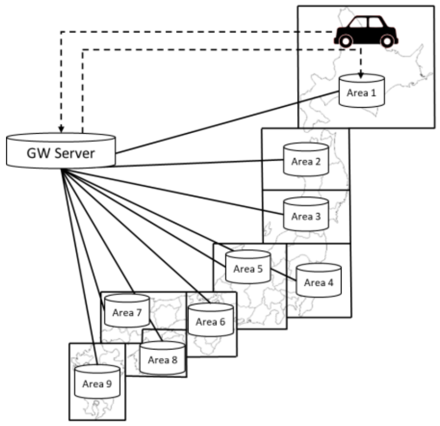


Figure 5. Example coverage of Japan with multiple area servers

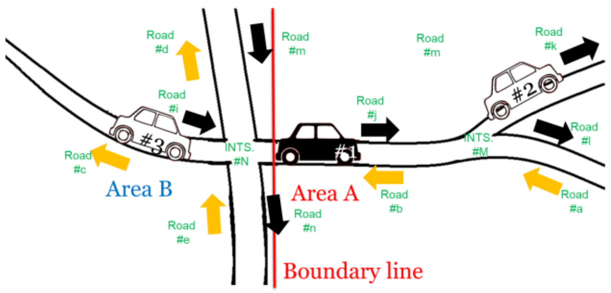


Figure 6. Image of vehicles traveling near area boundary line

D. *Storage structure*

Since TDMS needs very quick query processing, the memory structure is as shown in Figure 7. A FIFO buffer is created for each road and used to store data sets for each mobile object on that road during the specified period, such as 10 s from the present.

The newest data set is inserted into the first memory area of the buffer, and the latest previous data record is shifted to the second memory area. The remaining previous data records are similarly shifted. Data records flooded from the FIFO buffer are stored in an archive memory in the secondary memory, such as on a hard disk drive. The size of the FIFO buffer is set on the basis of the number of mobile objects during the specified period; it is set to greater than the optimum size during the start-up phase and then gradually reduced to the optimum size.

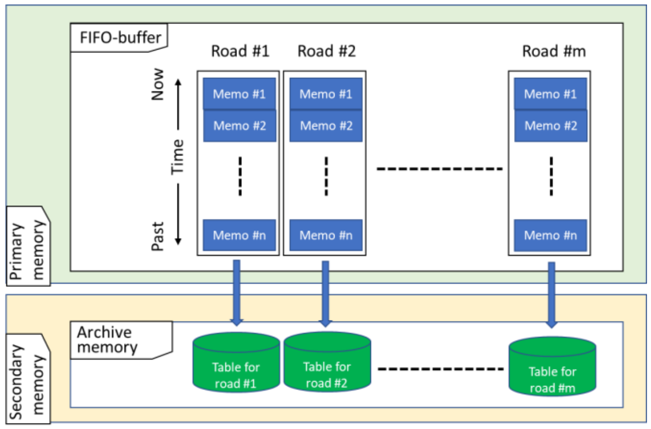


Figure 7. Storage structure in TMDS.

E. *Database commands*

The following database commands for a navigation terminal are used to insert data records for mobile objects into the FIFO buffer of an area server and for both a navigation terminal and PC to access an area server and extract and download selected data records for mobile objects.

[Create FIFO-buffer]

CREATE FIFO buffer, <road ID>;

used when the road estimation function detects a new road; sent to the management program, which creates a FIFO buffer for the designated road.

[Create other tables]

CREATE TABLE <table> (*attribute 1*, *attribute 2*, - - -);

used to instruct road estimation function or management program to creates a road table, road reference table, and intersection table; for road table, the create table command is

CREATE TABLE<road table> (Road ID, Start intersection ID, End intersection ID, Traveling direction, Address of each FIFO buffer).

[Insert data record into a FIFO-buffer]
INSERT FIFO buffer <road ID>
VALUE (v1, v2, v3, v4, v5, v6, v7, v8);
used to insert data sets for mobile objects sent from a navigation terminal into a FIFO buffer.

[Insert data record into other tables]
INSERT INTO <table> (attribute 1, attribute 2, - - -)
VALUES (v1, v2, - - -):
used to insert data into the road table;
INSERT FIFO-buffer <road> (Road ID, Start intersection ID, End intersection ID, Traveling direction, Address of each FIFO buffer).
VALUES (v1, v2, v3, v4, v5).

[Extract data records for mobile objects]
SELECT <MO_1>, <MO_2>, - - -
FROM <area server ID. road ID>,
WHERE <time>, <distance>:

Two types of attributes for mobile objects (MOs) are specified. One is object type (person, automobile, motorcycle, bicycle, etc. “*” means all types of objects. The other is the MO ID.

“time= present” means the newest data. The time can be set by using “T_1<=time<=T_2.” where T is in the format “ssmmhhddmmyy.”

The distance is calculated from both the object’s newest memorized time, location, and direction; “distance < M” means that the distance from its own is less than M. When an area sever is established for each grid zone, the area server ID is the grid zone ID.

For example, for vehicle 1 sending the following query command to an area server, the database management system on the area server identifies every mobile object within 10 m of vehicle 1 in the FIFO buffers for the road on which vehicle 1 is traveling and for the roads connected to that road:

SELECT Automobile
FROM 55R
WHERE time= present and distance < 10;

If vehicles 2 and 3 in Figure 1 are traveling within 10 m of vehicle 1, the data records for 2 and 3 are sent to vehicle 1 from the area server for which the grid zone is 55R.

V. CONCLUSION AND FUTURE WORK

The proposed Traffic Database Management System (TDMS) is designed to store and manage location, direction, speed, and payload data for every traffic-related mobile object worldwide. In this paper, its system design and database schemes to achieve the TDMS were introduced. Database commands that insert data records of mobile objects to the distributed area server, and request it to extract specified data record of mobile objects were introduced. The ability of TDMS to manage information for mobile objects over a wide area will make it useful for not only implementing the previously proposed “Cyber Parallel Traffic World” and for assisting drivers but also for reducing traffic problems, such as congestion, and for reducing energy consumption.

Future work includes evaluating the ability of TDMS to handle the big data necessary to manage information for traffic-related mobile objects worldwide.

ACKNOWLEDGMENTS

This work was supported by a Japanese Society for the Promotion of Science KAKENHI Grant (16K00276).

REFERENCES

[1] ETSI TR 102 863 V1.1.1, Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Local Dynamic Map (LDM); Rationale for and guidance on standardization, 2011.

[2] ETSI EN 302 895 V1.1.1, Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Local Dynamic Map (LDM) , 2014.

[3] Yoshitoshi Murata and Shinya Saito, "“Cyber Parallel Traffic World” Cloud Service in 5G Mobile Networks," Journal of ICT Standardization, River Publishers, Volume 2, No. 2 (November 2014), Special Issue on ITU Kaleidoscope 2014: “Towards 5G”, pp. 65-86, 2014.

[4] S. Ghemawat, H. Gobioff, and S. T. Leung, “The Google File System,” In Proceeding of SOSP 2013, 2013.

[5] F. Chang, et al., “Bigtable: A Distributed Storage System for Structured Data,” In Proceeding of OSDI 2006, 2006.

[6] J. Dean and S. Ghemawat, “MapReduce: Simplified Data Processing on Large Clusters,” In Proceeding of SOSP 2007, 2007.

[7] SP3 – SINTECH – Innovative Technologies, "Safespot Integrated Project - IST-4-026963-IP: Deliverable," http://www.safespot-eu.org/documents/SF_D3.5.4_ KeyConceptsAndExploitation_v1.2.pdf. [retrieved: May, 2019]

[8] Kenya Sato, et al., “Stream LDM: Local Dynamic Map (LDM) with Stream Processing Technology," The Science and Engineering Review of Doshisha University, Vol. 53, No. 3, pp. 28-35, 2012.

[9] The UTM Grid - Military Grid Reference System, Natural Resources Canada, https://www.nrcan.gc.ca/earth-sciences/geography/topographic-information/maps/9789, [retrieved: March, 2019]

[10] James C. Corbett et al., "Spanner: Google’s Globally Distributed Database," ACM Trans. Comput. Syst. 31, 3, Article 8, pp.1-22, 2013.