Jan Janech – Tomas Baca *

DISTRIBUTED DATABASE SYSTEMS IN VEHICULAR AD-HOC NETWORK

The article introduces basics of classic Distributed Database System (DDBS) and explains its unsuitability for dynamically changing networks such as Vehicular Ad-hoc Network (VANET). Thus a concept of a new architecture is proposed. This new architecture is designed for use in mobile networks as it solves problems of classic architecture, however as explained at the end of the article, it has some limitations of its own.

Introduction

Vehicular Ad-hoc Network (VANET) is a form of Mobile Ad-hoc Network (MANET) which interconnects vehicles among themselves and vehicles with nearby roadside equipment. The main goal of VANET is to provide safety and comfort for passengers. Typically, vehicles are moving rapidly in large geographic areas. Equipped with wireless network devices they would have direct network access only to several other nodes in close distance. Furthermore connection to some of them would be lost in a short time but others would become available.

In such an environment plenty of new data are produced every moment however they are to be obsolete by even newer data soon. In certain applications these data could become useful if they were accessible and aggregated in a proper moment. This leads to the idea of Distributed Database System (DDBS) in VANET.

A database of parking places integrated into vehicles can serve as an example of such a system. Each vehicle acts like one site in a distributed database system. After the vehicle is completed and leaves the car factory, its built-in database is empty. After the system becomes activated, the vehicle starts to query the distributed database for information about parking places in surroundings periodically. When the vehicle receives some data, it saves them into its own built-in database. The sites providing these data could be either other vehicles or parking places.

Based on a similar principle, the vehicle's navigation system could query for data about traffic situation. This way it would be able to provide up-to-date information about traffic jams or accidents to the driver.

In this article a new architecture of DDBS is introduced because, as shown below, classic architecture of DDBS is not suited for dynamic environment of VANETs.

1. Classic architecture of DDBS

To demonstrate the problems of implementing the classic DDBS architecture in a dynamic networks environment, some of the DDBS architecture basics are presented below. They are not supposed to provide an exhausting description of the architecture, but to focus on the parts causing problems with the queries in dynamic networks.

Fig. 1 shows the DDBS architecture from the viewpoint of data organization. It is a simple multi-layer model with four layers. Each one of the layers represents a certain view of the data themselves.

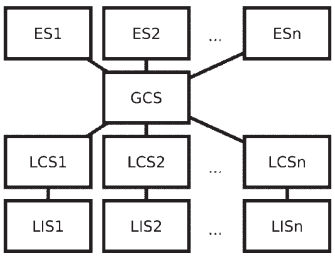

Fig. 1 DDBS reference architecture [7]

1. *LIS (Local Internal Scheme)* is a physical representation of the data stored in one site. It is an analogy of the internal scheme from centralized databases.
2. *LCS (Local Conceptual Scheme)* describes the logical organization of data in one site. It is used to handle the data fragmentation and replication.
3. *GCS (Global Conceptual Scheme)* represents the logical organization of data in the whole distributed database system. This

* **Jan Janech, Tomas Baca**
Department of Software Technology, Faculty of Management Science and Informatics, University of Zilina, Slovakia
E-mail: jan.janech@kst.uniza.sk

layer is an abstraction of the fact that the database system is distributed.
4. *ES (External Scheme)* represents the user view into the distributed database. Each external scheme defines which parts of the database the user is interested in.

The fact that the user uses only the global conceptual scheme, and they do so by means of the views defined in the external scheme, assures that the user can manipulate with the data regardless of their position in the distributed database system. Therefore it is necessary to have a mapping from every local conceptual scheme to the global conceptual scheme. This mapping, named *GD/D (Global Directory/Dictionary)*, is defined as a part of the distributed database system. Its main role is to provide access to mapping between local conceptual schemes and the global conceptual scheme. Thus it has to be accessible from each of the sites sending queries into the system.

There are several possibilities of storing GD/D, however all of them require that the system knows all of its parts all the time. Whether the GD/D is stored in one place or distributed throughout the system it has to be accessible as a whole.

This is not possible in a dynamic network. Either in an Ad-Hoc network, or even in a dynamically changing managed network, there is no way to ensure communication between all the sites in the system. Under these circumstances the GD/D cannot be used to locate data in a distributed system.

2. Existing solutions

Till now most of the real applications solve this problem by avoiding dynamic networks as a transport channel. They tend to choose reliable network types such as GPRS [3]. However, for a certain type of applications dynamic networks seem to become a suitable alternative. At the present time there are several attempts to create an architecture that would operate in MANET environment [1] [5]. All of them are based on the same principle. There are three types of nodes in the system:
1. *RN - Request Node* – A node that is able to query a database.
2. *DBN - Database Node* – A node that stores data and that is able to process user's queries.
3. *DD - Data Directory Node* – A node that holds GD/D. Its responsibility is to create a query plan and to control the execution of the plan.

A request node sends a query to a data directory node which creates a query plan and subsequently delegates an execution of its parts to proper database nodes.

Yet from the used terminology it is obvious that the design emphasizes the network communication. The biggest problem for deployment of distributed databases in the dynamic network environment is the necessity of the knowledge of all the sites in the system. This problem persists in this solution as well because the GD/D is still used.

3. Proposed solution

The only way to make sure it is possible to use the distributed database system in the dynamic networks environment is to remove the GD/D from the system and replace it with a different principle. As it has been already said, the GD/D describes the mapping between the local and global conceptual schemes. Without the mapping, the system does not know where the data are located and how to query them.

Using the GD/D in the dynamic networks environment is impossible because it requires knowledge of the whole system (global directory). In a dynamic network every site knows its immediate surroundings only. So querying a distributed database is fairly limited in such environment. The only sites which can be addressed with queries, except the one requesting information, are those in the immediate surroundings in the network. Thus the system naturally creates virtual clusters of sites that can communicate with each other. The clusters overlap, so every site of the cluster can communicate with a different set of sites.

This implies a possibility to move the directory from the global level to the cluster level. This principle may be called *Cluster Directory/Dictionary (CD/D)*. In this way the directory contains only the mapping of the global conceptual schemes of the sites accessible to the local conceptual schemes exclusively. However, there might be problems in this system due to clusters overlapping. Therefore, the local conceptual scheme of each site has to be mapped onto the global conceptual scheme in several CD/D.

The cluster has no central site as it is strictly peer-to-peer system; however CD/D has to be stored somehow. The only possible way is to store parts of CD/D locally only. One site has mapping from the global conceptual scheme onto its own local conceptual scheme. The CD/D then consists of all of the local mappings in the cluster.

Executing local queries in such a system is very simple because all of the accessible data and all of the mappings are available, thus not needing to communicate with other sites in the cluster. But since queries are normally done globally all over the whole accessible part of the database [8], the communication between the sites in the cluster is necessary.

Generally there are two possibilities of communication in a database cluster:
- communication with registration into the cluster,
- direct communication of the sites.

The first possibility requires registration of the site within the cluster. By registering each site into the cluster the system promotes the virtual clusters into associated groups of sites. The sites are aware of each other; therefore, they can create and store a complete copy of the CD/D locally. Because of the clusters overlapping, each site can have more than one CD/D stored. This may simplify the process of queries because every site knows where the data are located.

On the other hand, it is impossible to register every new site into the cluster when it comes to such a rapidly changing environment as VANET certainly is. Either if sites were searching for the nearby clusters or clusters were sending notifications about its presence to the surroundings, these notifications would have to occur so often that the network would be overloaded.

On the contrary, the direct communication does not require any form of registration. Instead, the sites communicate directly with each other. Therefore, the communication solely occurs when it is needed, and the network can be overloaded only by sending too many queries during a short period of time.

However, no site knows the whole CD/D, so the query execution cannot be entrusted to a particular site. Instead of that a query has to be sent to the entire cluster and each site needs to detect whether it can process it or not. The response to the queries can be sent directly to the query site. Since communicating with the whole cluster is required, the query has to be sent as broadcast or multicast messages.

4. Querying database in dynamic network

The new form of the site-to-site communication requires a new architecture of the distributed database system. In spite of that, there are some similarities with the traditional architecture.

There are two defining architectural principles of the traditional distributed database systems. "The first is that the system consists of a (possibly empty) set of query sites and a non-empty set of data sites."[6] "The second is that each site (query or data) is assumed to consist of a single independent computer."[6]

These principles are not specific to any particular type of the distributed database environment. So they can be applied equally to the distributed database system in the dynamic networks environment as well. This leads to the conclusion that there are two basic types of sites:

- *Data site* – contains fragments of the distributed data. From the viewpoint of data organization it preserves the local internal scheme and the local conceptual scheme. As it was stated before, it also contains a part of the CD/D.
- *Query site* – acts as the user's interface to access the distributed database system. From the data organization point of view it preserves the external scheme.

Queries are sent via the network as broadcast or multicast messages. This way a query site does not know who is to process the query, or if anyone is to process it at all. This restricts the query process in two ways:

- *Asynchronous communication* is required. There is no way how to determine the number of data sites that will respond to the query. Therefore, the querying site has to process responses after they come, instead of waiting for the known number of responses, like it would have to do in the traditional architecture.
- The query syntax needs to be *well structured*, so it can be easily divided into smaller subqueries. The data site which has received the query can only contain a part of the data required for the response (e.g. when the data are horizontally or vertically fragmented). So it has to send one part of the query back to the cluster.

The UML sequence diagram in Fig. 2 shows an example of the communication between querying sites and three data sites. A query in a form of an expression formatted in a certain way is sent by means of a broadcast message into the whole cluster. The data site 1 has some fragments of the queried data, thus it returns them directly to the query site. The data site 2 does not have any data, so it stops the query processing immediately. The data site 3 also has some data; therefore it returns them to the query site. It

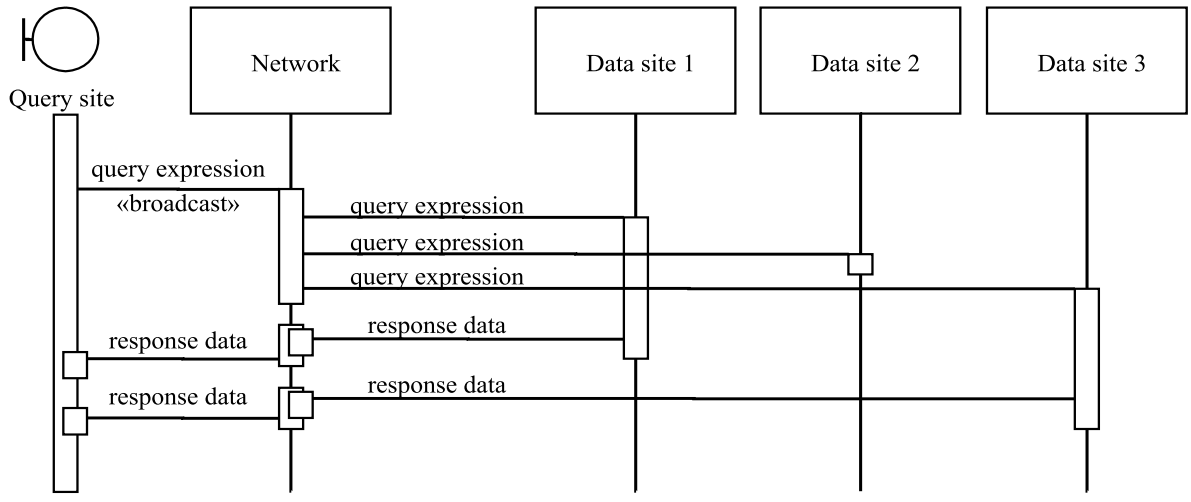

Fig. 2 Processing a query in the cluster

is up to the query site to decide when to stop receiving answers. It is important to be aware of the fact that the connection between any of the sites and the rest of the cluster can be interrupted at any time of processing the query.

Another question is formatting the query. A classic SQL query is not structured very well, and dividing such a query into smaller parts is not a simple task. However, the relational algebra allows doing so. The only problem is the bad format of the query, which resembles natural language rather than a relational algebra expression. Another possibility is using a relational algebra expression directly, in a form of lambda calculus [2] [4]. The lambda function syntax allows to extract any part of the expression, and to evaluate it individually. In this way the user can easily create queries, send them over the network and divide them if necessary.

5. Limitations of new architecture in VANET

The architecture introduced in this article does not intend to be a universal database system. It is a specialized solution for the implementation of distributed database systems in dynamic network environment. Its usage is limited in several ways which is obvious especially when used in VANET. They are caused by rapid movement of vehicles at a large geographic area.

Firstly, it is impossible to access the whole GD/D. Even the CD/D exists only virtually and every node can access only its local part. Thus it's very difficult to optimize queries.

It is impossible to provide referential integrity because the distributed database can be partially or even fully inaccessible. In fact, in different times it has access to different subsets of the database.

For the query site there is no way of how to find out whether it has already received all the data it has requested or if it has to wait longer.

This architecture allows just read-only access to the distributed database. Destructive operations can be executed locally only.

In wireless environment, all the nodes that are within the signal range and use the same channel, share the bandwidth of the transport media. Too many queries and broadcast messages can easily overload the network.

Due to movement of vehicles it is even possible that a data site would become inaccessible to the query site when the query is being processed. Thus it is important for response to be well structured and divided into small fragments. If every fragment can be understood separately, then the query site can make use of received data even if a part of the response has been lost.

Another problem of this architecture is that the implementation of AAA (Authentication, Authorization, Accounting) services is more difficult because the sites are not aware of each other. This can be partially solved by signing the requests electronically, and encrypting the data using the PKI (Public Key Infrastructure). However signing and encrypting goes against previously mentioned idea of small fragments, because every fragment would have to be signed or encrypted separately which means a lot of extra data.

These limitations are results of the environment rather than the defects of the architecture. Thus the architecture can only be used when these problems are not limiting.

6. Example of an application

VANET does not provide any kind of guarantee of how many data would request site receive as responses. In fact it may receive no data at all. Thus it is not suited for applications where reliable connection is required.

However, if query sites were interested especially in recent data that are specific to their present geographic location and data sites were producing new data on their way. In case that receiving all of the responses is not crucial but receiving at least some of them would pose some kind of advantage, then VANET would represent environment in which this is possible.

A database of parking places integrated into vehicles could serve as an example of such a system. Each vehicle would act like one site in a distributed database system. After the vehicle was completed and left the car factory, its built-in database was empty. After the system became activated, the vehicle started to query the distributed database for a list of parking places periodically. It would receive information from other vehicles about positions of parking places in its proximity.

Positions of parking places could be of course inserted to the vehicle's database using some kind of side channel, for example from data media. However, intelligent parking places are equipped with sensors detecting which slots are available and which are not. This information can be queried and stored by vehicles that are passing closely. Afterward as this vehicle moves to the different location they can be queried by others and provide them with information about situation on that parking place.

7. Conclusion

VANET is a kind of dynamic network with rapidly changing situation and without guarantee of service availability. As such, it is not suited for critical applications which rely on delivery of all the data. However, there are several applications that do not require availability of all the data but rather the most recent data that are present in the relative proximity to the querying site.

Examples of such an application are distributed database of parking places in the town or information about present traffic situation in the region the vehicle is headed toward.

The present architectures of DDBS are not suitable for such an environment. The proposed architecture removes their limita-

tions; however it brings few of its own as a result of principles of dynamic networks.

In a short time a prototype of the system will be created so that its behavior can be tested experimentally. It would be possible then to compare this new architecture with the existing ones.

Acknowledgement:

This contribution is the result of the project implementation: Centre of excellence for systems and services of intelligent transport, ITMS 26220120028 supported by the Research & Development Operational Programme funded by the ERDF.

"Podporujeme vyskumne aktivity na Slovensku/Projekt je spolufinancovany zo zdrojov EU."

References

[1] ARTAIL, H., SAFA, H., EL ZINNAR, R., HAMZE, H.: A Distributed Database Framework from Mobile Databases in MANETs, In: *Proc. of 3rd IEEE International Conference on Wireless and Mobile Computing, Networking and Communications,* Oct. 2007, pp. 54–61.

[2] HILLEBRAND, G.G., KANELLAKIS, P.C., MAIRSON, H.G.: Database Query Languages Embedded in the Typed Lambda Calculus, In: *Proc. of 8th IEEE Symphosium on Logic in Computer Science,* Montreal, Canada, June 1993, pp 332–343.

[3] KRSAK, E., HRKUT, P.: *Operativne riadenie a navigacia zasahovych vozidiel hasicskeho zboru pomocou informacneho systemu s distribuovanou architekturou* [Operative Management and Navigation of Fire Brigade's Vehicles Using Information System with Distributed Architecture], In: GIS : Ostrava, 2008, ISBN 978-80-254-1340-1 (in Slovak).

[4] MERUNKA, V.: *Objektove modelovani* [Object-oriented Modelling], In: Praha : Alfa Nakladatelstvi, 2008 (in Czech).

[5] RAHBAR, A., MOHSENZADEH, M., RAHMANI, A.M.: HDD3M: A New Data Communication Protocol for Heterogeneous Distributed Mobile Databases in Mobile Ad Hoc Networks, In: *Proc. of Intern. Conference on Education Technology and Computer,* April 2009, pp. 93–98.

[6] OZSU, M. T.: *Distributed database systems,* Available from: http://softbase.uwaterloo.ca/~ddbms/publications/ozsu/EIC/eic.pdf.

[7] OSZU, M. T., VALDURIEZ, P.: *Principles of Distributed Database Systems.* In: Pearson Education, 2nd edition, 1999.

[8] SOKOLOVSKY, P., POKORNY, J., PETERKA, J.: *Distribuovane databazove systemy* [Distributed Database Systems] In: Academia : Praha, 6th edition, 1992 (in Czech).