# Designing a Decentralized Traffic Information System – AutoNomos

Axel Wegener[1], Horst Hellbrück[2], Stefan Fischer[1], Björn Hendriks[3], Christiane Schmidt[3] and Sándor P. Fekete[3]

[1] Institute of Telematics, University of Lübeck, Germany, {wegener,fischer}@itm.uni-luebeck.de
[2] Department of Electrical Engineering, University of Applied Sciences Lübeck, Germany, hellbrueck@fh-luebeck.de
[3] Institute of Operating Systems and Computer Networks, Braunschweig University of Technology, Germany, {b.hendriks,c.schmidt,s.fekete}@tu-bs.de

**Abstract.** We propose a decentralized traffic information system—AutoNomos—that is based on a thorough investigation of the properties of traffic and recommends a hierarchical data aggregation and forwarding for providing individualized information and support to road users. Our approach differs from work in the field by consequently applying local rules and local decentralized data processing, which turns out to be a key property of robust and scalable computing systems. We present a flexible VANET middleware that assists the application development by providing generic functionality for traffic applications. We discuss the architectural design of the overall system and provide solutions of important design concepts demonstrating the innovation of the approach.

## 1 Introduction

Existing Traffic Information Systems (TIS) with fixed infrastructure and centralized control are complex to install and expensive. Additionally, most of them provide outdated or incorrect predictions. The imprecise analysis and forecast results from the fact that only a fraction of the road system is monitored.

In the recent past a lot of effort has been spent on developing TISs that use a Vehicular Ad-Hoc Network (VANET) for communication. TISs monitor road conditions and allow single vehicles to optimize their individual driving route as well as they improve the overall traffic flow. These applications span hundreds of square kilometers and measure traffic density or detect and avoid traffic jams.

In future, all these applications will work simultaneously and will deal with unreliable ad-hoc communication, the limited view of single vehicles and the resulting inconsistent states; also the interpretation of data between the vehicles need to be defined consistently. Our AutoNomos approach assists the application development by acting as a flexible VANET middleware providing generic functionality for collaborative traffic applications.

The next section provides an overview of related work. We present our AutoNomos middleware approach with its key concepts and architecture in Section 3. The paper concludes with a summary and directions for future work.

## 2 Related Work

We discuss the related work in two steps: first, decentralized traffic information systems, since those represent the major field of applications for our approach; second, an overview of existing middleware approaches.

In the scope of the German research program *Fleetnet* (2000-2003) and *Network On Wheels* (2004-2008) techniques for lower layers like radio interface, medium access and routing have been developed. Some groups dealt with applications like internet integration as well as TIS (see [1]).

Based on these research programs, the Car-2-Car Consortium was founded to achieve a standardized frequency allocation and compatible lower layer protocols between manufacturers (see internet draft for VANET communication [2], moreover, an EU-wide frequency band is currently reserved for VANETs).

In SOTIS (Self-Organized Traffic Information System [3]), the road is partitioned virtually into fixed sized segments, depending on the type of the road. Measured traffic density and mean velocity is distributed within the network without further processing or aggregation. Thus, similar to centralized approaches the problem of data processing and applying of strategies remains unsolved.

Various middleware techniques have been developed in the related research area of Mobile Ad-Hoc Networks (MANETs) to assist application programmers in order to keep such a large system manageable. We follow the classification of [4] and for clarification, we add one representative approach.

Since MANETs often monitor their environment, **event based programming** presents an obvious strategy. Classification of and reactions to certain events define the behavior of network nodes (cf. [5]). Since communication costs grow with increasing distance between nodes, actions are triggered by **local rules** based on nearby data (cf. [6]). In MANETs measured data is more important than (error-prone) nodes. A **data centric** approach hides communication aspects from the application programmer (cf. [7]). **Virtual machines** offer two advantages. On the one hand, the heterogeneity of network nodes can be hidden by providing a virtual machine on nodes as shown in [8]. On the other hand, a program can be modeled as a mobile agent that copies itself from one node to the other in order to complete a desired task (cf. [9]). Most middleware approaches provide **communication abstraction** to relieve the application programmer from the network communication part as shown, e.g., in [10].

Currently, research addresses various fields within inter-vehicle communication like medium access and applications, but we still lack comfortable programming abstractions for VANETs. Abstractions in related work are also advantageous for VANETs; thus, we incorporate them into our proposed AutoNomos approach in the next section.

## 3 AutoNomos Architecture

In this section, we present our *AutoNomos* architecture and provide an easy and structured programming model. In order to achieve this, two key concepts

are used. *Hovering Data Clouds* (HDCs) virtually store local data and are not bound to a specific vehicle. The higher level structure of *Organic Information Complexes* (OICs) handle large scale traffic structures. After introducing HDCs and OICs, we describe the used data dissemination scheme and introduce the software architecture inside the vehicles. A traffic jam scenario illustrates our approach in the following.

### 3.1 Hovering Data Clouds

In most cases data relates to some phenomenon that occurs on the road. If—as in road traffic—the network nodes move, it is no longer advantageous to bind data to individual nodes in an uncorrelated way. Instead, *Hovering Data Clouds* (HDCs) store data that is pinned to phenomena or specific locations and no longer bound to nodes; data moves independent of its carriers to follow its phenomenon. The set of nodes holding specific data can even change completely, passing the data on to oncoming nodes in the phenomenon's area. We have described this key concept in detail in [11] and have demonstrated the feasibility of the concept.

**Fig. 1.** A Hovering Data Cloud at traffic jam's back warns approaching vehicles.

Figure 1 shows an HDC at the back of a traffic jam. The first vehicles, that brake due to an obstruction, detect their deceleration below a certain threshold and send their positions to oncoming vehicles. Thus, with every braking vehicle, more data is gathered until a *Traffic Jam Back* HDC is established at the vehicles' positions. Due to slightly different local knowledge, vehicles may establish slightly different HDCs, that get correlated later on, approaching a common understanding. HDCs disseminate their data as *data units* for two reasons: first, vehicles in the phenomenon's area need to maintain the HDC (intra-HDC communication); second, vehicles outside the HDC's area get informed about the HDC's phenomenon (inter-HDC communication). We will deepen this thought in the software architecture part.

### 3.2 Organic Information Complexes

As described above, HDCs detect and follow traffic phenomena in a simple and efficient way. To obtain higher level information, data of several spatially distributed phenomena and their particular HDCs need to be aggregated. Figure 2 shows how an Organic Information Complex (OIC) achieves this objective by hierarchically aggregation.

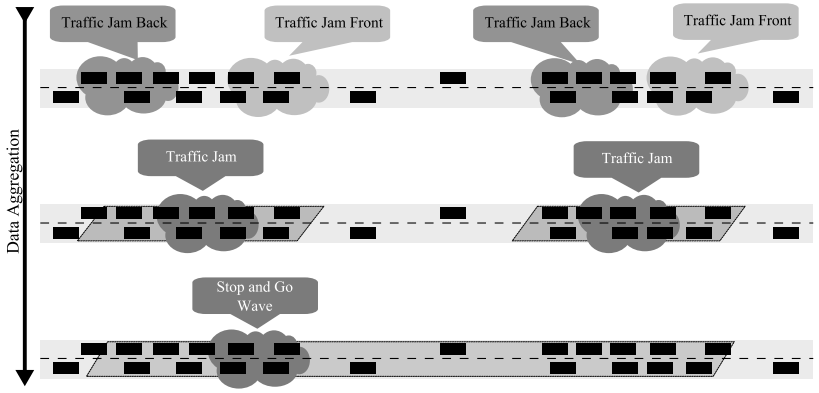


**Fig. 2.** Data aggregation to recognize a traffic jam by an Organic Information Complex

The topmost picture shows *Traffic Jam Back* HDCs and *Traffic Jam Front* HDCs as a result of local measurements of each vehicle. Those HDCs disseminate *data units*, that describe their observed phenomena. When data units of matching phenomena get together, *Traffic Jam* OICs are derived in this example as result of the first aggregation (cf. middle part of Figure 2). The aggregated data is the traffic jam's length and its movement over time, which could not be observed by one particular HDC.

To store the newly gathered information, again an HDC is used that arises just where the accordant data gets aggregated. Different views of the same phenomenon get correlated to approach a common understanding. Thus, instead of struggling with consistency of data, the AutoNomos system is able to turn the table and achieves a high robustness from redundant data processing—just by its design of evolving HDCs.

This aggregation process is repeated in a hierarchical manner. The lower part of Figure 2 shows a *Stop and Go Wave* as end result.

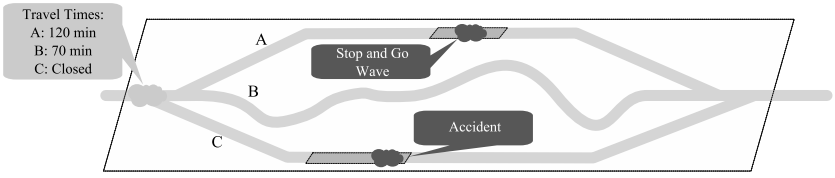


**Fig. 3.** Organic Information Complex on a road network aggregates travel suggestions

We continue with this example by looking from a higher perspective to a road network as depicted in Figure 3. Here, OICs discovered congestions on roads *A* and *C*. Those again send out data units towards incoming vehicles containing

information about their monitored phenomena. When these data units reach intersections on the road network a journey time prediction for road sections arises as result of the aggregation process that helps to choose an appropriate route. Due to this in-network data aggregation, the overall system remains scalable.

HDCs and OICs are very flexible and form themselves self-organized wherever necessary. Also correlation and aggregation is not restricted to fixed location, but happens as a result of matching data. Thereby, data structures and algorithms are not limited by our middleware; instead, application developers define HDCs and OICs as well as correlation and aggregation mechanisms to build their own applications.

### 3.3 Data Dissemination Protocol AutoCast

For the efficient intra- as well as inter-HDC communication between vehicles, we developed the *AutoCast* data dissemination protocol. AutoCast is optimized for scalability and dynamics as well as density variations in networks. A sending node/vehicle encapsulates data into *data units*, that contains a dissemination region, time stamp, lifetime and the data itself. The AutoCast transport layer on every vehicle handles a local set of data units automatically and offers transport service to the upper layers.

AutoCast itself adapts to network density, by implementing probabilistic flooding when the network is very dense. On the other extreme, if the network is sparse and partitioned, periodic beacons enable storing and forwarding of data units in an efficient way. Thereby, vehicles do not switch between these operating modes, but rather adapt their behavior smoothly based only on local rules. For protocol details and evaluation of AutoCast we refer to [12].

### 3.4 Software Architecture

After introducing the key concepts of HDCs, OICs and AutoCast, Figure 4 provides the software architecture that is implemented in every vehicle. Our approach assumes the availability of wireless broadcast communication, e.g., DSRC/WAVE based on IEEE 802.11p [13]. Based on the broadcast communication, our first layer (*Transport Layer*) provides AutoCast for robust data dissemination as introduced in the last section.

The second layer (*HDC Layer*) stores and processes data units. Therefore, a data repository stores all locally known HDCs, representing the status of a node. The HDC repositories of nearby vehicles do not need to be fully consistent, but converge to a common understanding by ongoing correlation.

HDCs are created by *Data Mining* of local physical sensors; relevant data (e.g., black ice that provokes the ESP system) is filtered out and fed into data units for further processing.

The central processing component on the HDC layer is the *Correlation Handling*. Here, all incoming data units are processed—no matter if they result (1) from local sensors, (2) from the transport layer, which means from other vehicles

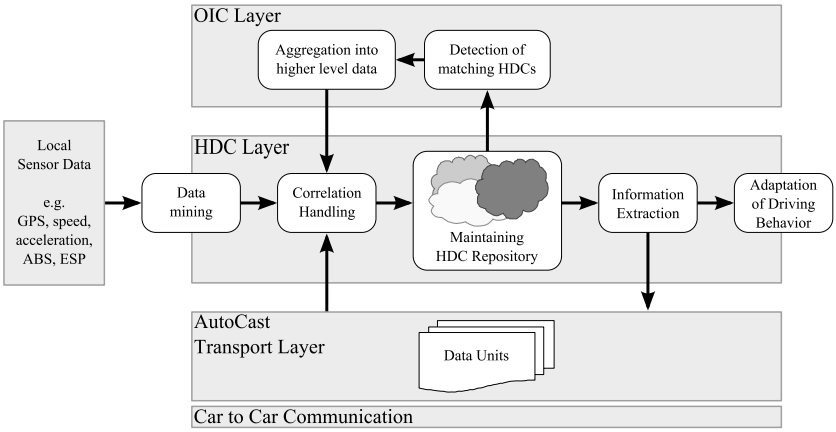


**Fig. 4.** Software Components available on a every Vehicle.

or (3) from the upper layer. The incoming data units are correlated to stored HDCs from the *HDC Repository* as exemplified in Section 3.1. If correlation functions identify one or more HDCs in the repository that correlate with the oncoming data unit, the particular HDCs get refreshed by applying update rules. Otherwise, a new HDC is formed based on the new data unit.

The HDC repository is maintained periodically to remove HDCs that are outdated or out of scope.

The *Information Extraction* component periodically generates data units from the stored HDCs that are disseminated by AutoCast. Such a data unit contains either a report, e.g., for approaching vehicles, with only a digest of the HDC's data (inter-HDC communication), or the complete data of an HDC to establish the same HDC in other vehicles. Doing so, an HDC can follow its phenomenon (intra-HDC communication).

Besides sending data units, extracted data can also be given to the *Adaptation of Driving Behavior* component. We foresee *Adaptable Distributed Strategies* (ADS), which are, e.g., able to dissolve traffic jams by local rules. Due to space limits, a detailed analysis of ADS is out of the scope of this paper.

The *OIC Layer* observes the HDC repository for matching HDCs that can be aggregated into a new, higher level HDC as demonstrated in Section 3.2. Once identified, the *Aggregation* component of the OIC layer semantically combines the particular HDCs into a higher level data unit, that is fed back into the correlation handling module of the HDC layer.

## 4 Conclusion and Future Work

In this article we propose a decentralized self-organizing traffic information system—namely AutoNomos—to match the challenges of road traffic as dynamic and unpredictable large scale system, where, e.g., the occurrence and behavior

of traffic jams is unforeseeable. We presented our system design and introduced the concepts of *Hovering Data Clouds* and *Organic Information Complexes*, that allow for the required data correlation and aggregation as well as information extraction to improve traffic flow. Application developers can implement VANET applications based on our approach that run concurrently without interfering with each other.

In the next step of our system development we will complete the implementation of the AutoNomos system and specify the application programming interface for the proposed middleware. In addition, we will further improve the algorithm for traffic jam detection and implement *Adaptable Distributed Strategies* to improve the overall road traffic performance.

## References


1. W. Franz, H. Hartenstein, and M. Mauve. *Inter-Vehicle-Communications Based on Ad Hoc Networking Principles.* Universitätsverlag Karlsruhe, 2005.
2. R. Baldessari, A. Festag, and M. Lenardi. C2C-C Consortium Requirements for Usage of NEMO in VANETs. Internet Draft, http://www.ietf.org/internet-drafts/draft-baldessari-c2ccc-nemo-req-00.txt, 2007.
3. L. Wischhof, A. Ebner, H. Rohling, et al. Sotis - a self-organizing traffic information system. *Proc. of the 57th IEEE Vehicular Technology Conference*, 2003.
4. K. Römer. Programming paradigms and middleware for sensor networks. *GI/ITG Fachgespräch Sensornetze*, 2004.
5. S. Li, S. H. Son, and J. A. Stankovic. Event detection services using data service middleware in distributed sensor networks. *Information Processing In Sensor Networks: 2nd International Workshop*, 2003.
6. K. Terfloth, G. Wittenburg, and J. Schiller. FACTS - a rule-based middleware architecture for wireless sensor networks. *Proc. of the First IEEE International Conference on Communication System Software and Middleware*, 2006.
7. S. R. Madden, M. J. Franklin, J. M. Hellerstein, et al. Tinydb: an acquisitional query processing system for sensor networks. *ACM Trans. Database Syst.*, 30(1):122–173, 2005.
8. D. Gorgen, H. Frey, and C. Hiedels. JANE – the java ad hoc network development environment. *Proc. of the 40th Annual Simulation Symposium*, 2007.
9. S. Dolev, S. Gilbert, E. Schiller, et al. Autonomous virtual mobile nodes. Technical report, 2005.
10. D. Pfisterer, C. Buschmann, H. Hellbrück, et al. Data-type centric middleware synthesis for wireless sensor network application development. *Proc. of the 5th Annual Mediterranean Ad Hoc Networking Workshop*, 2006.
11. A. Wegener, E. M. Schiller, H. Hellbrück, et al. Hovering data clouds: A decentralized and self-organizing information system. *Proc. of International Workshop on Self-Organizing Systems*, 2006.
12. A. Wegener, H. Hellbrück, S. Fischer, et al. Autocast: An adaptive data dissemination protocol for traffic information systems. *Proc. of the 66th IEEE Vehicular Technology Conference*, 2007.
13. D. Jiang, V. Taliwal, A. Meier, et al. Design of 5.9 ghz dsrc-based vehicular safety communication. *IEEE Wireless Communications*, 13(5):36–43, Oct. 2006.