A component-based approach for (Re)-configurable routing in VANETs

Vatsala Nundloll, Gordon Blair, Paul Grace

A Component-based Approach For (Re)-Configurable Routing in VANETs

Vatsala Nundloll, Gordon S. Blair, Paul Grace
Computing Department, Lancaster University, Lancaster, UK
{nundloll, gordon, gracep}@comp.lancs.ac.uk

ABSTRACT

With the increasing capability of vehicular communications technology, VANETs (Vehicular Ad-Hoc Networks) have witnessed significant development. Many VANET routing protocols have been proposed and operate well in the specific network they have been developed for such as dense or sparse networks but tend not to operate well or interoperate with different networks. This is a challenging requirement and we address this issue by proposing a (re-)configurable implementation that can adjust the behaviour of the protocol to different operating environments and application requirements. One possible way of achieving this is through a component-based architecture consisting of re-usable components that can be plugged in and out of the system. The paper illustrates how this componentization process can be carried out and how the given architecture is then configured to suit varying network states. The paper also identifies commonalities that exist across different routing protocols, leading up to the design of a generic component-based platform which can be used for VANET routing protocol development.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems – *Distributed applications*

General Terms

Design

Keywords

Vehicular ad-hoc networks, components, dynamic re-configuration, configuration, reflection

1. INTRODUCTION

VANETs are an upcoming type of network that typically consist of vehicles interacting with each other, known as Inter Vehicle Communication (IVC), or with the Internet through access points found along the road, known as Road-Vehicle Communication (RVC). The goal of VANETs is to share information such as traffic data and road safety warnings in order to avoid accidents and traffic congestion. Even though VANETs are considered to be a class of MANETs (Mobile Ad Hoc Networks) **[1]**, they exhibit specific challenges due to their distinctive network characteristics. Their salient features **[2]** can be summarised as: a frequently changing topology, high node mobility and repeated partitioning of the network leading to network disconnections. The operation of VANETs is also largely dictated by the network density and by other factors including drivers' behaviour. Dense areas experience an overflow of messages sent out to vehicles while sparse areas encounter loss of messages because of lack of vehicles to propagate them. This implies new buffering techniques to capture the messages both during a limited connectivity and a congested bandwidth. On the other hand, drivers may show different reaction to messages received and also their high speed may cause the network topology to change. Moreover, given the high flux of VANETs, considerable weight is imposed on the successful deployment of routing protocols in such networks.

ARM 2009, December 1, 2009, Urbana Champaign, Illinois, USA.

Consequently, it is imperative to devise new routing protocols for VANETs to ensure a reliable message distribution. However it remains a challenging task to find and preserve a route in VANETs, and as such the implementation of routing protocols is a complex and time-consuming task. Given the highly changing nature of VANETs, such an implementation cannot be a static one i.e., one that should be applied for a given set of conditions. Rather, there needs to be a way to make the implementation more dynamic, so that it can be easily configured to varying network conditions, e.g. as a vehicle moves from a dense to a sparse network, or communicating with other vehicles or the Internet.

A component-based approach is one possible and frequently advocated approach to provide a configurable and re-configurable solution. In **[3]**, Szyperski defines components as "*units of composition with contractually-specified interfaces and explicit context dependencies only….that can be deployed independently and are subject to composition by third parties*". A component approach allows the software to be designed in a generic manner for ease of re-use. It also allows an application to be configured at deployment time and even re-configured at run time.

The aim of this paper is to illustrate how a component–based architecture can improve the development of configurable and re-configurable VANET routing protocols. For this purpose, we first present a component architecture that we developed for the BBR routing protocol **[11]** that operates in sparse areas. We then demonstrate how this architecture can be adapted to changing network conditions, i.e. to dense traffic areas. Subsequently, we identify the commonalities that exist among VANET routing protocols and show how our architecture can be re-used in the implementation of a broader spectrum of VANET protocols.

The rest of the paper is organized as follows: Section 2 presents a survey of the different routing protocols used in VANETs. Section 3 introduces the component model. Section 4 explains the componentization of the BBR protocol. Section 5 evaluates this approach and finally, we conclude the paper in Section 6.

2. SURVEY OF VANET ROUTING

Finding and preserving a route in VANETs is a highly challenging task. A lot of research is being invested to devise effective routing protocols in VANETs to achieve robust message dissemination. This section focuses on the results of two different surveys carried out by the authors in **[4, 5]** on the various routing categories applied for VANETs. We use these surveys to identify commonalities and differences in routing protocol behaviour. These form the basis of a common component pattern for developing protocols (as described later). Our analysis is illustrated in Figure 1; this shows categories of routing protocols (described next) and discusses features from example protocols.

Ad-Hoc routing protocols can be adaptive or not; adaptive routing is preferred for VANETs given its capability to adjust to a changing topology and reduce overhead. However, because of the highly dynamic nature of VANETs, it has low throughput.

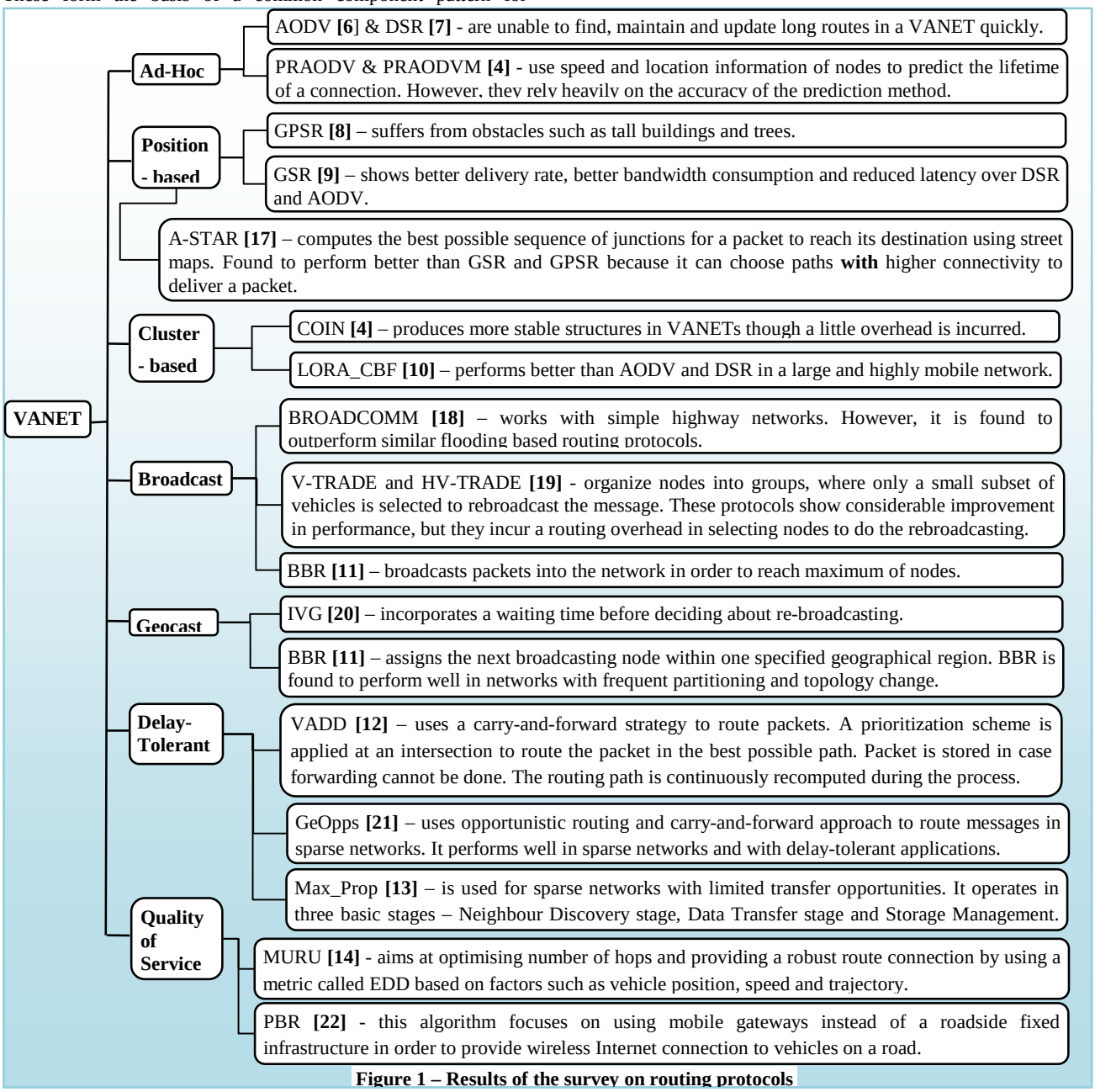

Figure 1 – Results of the survey on routing protocols

Position-based routing is well suited for VANETs since the nodes are known to move along established paths. Since routing tables are not used, no overhead is incurred when tracing a route. However, it can suffer from routing loops and network delays.

Cluster-based routing is performed in clusters. A group of nodes identifies themselves to be part of a cluster and the node designated to be the cluster-head will broadcast the packet to the cluster. Good scalability can thus be provided for large networks but network delays and overhead are incurred when forming clusters in highly mobile VANETs.

Broadcast sends a packet to all nodes in the network, typically using flooding. This ensures the delivery of the packet but bandwidth is wasted and nodes receive duplicates. In VANETs, it performs better for a small number of nodes.

Geocast is considered as a multicast service within a specific geographic region. It normally defines a forwarding zone where it directs the flooding of packets in order to reduce message overhead and network congestion caused by simply flooding packets everywhere. In the destination zone, unicast routing can be used to forward the packet. One pitfall of Geocast is network partitioning and also unfavourable neighbours which may hinder the proper forwarding of messages.

Delay-Tolerant protocols are mostly applied in sparse networks which are found mainly in rural areas but also in dense areas where there can be low concentration of vehicles especially at night time. In such cases, establishing an end-to-end route is not possible. This makes the routing of packets more difficult which is why such routing protocols need to implement a delay-tolerant aspect in order to cater for cases when there are no nodes available to forward a packet. The technique adopted is called carry-and-forward whereby a packet is forwarded only when nodes are available. Otherwise, it is simply carried.

A **Quality of Service** protocol normally guarantees a good performance through adequate resource reservation and availability of proper infrastructure. In an ad-hoc network such as the VANET, it is very difficult to provide such service unless there is a roadside infrastructure. Nonetheless, the protocols developed for VANETs estimate the stability of a route by analysing factors such as link delay, link reliability, vehicle velocity and trajectory, vehicle position and distance between vehicles; and calculate how long a route can remain connected and reduce the time required to repair a broken connection.

It is clear that there exist a range of routing requirements and network conditions in VANET applications; hence, there is potential for configuration and re-configuration of routing protocols. We revisit this survey in Section 5, identifying the commonalities and differences underlying these protocols; such commonalities form the basis of a common architecture which can be used in the development of VANET routing protocols.

3. COMPONENT MODEL

This section talks about the component model OpenCom used in our implementation. This is a lightweight model to tailor software architectures in a configurable manner to suit the requirements of a system. It also supports run-time re-configuration through reflection, enabling a high level of adaptability of the system. OpenCom employs a small runtime kernel to manage the lifecycle of the components. The authors in **[15]** provide three case studies showing the capability of OpenCom to adapt a system to different environments. Figure 2 explains the component model; a component advertises its services through an interface and binds to another component through a receptacle. An interface-receptacle binding is shown in Figure 2. The receptacle is used by B to bind to the interface of A in order to access the services provided by A. Likewise, B may also publicize its services through an interface which can be accessed by other components.

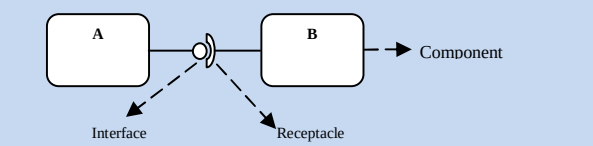

Figure 2. The Component Model

4. COMPONENTIZATION OF BBR

As a first step in our investigation, we present a component-based architecture for a protocol called Border Node Based Routing (BBR). This protocol **[11]** is designed for sparse areas and mainly uses broadcast to reach the maximum number of nodes in the network and unicast when required to send messages from one node to any other node. Its aim is to ensure a reliable message delivery with minimum delay in an ad-hoc network with low node density and high node mobility. It consists of two algorithms: Neighbour Discovery and Border Node Selection algorithms. Figure 3 shows our component architecture used to implement and execute the BBR protocol; we now explain the operation of BBR in terms of the component implementation.

The Neighbour Discovery algorithm is used to discover the neighbours of the current node and is implemented by the *NeighbourDiscovery* component. This component seeks the services of the *PacketSender* and *Timer* components in order to process the *NeighbourDiscovery* algorithm. Generally, the neighbouring nodes are found one-hop away within the radio transmission range of the actual node. All nodes advertise their presence by sending out "Hello" beacon messages periodically. The *PacketSender* component takes charge of broadcasting these beacon messages to the surrounding nodes. Since this needs to be done periodically, the *Timer* component is used to schedule the periodic transmission of these "Hello" messages. The information thus obtained is then stored into a Neighbour table by the *NeighbourDiscovery* component.

The Border Node Selection algorithm is the core process of this protocol used to designate the next node to broadcast a packet received and is implemented by the *PacketProcessor* component. This node is known as the ***border node*** as it lies furthest away within the transmission range of the source node. An example is illustrated in Figure 4. If s is the first node to broadcast a data packet in this scenario, then it is a border node by default. The circle delineates the transmission range of s (labelled as R). **C1**, **C2** and **C3** denote the transmission range of s, d and v respectively. Once s has broadcasted a data packet, the next border node needs to be identified among the neighbours of s so as to further forward the packet out of the transmission range.

The *DataPacketHandler* component is responsible for creating the data packet according to the format stated by the routing protocol. In case the packet is being broadcasted for the first time,

the *PacketSender* component will simply disseminate the packet to the surrounding network. When a data packet has been received, the *DataPacketHandler* component decides whether to carry or forward the packet. First, it uses the *Forward* component to check whether the packet received is a duplicate, in case of which, it is simply discarded. In case it needs to forward the packet, it uses the *PacketProcessor* component to trigger the *Border Node Selection* process based on the following two scenarios:

Case 1: Only 1 neighbour in neighbour list of received packet.
No border node selection is made. Instead, the *PacketProcessor* component triggers a periodic broadcast of the message *p* times through the services of the *PacketSender* and *Timer* components.

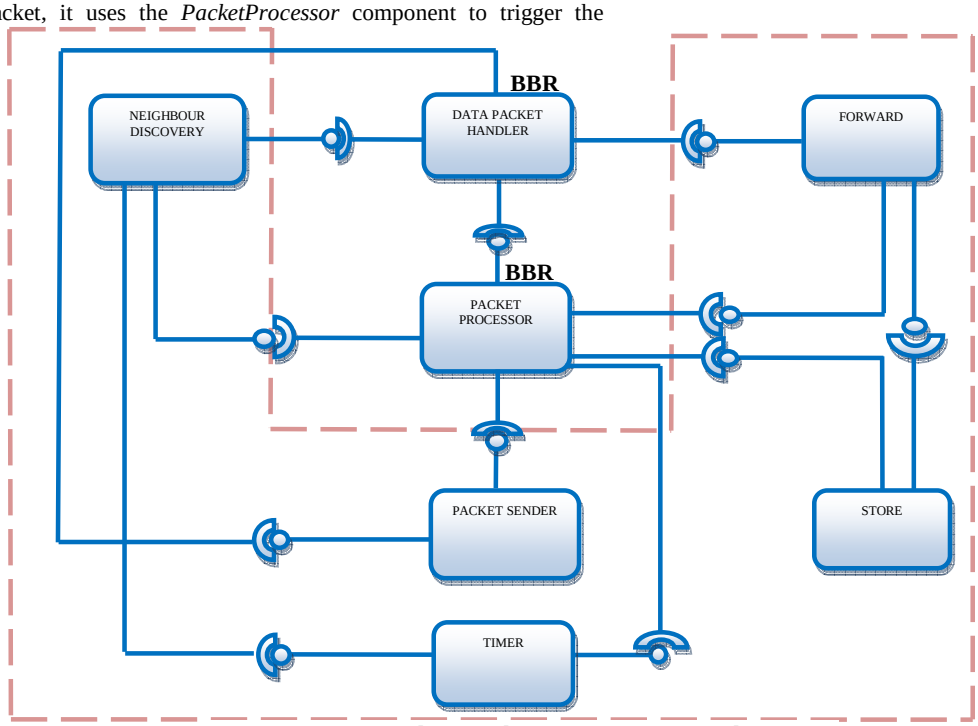

Figure 3. Commonalities in the BBR component architecture

Case 2: > 1 neighbour in neighbour list of received packet.
Border node selection is triggered and two timer processes are initiated: T_{ad} (access delay timer) and T_{max} (maximum delay timer), which are coordinated by the *Timer* component. At the end of T_{ad} timer, the node decides whether it has to rebroadcast, depending on the number of neighbours it has at that time. At the end of T_{max} timer, the node decides whether it is a border node, in that case, it re-broadcasts or stores the packet, subject to availability of neighbouring nodes.

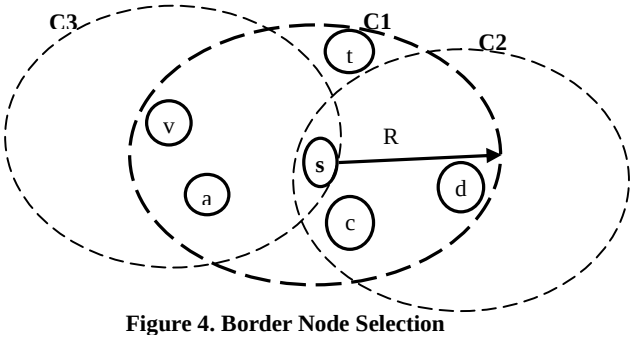

Figure 4. Border Node Selection

All nodes lying around the border of the transmission range of *s* (*d*, *v*, *t*) (see ***Figure 4***) represent the next border nodes. These have the least common neighbours (normally <= 1) compared to the neighbours of s. ***Table 1*** shows how this *Border Node Selection* algorithm is computed for s, based on information obtained from the *Neighbour* table. If the border nodes do not encounter any neighbours, the *DataPacketHandler* component will simply decide to store the packet using the *Store* component until potential new neighbours are met. Note that the *Forward* component is also responsible for maintaining the buffer table of the data packets. The *Timer* component acts as a clock and is in charge of scheduling various activities such as the broadcast of the data packets, periodic broadcast of "Hello" messages and coordinating the *Border Node Selection* process.

Table 1. Border Node Selection Process

Node	Neighbours	Common Neighbour #
s	{t, a, c, d, v}	Nil
d	{s, c}	1
v	{s, a)	1
a	{s, v, c}	2
c	{s, d, a}	2
t	{s}	0

The ***BBR*** protocol adopts a carry-forward strategy in order to minimize any packet loss when there are no nodes. Since packet loss is highly likely to occur in environments where there is low node density, providing measures to counter this problem in a sparse VANET is a fundamental task.

5. EVALUATION

The aim of this section is to evaluate our architecture for BBR at three levels. First, we evaluate how it can be configured to adapt to changing environment conditions. Second, we illustrate the re-configurability based upon the use of fine-grained components. Third, we identify the commonalities that exist among the various routing protocols presented in Section 2, and show how this architecture is re-usable. We show that the implementation of VANET routing can be geared towards a common architecture.

5.1 Configurability and Re-configurability

In the BBR protocol, when no border node is found (see Section 4), the packet received is broadcasted *p* times. This parameter *p* is *configurable* such that it can be increased or decreased to suit the given network density. Similarly, there are other parameters that affect the functioning of the routing protocol. For example, during the *Border Node Selection* process, at one point in time, nodes have to take decisions about whether they should re-broadcast a packet (T_{ad}) or whether they are the next border node to carry or forward a packet received (T_{max}). This decision is based on a given number of neighbours. These parameters can be configured in the architecture at design and run-time; e.g. the number of neighbours tested in T_{max} can be decreased in a low density so that a border node may still decide to re-broadcast.

Moreover, although the BBR component architecture is developed for a sparse network, it can be adapted to be deployed in a dense network. In this case, there is no need to carry packets given the availability of nodes, implying changes to the configuration such as removing the *Store* component and adjusting the controller parameters as mentioned above.

The *Packet Processor* component can be divided into sub-components to perform tasks such as:

1) Broadcasting a packet *p* times when no border node is selected.
2) Initiating the Border Node selection process.

In the implementation, these tasks are used with the same frequency and are encapsulated within the same component. However, in dense areas, it is possible that there will always be neighbouring nodes, implying that the second task will be executed most. This is encapsulated within a sub-component that is plugged into the system. This shows that the component architecture can be configured flexibly to integrate fine-grained components without affecting the operation of the protocol.

5.2 Commonalities in VANET Routing

We have considered 3 scenarios to identify commonalities and variabilities among routing protocols. All VANET routing methods perform sending and receiving of a packet, scheduling the execution of particular tasks or discovering the surrounding neighbours. The re-usable components that perform these functions have been enclosed within a dashed line in Figure 3 to highlight how we exploit these commonalities.

Scenario 1: Routing protocols from the same category. The category is **Broadcast;** we compare **TRADE [16]** to BBR.

TRADE: Since this protocol is similar to the BBR protocol, all the components of BBR can be re-used to adapt it to the requirements of TRADE. The latter also mentions a border node allocation process in order to broadcast a packet in the network, which is same as the BBR border node selection process. The other routing protocol in this category, BROADCOMM [**18**], also exhibits the same features as BBR, hence, implying that the same component architecture can be utilised as well**.**

Result: For protocols from the same category, a minimal change needs to be brought to the architecture such as the creation of components that process data packets as per the protocol specification. Our architecture can easily be adapted to another protocol, thus facilitating the new protocol's implementation.

Scenario 2: Protocols from different categories. The category is **Delay-Tolerant** and the protocols are **VADD [12]** and **Max_Prop [13]**.

VADD: In VADD, three different modes are applied for packet dissemination: Intersection, StraightWay and Destination. Upon analyzing VADD, it is found that its implementation consists of the following components: *NeighbourDiscovery*, *PacketSender*, *DataPacketHandler*, *Store*, *Forward*, *StreetMapData* (to obtain street map data), *IntersectionMode*, *StraightWayMode* and *DestinationMode*. These last three components show different ways of forwarding a packet at an intersection, on a straight road and at a destination respectively. If no node is found in any of these modes, then the packet is stored. This analysis shows that most of these components are already available from the given architecture of *BBR* and are re-usable. The two components labelled "BBR" in Figure 3 can be replaced in VADD in the following way: The *DataPacketHandler* component will need to be customized so that it creates packet formats for VADD. The *PacketProcessor* component can be replaced by the *Intersection*, *StraightWay* and *Destination* modes, which represent the core function of the protocol. All of these will be supplied with street map information from a *StreetMapData* component. [12] also mentions variations in the VADD algorithm, namely L_VADD, D_VADD, MD_VADD and H_VADD. Each of these has a different approach to deal with the *Intersection* mode. This means that the *Intersection* component can be further refined into sub-components. H_VADD, being a hybrid of the other versions, can be implemented as a controller to coordinate the actions.

MAX_PROP: Max_Prop operates in three basic stages (*Neighbour Discovery*, *Data Transfer* and *Storage Management*). The common components that we have identified are *NeighbourDiscovery*, *PacketSender*, *Timer*, *Store*, *Forward* and *DataPacketHandler*. However, the last two components need to be adapted: the *Forward* component must accommodate deletion of packets based on whether they have been acknowledged as delivered. The *DataPacketHandler* must be customized to the packet format of Max_Prop. The new component to be added is a *Prioritizing* component (used to allocate priority to messages before they are actually distributed). Since the functions coordinated by the latter component are complex, it may need to be refined into sub-components. For instance, one of its functions is to calculate the probability of node meetings, required in message routing. This can be separately developed as a sub-component to be used in other protocol implementations.

We repeated the scenario; this time with a different category i.e. **Cluster-Based** and the protocol analyzed is **LORA_CBF [10]**.
LORA_CBF: This protocol can be summarized in four stages: *Cluster Formation*, *Location Discovery*, *Routing* of *Data Packets*

and *Maintenance* of *Location Information*. These can in turn be mapped onto the following components (similar to those in BBR): *NeighbourDiscovery*, *Timer*, *PacketSender*, *Store*, *Forward*. The *PacketProcessor* component needs to be replaced by a new one to perform *Cluster Formation* instead of Border Node Selection.

Result: The scenario shows the potential for re-usable components. The component architecture can be configured to accommodate the variabilities of different categories of routing, combining reusable components with the addition of fine-grained components specific to individual protocol behaviour.

Scenario 3: Show different component architecture of a routing protocol from a different category, yet identifying a percentage of commonality. The category is **QoS** (Quality of Service) and the protocol analyzed is **MURU [14]**.

MURU: The functionalities of MURU are: *calculate the EDD metric*, *capture street map data*, *calculate shortest trajectory to the destination*, find vehicle location, check vehicle speed, *generate RREQ message* and *perform a pruning mechanism*. Each of these can be mapped onto individual components, among which those common with BBR are: *DataPacketHandler* when customized to properly format packets and also create RREQ messages, *PacketSender* and *Timer* (must be adapted to perform the pruning to allow a node to delay the forwarding of a RREQ message). No *Store* and *Forward* components are required.

Result: This shows that when variabilities exceed commonalities for a particular routing strategy, then there is little potential for configuration of the component architecture. However, the common architecture remains useful particularly when adapting behaviour e.g. if it was required to move from the simple BBR behaviour to more complex MURU behaviour.

6. CONCLUSIONS & FUTURE WORK

In this paper, we have highlighted the benefits of a component-based approach in the implementation of VANET routing protocols. We have demonstrated that most protocols within our categories (for example position-based, cluster-based) share common patterns and we also demonstrated how these can be mapped on to common components. The variabilities can also be mapped on to new components specific to the protocol and can be plugged into the system. The common components can be separated from the more protocol-specific ones and hence provide a common platform in the implementation of VANET routing. We have shown that, through such an approach, the implementation is configurable at deployment time. As future work, we intend to show how such component architectures can be dynamically re-configured in order to adapt the system to the ongoing changing network conditions; hence facilitating both the development and deployment of more flexible VANET routing protocols.

7. ACKNOWLEDGEMENTS

This work is funded under the CONNECT Project (http://connect-forever.eu) (FP7 Theme / ICT-2007.8.6, FET Proactive 6: ICT Forever Yours).

8. REFERENCES

[1] Liu, C. and Kaiser, J. 2005. A Survey of Mobile Ad Hoc network Routing Protocols. Tech. Report. University of Ulm.

[2] Jakubiak, J. and Koucheryavy, Y. 2008. State of the Art and Research Challenges for VANETs. In Proceedings of 5th IEEE CCNC (Las Vegas, Nevada, Jan. 2008), 912-916.

[3] Szyperski, C. 1998. Component Software: Beyond Object-Oriented Programming. Addison-Wesley.

[4] Li, F. and Wang, Y. 2007. Routing in vehicular ad hoc networks: A survey. IEEE Vehicular Technology Magazine. 2, 2 (June 2007), 12-22.

[5] Bernsen, J. and Manivannan, D. 2009. Unicast routing protocols for vehicular ad hoc networks: A critical comparison and classification. Pervasive Mob. Comput. 5, 1, 1-18.

[6] Perkins, C. and Royer, E. 1999. Ad-hoc on-demand distance vector routing. In Proceedings of 2nd IEEE Workshop on WMCSA (New Orleans, LA, Feb. 1999). 90-100.

[7] Johnson, D. and Maltz, D. 1996. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*. Kluwer Academic Publishers. 153-181.

[8] Karp, B. and Kung, H. T. 2000. GPSR: greedy perimeter stateless routing for wireless networks. In Proceedings of the 6th MobiCom Conference (Boston, MA, August 2000). 243-254.

[9] Lochert, C., Hartenstein, H., Tian, J., et al. 2003. A routing strategy for vehicular ad hoc networks in city environments. In Proc. Intelligent Vehicles Symposium (Traverse, MI). 156-161.

[10] Santos, R., Edwards, A., Edwards, R., and Seed, N. 2005. Performance evaluation of routing protocols in vehicular ad-hoc networks. Int. J. Ad Hoc Ubiquitous Comput. 1, 1/2, 80-91

[11] Zhang, M and Wolf, R. 2007. Border Node Based Routing Protocol for VANETs in Sparse and Rural Areas. IEEE Globecom Autonet Workshop (Washington DC, Nov. 2007). 1-7, 26-30.

[12] Zhao J. and Cao, G. 2006. VADD: Vehicle-assisted data delivery in vehicular ad hoc networks. In Proceedings of 25th IEEE International Conference on Computer Communications (Barcelona, Spain, April 2006). 1-12.

[13] Burgess J., Gallagher, B., Jensen, D. and Levine, B. 2006. MaxProp: Routing for vehicle-based disruption-tolerant networks In Proceedings of 25th IEEE International Conference on Computer Communications (Barcelona, Spain, April 2006).

[14] Mo, Z., Zhu, H., Makki, K. and Pissinou, N. 2006. MURU: A multi-hop routing protocol for urban vehicular ad hoc networks. In Proceedings of 3rd International MOBIQUITOUS Conference (San Jose, CA, June 2006), 1-8.

[15] Coulson, G., et al. 2008. A generic component model for building systems software. ACM Trans. Comput. Syst. 26, 1-42.

[16] Sun, M., Feng, W., Lai, T., et al. 2000. GPS-based message broadcast for adaptive inter-vehicle communications. 52nd IEEE Vehicular Technology Conference. 2685-2692.

[17] Liu,G., Lee, B, Seet, B., et al. 2004. A Routing Strategy for Metropolis Vehicular Communications. Proc. International Conference on Information Networking (ICOIN), February, 2004.

[18] Durresi, M., Durresi, A., and Barolli, L. 2005. Emergency Broadcast Protocol for Inter-Vehicle Communications. In Proc. 11th International ICPADS Conference. Workshops v2, 402-406.

[19] Sun, M., Feng, W., Lai, T., et al. 2000. GPS-Based Message Broadcasting for Inter-vehicle Communication. In Proceedings of the 2000 international Conference on Parallel Processing.

[20] Bachir, A. and Benslimane, A. 2005. A multicast protocol in ad hoc networks inter-vehicle geocast, In 57th Vehicular Technology Conference. 2456-2460.

[21] Leontiadis, I. and Mascolo, C. 2007. GeOpps: Geographical Opportunistic Routing for Vehicular Networks. In Proc. WoWMoM (Helsinki, Finland, June 2007). 1-6.

[22] Namboodiri, V. and Gao, L. 2007. Prediction-Based Routing for Vehicular Ad Hoc Networks. IEEE Trans. Vehicular Technology. 56, 4. 2332-2345.