A Semantics-based Multi-agent Framework for Vehicular Social Network Development

Xiping Hu[1], Jidi Zhao[2], Dizhi Zhou[3], Victor C. M. Leung[1]
[1]Dept. Electrical and Computer Engineering, The University of British Columbia, Vancouver, Canada V6T 1Z4
[2]Institute of System Engineering, Shanghai Jiaotong University, Shanghai, China 200052
[3]Faculty of Computer Science, University of New Brunswick, Fredericton, Canada E3B 5A3
xipingh@ece.ubc.ca, judyzhao33@gmail.com, a0kkh@unb.ca, vleung@ece.ubc.ca

ABSTRACT
This paper proposes a semantic-based multi-agent framework to support development of vehicular social network applications. In the programming model of the framework, software platforms of vehicular social network systems can be developed by the collaboration of mobile agents and service (or resident) agents, where resident agents provide application services on devices and mobile agents provide communication services on behalf of owner applications. On top of the device infrastructure, the architecture of the proposed framework consists of three layers: framework service layer, software agent layer and application layer, to fully support dynamic and collaborative tasks of vehicular social networks. The multi-layer architecture design of the framework fully supports self-adaptive applications in vehicular social network environments, and is readily extensible to support new features. Developers can easily and effectively develop diverse applications for the vehicular social networks.

Categories and Subject Descriptors
C.2.4 [**Computer Systems Organization**]: COMPUTER-COMMUNICATION NETWORKS - Distributed Systems - *Distributed applications.*

General Terms
Algorithms, Design, Economics, Human Factors, Languages, Theory.

Keywords
Vehicular social networks, software agent, semantic, service, multimedia transmission, application development.

1. INTRODUCTION
Vehicular ad hoc networks (VANETs) are a special form of mobile ad hoc networks (MANETs), which enable communications among nearby vehicles, i.e., vehicle-to-vehicle (V2V), and between vehicles and nearby roadside infrastructure, i.e., vehicle-to-roadside (V2R). Every day, a large number of people in areas of dense population spend hours travelling along the same routes at the same time on their commute to/from work. Their travel patterns are highly predicable and regular. Consequently, there is an opportunity to form periodic virtual mobile communication networks between these travelers or their vehicles, which can be referred as vehicular social networks [1]. Furthermore, research has shown that knowledge of nodes' social interactions can help improve the performance of mobile systems [2, 3], thus applications of vehicular social networks include many fields. There are three general types of applications over vehicular social networks: (a) Safety improvements, such as applications that improve the safety of the passengers on the roads by notifying the occupants of vehicles about any dangerous situation in their neighborhood [4, 5, 6]; (b) Traffic management, applications that provide users who join the vehicular social network with traffic information enabling them to improve the traffic efficiency and driving behavior [7]; (c) Entertainment, applications that enable the drivers and passengers to share multimedia files in the vehicular social network [8].

Because of the high speed of vehicles' movements, the topology of a VANET is highly dynamic, and connectivity status of the VANET could also change frequently, which results in dynamically changing environments and dynamic service requirements of vehicular social networks. Thus, it is a challenge to make the applications in vehicular social networks more conscious of the dynamically changing environments. Meanwhile, wireless links in a VANET tend to be unreliable and have short lifetimes; consequently it is challenging to provide desirable vehicular social network applications to users (i.e., drivers and passengers), as well as efficient multimedia transmission techniques tailored for the context of vehicular networks.

A software agent can be seen as a composition of software and data, which is able to work autonomously. It can also transport its state from one environment to another with its data intact, and be capable of performing appropriately in the new environment. When an agent decides to move, it can save its current state, transfer this saved state to a new host, e.g., a mobile device, and then recover from the saved state [9]. Mobile agents are software agents that act on behalf of their creators and move independently among hosts [10]. After being dispatched, a mobile agent becomes independent of the process that created it and can operate asynchronously, reacting dynamically and autonomously to environmental changes, and requiring little interaction with users. Furthermore, mobile agents have the capability to coordinate among themselves, resulting in collaborating services and resources among the devices, e.g., in a vehicular social network. Consequently, mobile software agents are very suitable to execute applications for vehicular social networks.

Dynamism is an intrinsic feature of VANETs. Traditional approaches require the mobile agent to know the availability and

DIVANet'11, November 4, 2011, Miami, Florida, USA.

every detail of the services in advance. This usually limits the flexibility of applications based on a service-oriented architecture, particularly in such a dynamic environment. The Semantic Web was first introduced by Tim Berners-Lee as an evolutionary extension of the World Wide Web, in which a well-defined meaning of Web content is provided through semantic markup, making it more machine interpretable [11]. Through W3C led initiative, the Semantic Web touts technologies like Resource Description Framework (RDF) [12], Web Ontology Language (OWL) [13] and Rule Interchange Format (RIF) [14] as knowledge representation formalisms for the semantic markup. These semantic based technologies have been experiencing intensified interest in the Internet, such as Web 2.0 and Web 3.0, and in different application domains such as matchmaking in Web services [15], classification of genes in bioinformatics [16], multimedia annotation [17], and travelling planning [18]. However, to the best of our knowledge, few researchers have investigated the possibility of and challenges of their applications in a vehicular social network.

There is a broad range of infrastructure that can support vehicular social networks [19, 20]. For example, drivers or passengers may carry smart phones with various forms of wireless communication capabilities (e.g., 3G cellular, WiFi, WiMAX), or vehicles may be equipped with embedded computing systems, which support V2V and V2R communications, with deployed roadside infrastructures in the latter case, using Dedicated Short Range Radio (DSRC) communications. In order to develop a vehicular social network system, we should first develop a VANET connecting the hardware devices onboard vehicles, and determine its specifications. The second step is to develop a software platform on which to develop and install different applications that work in the vehicular social network. In this paper, we only consider the software development for vehicular social networks. We consider the following three main goals for developing a software platform for vehicular social networks: to support high-level application programming, to provide a systematic approach and model of vehicular social network applications, and to support easy and effective application development for vehicular social networks.

In this paper, we propose S-Aframe, a semantic-based multi-agent framework for vehicular social networks. It is based on a multi-layer architecture built on existing operating systems, and implemented using Java and AmbientTalk [21]. S-Aframe supports easy and effective application development for vehicular social networks. It supports applications that are self-adaptive in a dynamically changing environment, dynamic application services, resources deployment, and communications among agents in a coordinated manner. Application programmers can easily develop their application services by extending S-Aframe's services.

The outline of the rest of the paper is as follows. Section 2 introduces the programming model, architecture and services of S-Aframe. Section 3 analyzes the challenges of developing vehicular social network applications, and describes how they are met with S-Aframe. Section 4 shows an application example of vehicular social networking based on S-Aframe. Section 5 provides comparisons and evaluation of S-Aframe against other projects. The paper ends with conclusions and prospects in Section 6.

2. OVERVIEW OF S-AFRAME

2.1 Programming Model of S-Aframe

The programming model of S-Aframe is shown in Figure 1. Each node contains the framework services, which are the generic services of S-Aframe. Meanwhile, S-Aframe provides a series of Java application program interfaces (APIs). Based on these Java APIs, programmers can use Java to develop resident agents, mobile agents, owner applications and specific application services according to different application scenarios of vehicular social networks. Table 1 shows the Java APIs of S-Aframe.

In general, the architecture of conventional agent systems depends on specific applications. In those systems, mobile agents need to contain all the relevant codes when they want to accomplish some tasks, although some of the codes may never been used. It will raise network overhead when mobile agents transfer such data. On the other hand, programmers have to develop a new mobile agent completely when implementing a new application of vehicular social network; even both the new application and former applications are closely related and use similar application services. In this case, programmers have to spend much time developing different applications for vehicular social networks, resulting in low developing efficiency. Thus, in S-Aframe, we use resident agents to provide all the application services to mobile agents.

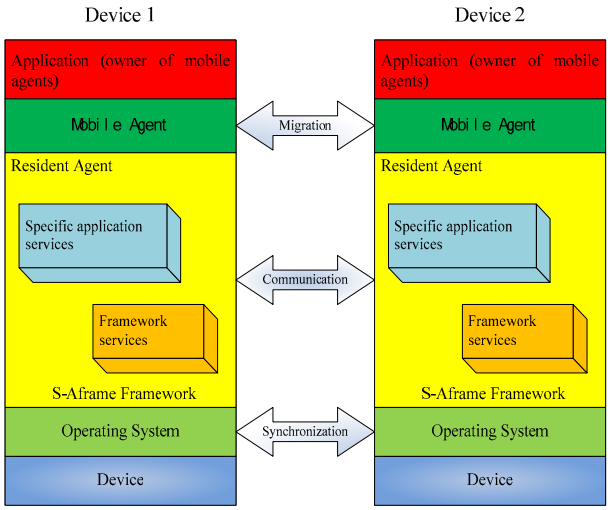

Figure 1. Programming model of S-Aframe

Mobile agents can automatically migrate around vehicular social networks with their state and execution results. They can dynamically use different application services provided by resident agents on local nodes to accomplish specific application tasks. Mobile agents are created by applications. An application, as the owner of a mobile agent, can send the mobile agent to travel in an underlying VANET and retrieve it back to the application's node.

2.2 Architecture of S-Aframe

The S-Aframe architecture has four layers: Framework Services Layer, Resident Agent Layer, Mobile Agent Layer and Application Layer. The architecture is shown in Figure 2.

Table 1. Java APIs of S-Aframe

Java APIs	Introduction
public String executeService();	Execute application services
Public void getApplicationSpecificMobileAgent(String key, String type);	Initialization of mobile agent
public void setAppName(String appName);	Configure the ID name of mobile agents' owner application
public void setOwner(String name);	Configure the ID name of the node
public void execute();	Executing mobile agent
public void transitToNextNode (boolean clean, boolean takenAway);	Migrating to next node
public void setTransitionSet(ArrayList <String> set);	Configuration of the subset of migration nodes
public void setTransitionMode(String mode, Sequential);	Configuration of migration mode
public String getResidenceAgentName();	Getting ID name of local node
public String getCurrentTime();	Getting current time of local node
public HashMap<Object, Object> getAgentList();	Classify the nodes
public void sendBinData(String targetID, File sourceFile);	Transmission of common files
public void sendBinDataIndirect(String targetID, File sourceFile);	Transmission of large files

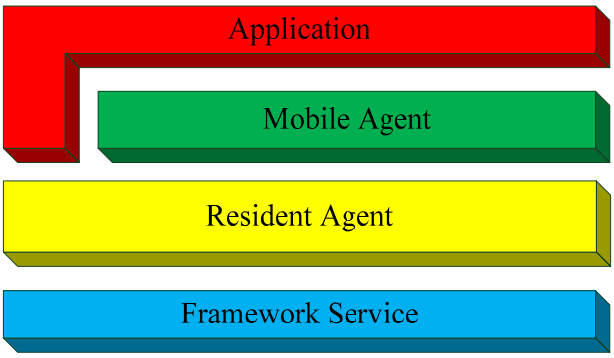

Figure 2. Architecture of S-Aframe

A. Framework service layer

At the bottom of the framework, the framework service layer provides the core functions and generic services to the upper layer, so as to help applications of vehicular social networks self-adapt to dynamically changing environments, such as adapting to dynamic network topology as well as dynamic application services requirements of the users in the vehicles. The framework services are only accessed by the resident agent layer but not by other layers. The resident agent layer can invoke any services from this layer, and then provide them to mobile agents. More details about the services in this layer will be given in Section 2.3.

B. Resident agent layer

Resident agents provide all the local application services to visiting mobile agents. A resident agent can be automatically deployed to a new mobile node when it is joining the underlying vehicular network. Once a resident agent is deployed onto a mobile node, it will stay there to provide application services. Mobile agents can directly use all the services which the resident agents provide. Application services provided by the resident agent layer can be built on the services provided by the framework services layer. Framework services only implement basic and generic functions and services for applications. Application developers of S-Aframe can develop new application services and deploy them to this layer. Thus the services provided by this layer contain two parts: services directly provided by the framework services layer and specific application services developed by application developers.

C. Mobile agent layer

Mobile agents run on top of resident agents and are used to execute different application services provided by resident agents. They do not contain different application services in their code. All the services which a mobile agent needs to use come from resident agent layer, such as the services for specific application, communication, and migration, etc. It just contains some basic information, such as its migration strategy, processing strategy of executing results, as well as computation and communication results. Meanwhile, mobile agents are also used to transfer necessary resources or application services to some mobile devices when they do not contain such resources or services.

D. Application layer

Applications are owners of mobile agents. An application provides the user interface to its users who use mobile devices in their vehicles. Using the user interface, the user can select and release a mobile agent to execute a specific application of the vehicular social networks, or release multiple mobile agents to accomplish different applications simultaneously. S-Aframe supports multiple agents with multiple application owners working at the same time.

2.3 Services of S-Aframe

As mentioned in the previous section, the framework service layer contains all the generic services (framework services) of the S-Aframe framework. The following are more detailed descriptions about the services in this layer.

-*ID name service*: When there are multiple agents and multiple applications working at the same time in a vehicular social network system, naming collision will happen. In this situation, a unique ID name of each mobile node is very important. Using this service, once the resident agent is initialized in a mobile device in the vehicle, it will generate a universally unique identifier (UUID) as ID name to each mobile device. Upper layers can read this information but not configure it. Meanwhile, in S-Aframe, according to some parameters, such as the strength of connections among the users, their common behaviors in social interests as well as the frequency of recurring encounters, the nodes in the

local vehicular social network system will be classified to different groups by using semantic technique.

-*Data fetching and storage service*: In order to reduce network overhead and improve application efficiency in the vehicular social network systems, S-Aframe divides data into two types: shared and non-shared. The shared data only contains the local ID name and available services of local device; resident agents will share this information to the vehicular social network system. On the other hand, non-shared data contains code of agents, specific application services, existing data in local devices and executing result of mobile agents, etc. Meanwhile, when programmers develop mobile agents, they can decide whether mobile agents should store the executing result in local devices, and what data mobile agents need to carry away to the next migrating devices.

-*Network status service*: This service makes use of the network information provided by AmbientTalk. In AmbientTalk, every node can listen to the whole local network to determine how many nodes are currently available in the local vehicular social network system. Also, resident agents can share all the IDs of currently connected nodes, as well as available services of local devices. By using this service, the framework service layer can inform the upper layer of the latest status of the vehicular social network system, such as the ID names list and available services list. Then agents can adapt to the dynamically changing environment. Meanwhile, users of vehicular social network systems can access the current available services from user interfaces based on this service.

-*Migration service*: It provides the basic migrating environment to the upper layer. Resident agents directly invoke this service and provide it to mobile agents. Using this service, mobile agents can migrate around the nodes in a vehicular social network. In S-Aframe, we provide three migration strategies for application developers to develop mobile agents: (i) migrating among all the mobile nodes in the local vehicular social network; (ii) migrating among some of the mobile nodes. For example, a mobile agent only migrates around some nodes of a defined group in the local vehicular social network; (iii) only migrating to a defined mobile node. Consequently, based on the latest ID name list and available services list from S-Aframe, developers can flexibly select a migration strategy when they are developing mobile agents to accomplish some specific applications. Combined with semantic technique, mobile agents can also choose dynamically an efficient migration sequence according to heterogeneous situations.

-*Time service*: In some situations, like a roadway, when a user (e.g., passenger and driver) wants to join the local vehicular social network system and start some applications in a defined period, temporal information is very important. Therefore, using the time service, the vehicular social network system can automatically release mobile agents to execute any applications that the system may provide at the scheduled time. Moreover, when a mobile agent is migrating to another device, it can get the current local time by invoking this service through the resident agent. On the other hand, we can test the time efficiency of each application in a vehicular social network by invoking this service.

-*Deployment service*: Since the network topology of a vehicular network is dynamic and frequently changes, every mobile node can join and leave the network anytime and anywhere. Using this service, S-Aframe can dynamically and automatically deploy the core functions and services to a new mobile node when it joins the local vehicular social network. At the same time, if a mobile device in the underlying vehicular network does not have the necessary resources or services, mobile agents can also automatically transfer the necessary resources or services between mobile nodes by using this service through the local resident agents.

-*Dynamic invocation service*: In a vehicular social network system, diverse users usually have different services requirements, and the service requirements may change during the mobile agents' runtime. Meanwhile, because the dynamism of VANETs, it is possible that the changing identities of the users in the local vehicular social network system result in new services requirements. In S-Aframe, we incorporate semantic techniques in this service, so as to support mobile agents dynamically and intelligently invoke different application services in different mobile nodes according to practical scenarios, such as users' identities and interests.

-*Multimedia transmission service*: Due to the large data stream of multimedia (e.g., video and audio), as well as unstable links with short lifetime of a vehicular network, an efficient transmission strategy for multimedia transmission is very important in a vehicular social network system. Thus, S-Aframe combines with multipath transmission techniques and provides flexible transmission strategies in this service. Based on this service, mobile agents can automatically and dynamically choose appropriate strategies when the users want to transfer (or share) multimedia in the vehicular social network systems.

3. CHALLENGES/IMPLEMENTATION

S-Aframe is designed to satisfy the unique challenges of vehicular social networks, such as supporting applications self-adaptation in dynamically changing VANET environment, providing desirable services to users of vehicular social network systems, as well as providing efficient multimedia transmission techniques. S-Aframe works as a software platform of vehicular social network systems on which to deploy diverse applications. Furthermore, S-Aframe aims to provide a systematic development approach and programming model to developers, so as to enable them to effectively develop applications for vehicular social network systems. In this section we will first present and analyze the design challenges of vehicular social network systems, and then describe how to implement S-Aframe so as to address these challenges.

3.1 Dynamic Changing Environment

VANET environments are characterized by their high dynamism. The set of vehicles present in such an environment is constantly changing, and consequently, the set of application services offered by the devices in vehicles are varying as well [4, 22]. Therefore, it is a challenge to let applications of vehicular social network systems self-adapt to a dynamically changing environment. In S-Aframe, we mainly consider two situations when mobile agents are executing the applications while the environment of vehicular social network systems changes:

- Mobile nodes get disconnected when mobile agents are migrating around the local vehicular social network systems.
- New mobile nodes join the local vehicular social network systems when mobile agents are executing applications.

In S-Aframe, the framework service layer can provide the latest status of the vehicular social network system to the resident layer. At the same time, there are two agent cooperation mechanisms in S-Aframe: cooperation among resident agents and cooperation between resident agents and mobile agents. By the cooperation

among resident agents, they can provide the latest network status to mobile agents. With the cooperation between resident agents and mobile agents, mobile agents become more flexible since resident agents provide sufficient application services from local nodes to support mobile agents' application tasks. Thus, by invoking the network status service of S-Aframe, developers can get the latest ID name list and the latest available services list. Based on these lists and migration service, developers can flexibly select a migration strategy, define a migration list and get a primary identification of service requirements when they are developing mobile agents to implement applications of vehicular social networks.

Therefore, for the first situation, there are two main issues: (a). which mobile agents are migrating to a mobile node that gets disconnected and (b) which mobile agents are currently visiting a mobile node that gets disconnected. In S-Aframe, a mobile agent can obtain the latest ID names list of the currently connected nodes from the resident agent when it migrates to a new node. We also use the strategy that a mobile agent can store the list of the nodes which it has visited before. Thus, the mobile agent can compare the mobile agent's migration list, latest ID names list and history list every time when it is migrating to a new node, to decide the next migrating target automatically. Meanwhile, because the migration time of a mobile agent from one node to another is very short, the probability that the next visiting target node will get disconnected during a mobile agent's migration is low. Moreover, if a node which a mobile agent is currently visiting gets disconnected, we use the strategy that mobile agent's owner application will wait for some pre-assigned time, which is decided by the developer. When the time runs out, the owner application can automatically release a new mobile agent. Consequently, in S-Aframe, mobile agents can self-adapt when nodes get disconnected as they are executing applications in a vehicular social network system.

On the other hand, connectivity situations will change and new services requirements will arise when new nodes join the vehicular social network system. As mentioned above, mobile agents can get the latest connectivity status when they migrate to a new node and automatically decide the destinations. Therefore it will not impact mobile agents' migration when new nodes join the vehicular social network systems and the connectivity situations change. The main issue for the second situation is the varying services requirements. We first assume that all the nodes have the initial executing environment of S-Aframe when they join the vehicular social network system. Since S-Aframe provides the deployment service, we use the strategy that when a new node joins the vehicular social network system, it will automatically send a request to other nodes by using AmbientTalk [8]. Once an existing node of the system gets the message, it can automatically deploy the framework service and the resident agent to the new node. After that, as previously mentioned, based on the framework service, the new node can get the latest status of vehicular social network system, such as the latest ID name list of the users and latest available services list. Therefore, based on these services, when mobile agents are migrating to a new node which just joined the vehicular social network system, and that node does not contain the necessary services or resource for mobile agents' task, mobile agents can automatically transfer the services and resources to it from other nodes through the local resident agent, and thus help resident agents extend application services. Meanwhile, a user of new node can release a mobile agent to collect the necessary services and resources. Moreover, in S-Aframe, mobile agents can also share their messages through resident agents when they are cooperating to perform a task, so as to finish the task faster.

3.2 Semantic Technique in S-Aframe

Dynamically changing environment and topology are intrinsic features of a vehicular social network. Therefore, the ability to be adaptive is critical for its performance and flexibility. We introduce semantics and semantic web technologies into vehicular social networks and explore the integration of semantics and services. By introducing semantics into such networks, we try to determine the meaning of various data and information, support corporations between mobile agents in vehicular social networks, and then help mobile agents automatically discover which services are available and learn what they should do in such dynamic environments. We propose semantic services using semantic techniques as the services part of the S-Aframe. We design and define an ontology for describing services, which we call a service ontology and then we propose a system architecture for applications using the service ontology.

A. The service ontology

What is the name of a service? What is the service about? Where does the service come from? And what can the service do? These are the questions naturally arising from end-users who want to make use of a service. We first define a general service ontology to address the above questions and then use it as a standard to follow when defining a service instance.

A service identity is a unique name or Uniform Resource Identifier (URI) for the service. We define the class Service and specify a service identity as an instance of the class. We also define the class Operation to represent the set of different operations including Input and Output.

Basic properties for the service ontology include:

1. *hasName*, this property describes the service name of a service. The domain of this property is the class SERVICE and its range is a string value.
2. *hasDescription*, the value of this property is a string that specifies any description of a service. This property is optional.
3. *hasProvider*, the property specifies the creator of a service and the value is a string.
4. *hasOperation*, this property specifies service operations. The value of this property is an instance of the Operation class. The service operation instance must have a value for the hasInput property and the value must be an instance of the Parameter class. This Parameter instance has a value for the hasObjType property which resolves to the Input class. The service operation instance must also have a value for the hasOutput property and the value must be an instance of the Parameter class. This Parameter instance has a value for the hasObjType property which resolves to the Output class.

B. The system architecture of semantic service

The system architecture of the semantic services based approach is shown in Figure 3. Mobile agents access semantic services through the API/query transformation component and the API/RDF transformation component. The service interface definition must be an instance of the service ontology. Note that, except for the query and the output, all the other communications in the diagram are in RDF graphs or OWL files.

A mobile agent can perform two kinds of queries for a semantic service. First, it gets the input parameters if any. A mobile agent submits a query through some Java API which is then transformed to a form in some semantic representation language, for example, SPARQL. The transformed query includes one or more service attributes: service identity, service name, service description. The system searches the service interface definition. If its inference engine finds one or more services, it resolves the value of its hasOperation property to the Input class and returns the class to the mobile agent. Most existing ontology reasoners such as Pellet, Racer, KAON2, and OOjDrew can be adapted and integrated into this architecture. Second, it gets the output. A mobile agent configures an instance of the Input class and constructs and submits a query. The system then performs reasoning on the knowledge base and returns an instance or instances of the Output class as the outcome.

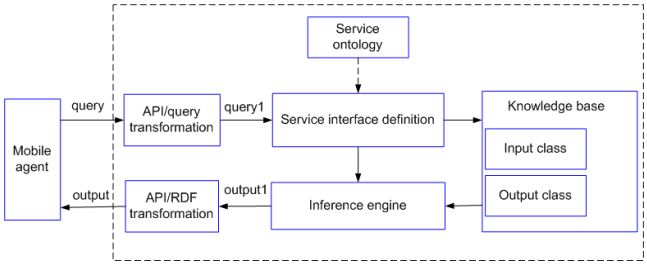

Figure 3. System Architecture of Semantic Services

3.3 Multimedia Transmission Strategy

The amount of multimedia service in mobile market is growing quickly in recent years. It is a big challenge for S-Aframe to provide efficient, stable transmission service to multimedia application with quality-of-service (QoS) assurance in highly dynamic vehicular social network. Thus, two multipath methods will be introduced in this section which can be seen as two approaches to implement the multimedia transmission service. We will also discuss the upcoming issues when introducing those methods in vehicular social networks.

The first class of multipath transmission method is called peer-to-peer (P2P)-like transmission. In this approach, a subset or all of mobile devices receive data from base station and share the portion they get with other mobile devices in vehicular social network by short-distance radio, such as Wi-Fi. The typical application scenario of this approach is video streaming, in which several mobile devices are requesting the same video data. Thus, each of them can receive a portion of video and sends their own portion to others [23, 24]. By this way, each user does not need to download all video data so that it can decrease the traffic in the wireless network and aggregate the bandwidth of all P2P nodes. Usually, it needs a control node in such a cooperative community so that this control node can allocate the download mission for each P2P nodes and manage the join and leave event in this P2P network. However, different from the control node in traditional P2P network, which is usually a server with a public IP, the control node in vehicular social network must be deployed in ordinary mobile node because of highly dynamic topology. In addition, once this control node leaves the vehicular social network, the control function must be transferred to other available node immediately (precisely, normal P2P nodes will periodically communicate with the control node; thus, if we can migrate the control function to other nodes and notify the new control node's ID to others within this period, there is little harm on the whole P2P network. This time varies from several seconds to several minutes, depending on the implementation).

Therefore, how to use current service to setup and maintain such s control function is a key challenge for using P2P-like multimedia transmission scheme in vehicular social network. One possible solution is to use the migration service to setup the control function. For example, when the mobile user wants to see a movie online, the S-Aframe first sends a request message to control node asking whether there is any existing P2P network containing such a movie. Once it gets the reply from the control node, it will join this P2P network. In addition, the control node will also re-assign the download mission among related nodes who request the same video. On the other hand, once one P2P node fails to connect to the control node, while it can still detect other nodes (means that it is still in the network), this node will send a P2P control node searching message to other P2P node and this message will be transmitted one by one using migration service. Specifically, each middle node will add information of its capability (such as signal strength, computing ability) and ID to this message and transmit it to the next available P2P node which can be determined by checking the left nodes on the local ID list. Once the last node receives this message, it will choose a best node that has biggest capability, and transmit a control request message to it. Therefore, using migration service, we can achieve the P2P control function reassignment once the original control node leaves the network. However, the performances of such scheme still need to be tested in the real network.

The second class of multipath transmission method is called pure relay. In this approach, intermediate mobile device acts as pure relay node to destination mobile device [25, 26]. For example, in one vehicular social network, mobile device A sends a video to mobile device B. Mobile device A can setup a temporal relay relationship including intermediate mobile devices C and D who can receive and forward video data to B. Those mobile devices do not need to exchange different portion of video data as they do in P2P-like transmission approach. The key technical issues in this approach include how to find the relay node, what is the principle of assigning the packet delivery mission to each relay node, and what are the motivations for other mobile device to relay packet for others although they are located in one vehicular social network. Another way to consider this relay reassignment issues is to consider relay as a function and can be transferred to other nodes using migration service. This scheme is very similar to what we do for P2P control node migration. However, the migration duration in this case is stricter than it in P2P-like transmission because of potential high packet loss rate.

Obviously, P2P-like transmission and pure relay transmission complement with each other. P2P-like transmission aims to improve the performance of multimedia service from public server to vehicular social network. On the other hand, pure relay transmission is targeted to enhance the multimedia QoS between different nodes within vehicular social network. Therefore, these two schemes can be combined together to achieve the maximum performance gain.

4. APPLICATION EXAMPLE

With the development of mobile social network techniques, more forms of such technology are used in VANETs. For example, Toyota Motor Corporation (TMC) had proposed a private social network system for Toyota customers and their cars named 'Toyota Friend' which works in vehicular network environment [27]. As discussed above, S-Aframe is designed to address the

unique challenges of vehicular social networks, such as the cooperation between agents in VANETs to improve the efficiency of performing various tasks. Especially due to the high dynamism of VANETs, the time efficiency is very important. Also, we can imagine that in a local social network, one major requirement is a common picture about the positions of the people, as well as some relative information of them. Therefore, we designed and developed a demonstration application of vehicular social network with S-Aframe about searching friends.

```
1: public transitionService (target, owner, appName, Type, executedList, takenAway, mode)
2: //1 targetName; 2 owner ID; 3 appName; 4 mobileAgentType; 5 mobile agent history;
3: //6 takenAway data; 7 migration strategy;
4: public HashMap<Object, Object> getAgentList();
5: //return the hashmap contains the IDs of all nodes in vehicular networks
6: public class NameAndTimeService implements IApplicationService {
7: //get name(or identity) and local time
8: public String executeService() {
9:   return "application service about get time and name";
10: def mobileAgentGetNameAndTimeService := /.aframe.generalMobileAgent.new();
11: mobileAgentGetNameAndTimeService.getApplicationSpecificMobileAgent
12:  ("nameAndTimeService");
13: application.addMobileAgent
14:  ("nameAndTimeService", mobileAgentGetNameAndTimeService);
15:  //invokes the application services and provides to mobile agent
16:   }
17: }
```

Figure 4. Portion of the resident agent

```
1:  public class PositionlistMobileAgent {
2:  public void launch() {
3:     mobileAgent.execute();
4:     mobileAgent.transitToNextNode(true, true); // which mobile agent should travel through
5:                          }
6:  public void init(String owner, String mode, String appName,
7:                Collection<String> subset) {
8:    mobileAgent.setOwner(owner); // id name of the mobile agent's owner
9:    mobileAgent.setTransitionMode(mode); // migration strategy of mobile agent
10:   mobileAgent.setAppName(appName); // application name which releases this mobile agent
11:   mobileAgent.setTransitionSet(subset); //only valid when node parameter set to "subSet"
12:       }
13:  }
```

Figure 5. Portion of the mobile agent

For this application, we first developed a resident agent to provide all the necessary application services at a local node to a visiting mobile agent. In order to help the visiting mobile agent dynamically and self-adaptively migrate around the underlying vehicular network, the resident agent provides the migration service and the network status service. These two services can be directly invoked from S-Aframe's framework service layer. Also, in a vehicular social network scenario, the identities of users and timestamps of them when they participate in the systems are also very important. Thus, the resident agent also invokes two other framework services: ID names service and current time service. By using these two services, the resident agent provides the name or identity of the local node to the visiting mobile agent, as well as the time stamp when the owner of the device or node is found. On the other hand, because the framework service layer does not provide device position service, thus we developed a position service in the resident agent to provide the position service to the visiting mobile agent. Second, we developed the mobile agent and its owner application for this application. The mobile agent's task is to automatically collect the names (or identities), positions and the local time of each mobile device in the underlying VANET by visiting each of them one by one. The application creates and sends the mobile agent to execute the task in the underlying VANET and display the devices' positions and some related information on the Google Map. Figure 4 shows a portion of the resident agent program. Figure 5 shows a portion of the mobile agent program. Figure 6 shows a portion of the owner application.

```
1:  public class App {
2:    public PositionlistMobileAgent getPositionlistMobileAgent() {
3:      return positionlistMobileAgent;
4:    }
5:  public void setPositionlistMobileAgent(PositionlistMobileAgent
6:      positionlistMobileAgent) {
7:      this.positionlistMobileAgent = positionlistMobileAgent;
8:  }
9:  public void setResidenceAgent(IResidenceAgent residenceAgent) {
10:     this.residenceAgent = residenceAgent;
11 :  } // resident agent is an object offering application services
12: public void launch () {
13:    mobileAgent.execute():
14:    mobileAgent.transitToNextNode(true, ture);
15:  } //launch the application
16: public void mobileAgentComeBack(String key, String data) {
17:     String[] array = data.split("#takenAwayDataSplit#");
18:     String text = "mobile agent executing result: \n";
19:     for (int i = 0; i < array.length; i++) {
20:       String line = array[i];
21:       text = text + line + "\n";  //this method is invoked by mobile agents, implement
22:                                   //it to process result taken back by mobile agents
```

Figure 6. Portion of the owner application

In the experiment, we used two laptops and one Android cell phone (Motorola-Milestone) for testing, all of which contain GPS components to get their local positions. Firstly, we deployed the resident agent and the services of S-Aframe to two of the mobile devices. Then we assumed that a new user named Judy using the laptop to register in the local vehicular social network system. Once Judy had registered in this system, as mentioned above, the former mobile devices can automatically deploy the agents and generic services to her laptop by using the deployment services of S-Aframe. From the user interface of her laptop, Judy selects some service to collect the information about other users in the same group of the local vehicular social network system, then S-Aframe will automatically releases a specific mobile agent to execute this service in this system. As introduced before, resident agents can provide necessary application services to mobile agent when it is visiting them. Thus, the mobile agent can automatically migrate around all the three mobile devices, automatically executing there and finishing this task and back to its owner. Finally, from Figure 7, we can find that the mobile agent had successfully collected the user list and information (e.g., their positions and photos) relative to its owner, and Figure 8 shows a common picture about their positions and the related information.

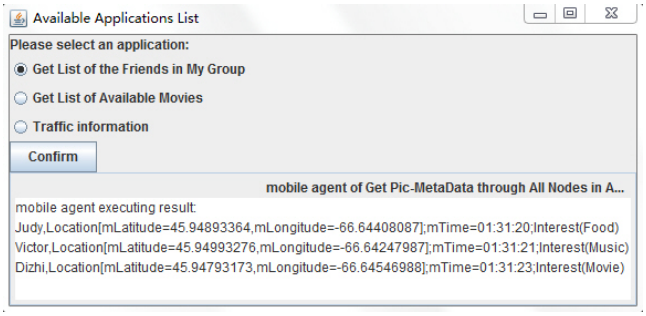

Figure 7. Users' information collected by mobile agent

Figure 8. Common picture for users displayed by application

5. COMPARISON AND EVALUATION

Recently there is an increasing trend towards mobile social networks, and several types of applications and systems have been proposed for vehicular social networks, thus in this part we compare with some other techniques which can be used for vehicular social networks. MobiSN is a semantic based framework for mobile ad-hoc social network [28]. It is implemented in Java 2 Micro Edition (J2ME), and supports users using their mobile devices to form mobile ad-hoc social networks, which are self-configuring. Meanwhile, MobiSN provides core implemented core functions and services for a mobile ad-hoc social network, such as the profile generating, friend matchmaking etc. However, due to the high dynamism of the vehicular social networks, it is hard to track down every possible scenario in advance, while MobiSN does not provide the extensibility support to developers, thus it is difficult for it to provide sufficient applications and services to users of vehicular social network systems.

Table 2. Projects summary

Project	Programming model	Systematic approach	Effectiveness
AmbientTalk	Actor	Partial support	Low
MobiSN	None	Partial support	Low
RoadSpeak	None	Partial support	Middle
S-Aframe	Multi-layer	Support	High

RoadSpeak is a framework for vehicular social networks, which allows users to automatically join voice chat groups along their roadways [1]. Different from traditional social networks, RoadSpeak also considers the time and location besides the interests of users in the users' definition when their groups are formed. Meanwhile, RoadSpeak partial supports extensibility. It provides some Java APIs to application developers, based on which developers can extend RoadSpeak clients to provide enhanced functionality. Nevertheless, applications of RoadSpeak rely on a server of this system. In a VANET environment, it is difficult to provide a stable server among the vehicles all the time. Also, RoadSpeak just provides voice chat service, but does not provide other services; thus it can hardly fulfill the diverse service requirements of users in the vehicular social network systems.

The basic idea of the AmbientTalk programming paradigm is that it can incorporate network failures in its programming model [21]. Meanwhile, AmbientTalk employs a purely event-driven concurrency framework based on actors. In AmbientTalk, actor executions can be concurrent, with asynchronous actor method invocations, thus AmbientTalk is very suitable for high dynamism networks. Also, AmbientTalk combines the Java virtual machine (JVM) as a platform, so it is easy for AmbientTalk programs to use Java libraries. However, AmbientTalk is a completely new language, which means that programmers have to spend a long time to become familiar with it before using it to develop applications for vehicular social network systems. Furthermore, AmbientTalk does not provide implemented application services, so the developing efficiency by AmbientTalk is low. S-Aframe makes use of the mechanism of AmbientTalk symbiotic programming with Java, and integrates generic services in its framework, thus S-Aframe supports programmers easily and effectively develop applications for vehicular social network systems by using Java.

6. CONCLUTION AND FUTURE WORK

In this paper, we have presented S-Aframe, a semantic based multi-agent framework for vehicular social networks. By integrating software agent and semantic techniques, as well as multi-layers framework, S-Aframe provides a high level software platform to developers of vehicular social network systems. Meanwhile, S-Aframe combined with multipath transmission technique provides an efficient strategy for multimedia transmission. Based on S-Aframe, a demonstration application used in vehicular social networks was designed and implemented.

S-Aframe hides the complexity of handling different connectivity status, varying services requirements and diverse devices in vehicular social networks, as well as with extensibility support. Thus it simplifies the development of applications for vehicular social network systems. Meanwhile, S-Aframe provides a multi-layer architecture, integrates with semantic technique, and supports cooperation among multiple owner applications with

multiple agents working at the same time. Thus applications can be more autonomous and adaptive to dynamically changing environments. Moreover, S-Aframe provides sufficient framework services for supporting application developers using the Java programming language with standard API format to develop applications. Consequently, S-Aframe improves efficiency and effectiveness of application development of vehicular social networks systems.

Further work of developing S-Aframe can be considered in many aspects. For example, security and privacy are important considerations in mobile social networks. We plan to use two strategies to address security issues. One is the identification of the ID names when a new mobile node wants to access the vehicular social network system. The other strategy is the dynamic services security identification. By integrating some techniques, such as semantic and data mining, S-Aframe can authorize only specific applications services to each mobile node according to its identity (or some other parameters). When a user releases a mobile agent, it can only use the authorized application services. S-Aframe can also analyze the activity of each mobile agent, then based on the mobile agent's activity history and its owner's identity, to decide whether the mobile agent is secure or not.

Also, since the framework service layer of S-Aframe is implemented by AmbientTalk, the layers below application layer are transparent. However, the relay selection for multimedia transmission is usually below the application layer. For instance, some research uses channel condition as measurement to find the best relay for source node [29]. One available solution is that we redevelop the lower layers of AmbientTalk, so as to extend the network status service to these layers and attain some useful parameters from them, such as the signal strength between the nodes. Meanwhile, we can compare the different network status obtained at different time and choose the relays that their network performance is most stable. In addition, when the relay nodes are disconnected with vehicular social network, the source node or destination node must select new relay node as soon as possible to prevent packet losses. In order to limit the data loss caused by relay failure to the minimum extent, we can use some redundant transmission schemes that send multiple copies using different relay nodes.

7. ACKNOWLEDGMENTS

This work is support in part by the NSERC DIVA Strategic Research Network.

8. REFERENCES

[1] S. Smaldone, L. Han, P. Shankar, and L. Iftode. 2008. RoadSpeak: Enabling Voice Chat on Roadways using Vehicular Social Networks. *In Proceedings of the First International Workshop on Social Network Systems* (SocialNets'08, in conjunction with EuroSys).

[2] C. Boldrini, M. Conti, and A. Passarella. 2007. Impact of Social Mobility on Routing Protocols for Opportunistic Networks. *In Proc. of AOC Workshop*.

[3] A. G. Miklas, K. K. Gollu, K. K. W. Chan, S. Saroiu, K. P. Gummadi, and E. de Lara. 2007. Exploiting Social Interactions in Mobile Systems. *In Proc. of Ubicomp*.

[4] F. Li, and Y. Wang. 2007. Routing in vehicular ad hoc networks : A survey. IEEE Vehicular Technology Magazine, vol. 2, no. 2, pp. 12-22.

[5] T. Nadeem, S. Dashtinezhad, C. Liao, and L. Iftode. 2004. TrafficView: A Scalable Traffic Monitoring System. *In Proc. of MDM*.

[6] M. Roccetti, M. Gerla, C. E. Palazzi, S. Ferretti, and G. Pau. 2007. First Responders' Crystal Ball: How to Scry the Emergency from a Remote Vehicle. *In Proc. of IPCCC 2007/NetCri07*.

[7] L. Wischhof, A. Ebner, and H. Rohling. 2005. Information dissemination in self-organizing intervehicle networks. IEEE Transactions on Intelligent Transportation Systems 6 (1) (2005) 90–101.

[8] J.-S. Park, J.-S. Park, J. Yeh, G. Pau, and M. Gerla. 2006. CodeTorrent: Content Distribution using Network Coding in VANETs. *In Proc. of MobiShare*.

[9] IEEE Foundation for Intelligent Physical Agents (FIPA). Agent Management Specification.

[10] R. Bindhu. 2010. Mobile Agent Based Routing Protocol with Security for MANET. International Journal of applied engineering research, dindigul, Volume 1, No1.

[11] T. B. Lee, J. Hendler, and O. Lassila. The semantic web. Scientific American 284 (2001), no. 5, 34 - 44.

[12] D. Brickley, and R. Guha. 2003. RDF vocabulary description language 1.0: RDF schema, Tech. report, W3C Recommendation 10 February 2004.

[13] D. L. McGuinness, and F. van Harmelen. 2004. Owl web ontology language overview, http://www.w3.org/TR/2004/REC-owl-features-20040210/.

[14] H. Boley, and M. Kifer. 2008. RIF framework for logic dialects. W3C Working Draft, Tech. Rep. http://www.w3.org/TR/rif-fld/.

[15] F. Martin. Recuerda, and D. Robertson. 2005. Discovery and uncertainty in semantic web services. *In Proceedings of Uncertainty Reasoning for the Semantic Web*. p. 188.

[16] G. Stamou, J. van Ossenbruggen, J. Z. Pan, and G. Schreiber. 2006. Multimedia annotations on the semantic web. IEEE MultiMedia 13 (2006), 86{90.

[17] R. Stevens, M. E. Aranguren, K. Wolstencroft, U. Sattlera, N. Drummond, M. Horridge, and A. Rectora. 2007. Using owl to model biological knowledge. International Journal of Human-Computer Studies 65 (2007), no. 7, 583{594.

[18] TripIt. http://www.tripit.com/, accessed June 2011.

[19] M. Wellens, B. Westphal, and P. Mahonen. 2007. Performance evaluation of IEEE 802.11 based WLANs in vehicular scenarios. *In Proceedings of IEEE Vehicular Technology Conference (VTC)* Spring, pp. 1167–1171.

[20] D. Gray (Ed.). 2007. Mobile WiMAX Part I: A Technical Overview and Performance Evaluation v2.8, Apr 2006.

[21] J. Dedecker, T. Van Cutsem, S. Mostinckx, T. D'Hondt, and W. De Meuter. 2006. Ambient-oriented Programming in AmbientTalk. *In Proceedings of the 20th European Conference on Object-Oriented Programming (ECOOP), Dave Thomas (Ed.)*. Lecture Notes in Computer Science Vol. 4067, pp. 230-254, Springer-Verlag.

[22] E. Hossain, G. Chow, V.C.M. Leung, R. McLeod, J. Misic, V.W.S. Wong, and O. Yang. 2010. Vehicular Telematics Over Heterogeneous Wireless Networks: A Survey. Computer Communications, vol. 33, no. 7, pp. 775–793.

[23] M. Ramadan, L. El Zein, and Z. Dawy. 2008. Implementation and evaluation of cooperative video streaming for mobile devices. *In Proceeding of IEEE 19th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*.

[24] F. Albiero, M. Katz, and F.H.P. Fitzek. 2008. Energy-Efficient Cooperative Techniques for Multimedia Services over Future Wireless Networks. *In Proceeding of IEEE International Conference on Communications (ICC)*.

[25] G. Ananthanarayanan et al. 2007. COMBINE: leveraging the power of wireless peers through collaborative downloading. *In Proceedings of the 5th international conference on Mobile systems, applications and services*. pp. 286-298.

[26] T. Seenivasan, and M. Claypool. 2011. CStream: neighborhood bandwidth aggregation for better video streaming. *Multimedia Tools and Applications*, pp. 1-30.

[27] http://pressroom.toyota.com/releases/toyota+friend+social+network.htm

[28] J. Li, and S. U. Khan. 2009. MobiSN: Semantics-based Mobile Ad Hoc Social Network Framework. *In Proceeding of IEEE Global Communications Conference (Globecom 2009)*, Honolulu, HI, USA.

[29] Y. Zhao, R. Adve and T. J. Lim. 2007. Improving amplify-and-forward relay networks: optimal power allocation versus selection. IEEE Transactions on Wireless Communications, vol. 6, no. 8, pp. 3114-3123.