This SocioTelematics application has been designed and implemented as a part of Ashad Kabir’s PhD research project funded by AutoCRC (http://www.autocrc.com/), under supervision of Professor Jun Han, Dr. Alan Colman and Dr. Jian Yu, at the Faculty of Information and Communication Technologies (ICT), Swinburne University of Technology (SUT). The team collaborates with IT industries, and is comprised of academics with expertise in mobile technology, human-computer interaction (HCI), adaptive systems, and service-oriented computing.

Ashad Kabir received his MSc in computer science from the Pusan National University (PNU), South Korea. Mr. Kabir has more than five years experience in developing mobile applications in different platforms such as J2EE, Windows mobile and Android. Jun Han received his Ph.D. in Computer Science from the University of Queensland in 1992. Since July 2003, he has been Professor of Software Engineering in the Faculty of ICT at SUT. Alan Colman is a Lecturer at the Faculty of ICT, SUT. He received a PhD and Master of IT in HCI from the same university. Jian Yu is a Lecturer at SUT.

Please describe your app concept. What is it? What does it do? How does it do it? Who will use it?

The SocoTelematics application makes the travel safe and convenient by allowing drivers to collaborate and interact with each other. The application allows two or more vehicle drivers to form a cooperative convoy by specifying themselves as a follower (i.e., driver of a following car) or a leader (i.e., driver of a leading car) and interacting with each other to make travel safe and convenient. Through these interactions a driver can share its domain-specific information - acquired from service providers (e.g., best route from a travel guide service), infrastructure (e.g., road block information from a traffic management system) and in-vehicle devices (e.g., vehicle location from GPS) - with other drivers in a convoy. For instance, the application allows drivers to share their GPS locations so that they see each other’s positions on the map. The interactions! between drivers are captured in interaction-relationship models which run on a web server (named SCaaS middleware), and are supported in terms of sending messages over the internet by SocioTelematics application running on the drivers’ mobile devices.

To reduce distraction during driving, the application allows drivers to automate message forwarding. For instance, a driver can specify message forwarding rules such as, when the best route information is received from a travel guide forward that to another car. The application also can remind drivers of convoy constraints as popup messages, which are already specified in the interaction relationships by drivers. For instance, it can warn drivers when they are not keeping their distance less than 500 meters as defined. Using the management interface of the application, the leading car driver can add a new car in the convoy while the convoy is in progress. A broken down car can leave the convoy and another car can take its place. The application allows changes in the defined agreements and constraints in the interaction-relationships at runtime to adapt to the changes of requirements and environmental context. For instance, a driver can change the maximum distance value, for example, 500m to 300m due to heavy rain.

The application executes a number of tasks running as background services: (1) Location sharing service – Listen to the location update from specified location provider (e.g., GPS) and send the position to the other cars as positionUpdate interaction message, (2) Message fetching service – Pull the received messages from the SCaaS middleware running on the cloud, (3) Message processing service – Extract information from the fetched message, evaluate and fire coordination rules, and update interaction history accordingly, (4) Map viewer service – Periodically update the cars’ (both leading and following) positions on the mapviewer so that drivers can see each other’s positions and (5) Text to speech conversion service – To reduce driver distraction, this service provides voice notification.
What problem (big or small) does your app solve?

Existing GPS/Navigation systems only allow a vehicle to see its own positions while the SocioTelematics application also allows to see other vehicles' positions. In particular, the application allows a driver to collaborate with other drivers by sharing information. For instance, if any vehicle experiences a mechanical problem, the application informs the other drivers as well as the roadside assistance. To reduce driver distraction, the application supports drivers to specify their information forwarding rules for outgoing messages and voice notification for incoming messages. Furthermore, the application provides an interface to change the defined interaction-relationships at runtime (see our UIC2012 paper\(^1\) for details).

Who is going to use your app and why? Please describe the market need for your app.

People often travel together in multiple vehicles having different facilities and want to collaborate and know each other positions. The SocioTelematics application reduces drivers communication gaps and increases awareness by enabling collaboration and interactions. This application is useful for logistics transportation vehicles, a group of people (e.g., tourists) travelling together in multiple vehicles, and so on.

How will your customer use your app? Describe how they will interact with the app and describe the user interface. (Please upload any supporting screenshots below.)

Before starting convoy, users need to specify their interactions-relationships. They can use provided built-in interaction-relationships model or can specify a new model based on their requirements. A web based graphical user interface will be provided to the users to assist them in defining and deploying interaction-relationships model to the server.

![Convoy panel](image)

**Figure 1. Convoy panel**

After specifying and deploying interaction-relationships models, users need to configure the SocioTelematics application running on their mobile devices. The application has three fundamental panels such as convoy, settings and interactions. Fig. 1 shows the convoy panel from the graphical user interface of the prototype. The map of the screen shows the cars' positions in the cooperative convoy. The telematics application uses GPS to show its car position on the map, while it gets the other car's position once it receives a positionUpdate message from the other car. Also it shows a warning message as pop-up, if the distance between the two cars violates their agreed distance.

---

When the application starts, the driver needs to configure the application using settings panel (see Fig. 2) by providing a user ID (to recognize the car) and a password. Using this ID the application invokes getRoleList functions at the server side, fetches roles associated to that user and populates those roles in the main screen. Also based on each role in the role list, the application invokes getInteractionList functions and accumulates associated interaction definitions for each role. This approach means that the application is not tightly coupled with roles played by the car in a cooperative convoy and also the company from where the car has been rented. Thus, it serves the purpose of multi-modal use, i.e., without any internal logic change this application can be used in a car from any rental company as well as by any driver in a cooperative convoy.

During driving, for the driver to perform the additional tasks in the cooperative convoy (i.e., forwarding information) as agreed may cause distraction. To facilitate such collaboration but with less distraction, the application allows the driver to specify coordination rules of forwarding such information. In the settings panel, a driver can select an incoming interaction message and its corresponding outgoing message(s) as his/her coordination preferences. Thus, when the application receives a message that matches with an incoming message in coordination preferences, the application sends the corresponding outgoing message(s) immediately on behalf of the driver. Also, the application provides voice notification (message name and contents) when a message comes.
The interaction panel (see Fig. 3) shows: the role list – associated with the driver of the car, the interactions list – list of interactions corresponds to the selected role in the role list, and the interactions log – interactions so far occurred.

**How will your app make money?**

The range of V2X interactions such as In-Vehicle (inV), Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I), and Vehicle-to-Service Provider (V2SP), together with the recent advancement of Internet and mobile technologies illustrates significant potentials of developing more intriguing user-centric applications such as “cooperative convoy telematics” – where drivers can share information with each other to make travel safe and convenient. Automotive software systems have started evolving with the proliferation of Internet technologies (such as cloud computing, service oriented computing, etc), and the ever growing popularization of mobile devices, especially smart phones. DENSO corporation in Japan – a leading supplier of advanced automotive technology – recently proposed an Automotive Cloud Service System software architecture for integrating vehicle software with the cloud services. The recent shift from hardware to software provides new opportunities for telematics systems to arise [http://analysis.telematicsupdate.com/]. Both academia and industry have started to discuss the potential of cars’ socializing [http://www.pervasive.jku.at/AutoUI12_SocialCar/] [http://www.socialcar.com/]. Thus we believe that the SocioTelematics application has a bright perspective and a huge amount of profit can be made from it.

**What is the stage of development for your app? Is a prototype available?**

So far, we have developed a prototype of the client application for android and a middleware for managing drivers’ interaction relationships. We evaluated the application’s communication latency and adaptation overhead using two cars in a cooperative convoy in 50 kilometres of driving where the middleware was deployed in the Amazon Cloud and the two client SocioTelematics applications were running on two Android Samsung Galaxy Tabs with 3G connections. We have found that on average the application takes 1.6 seconds to pass a message from one client to another (see our UIC2012 paper\(^2\) for details).