Service Management for Co-operative Vehicular Systems.

Gary O' Connor, Olivia Brickley, Dirk Pesch.
Centre for Adaptive Wireless Systems,
Cork Institute of Technology, Ireland.
(Gary.OConnor, Olivia.Brickley, Dirk.Pesch)@cit.ie

ABSTRACT

The primary objective of Intelligent Transportation Systems (ITS) is the creation of advanced road traffic systems for improved traffic safety, efficiency, and traveling comfort. Applications such as trip planning, automatic tolling and emergency warnings, among others, are envisaged in a system which can potentially reform modern transportation. Basic vehicle and roadside infrastructure collaboration allows an increase in efficiency and safety and acts as a foundation for an extensive application set to achieve these ITS goals. Strengthened cooperation among vehicles and infrastructure instigates the deployment of a more diverse range of commercial and safety services, assisting travel security and efficiency. As the number of services planned for deployment in the transportation domain grows, it is clear that effective management is necessary to ensure correct operation and satisfactory performance. This service management framework should be flexible and extensible, ensuring adequate support for future intelligent transportation service provision. This article presents a generic architectural service management solution for co-operative vehicle-infrastructure systems. Designed for a Service Oriented Architecture (SOA) based environment and using existing protocols such as SIP and XML, a cooperative monitoring platform is proposed to manage cooperative services.

1. INTRODUCTION

The concept of Intelligent Transportation Systems (ITS) presents new R&D challenges in the transportation and ICT sectors and is currently receiving considerable interest from the research community. Initiatives by the European Union and Japanese and US governments offer extensive support for such research with the expectation of future improved road safety and travel efficiency, as well as environmental benefits. Projects such as PReVENT [1], SAFESPOT [2], CVIS [3], COOPERS [4], VII [5], UTMS [6] and others are all exploring the impact of and technological solutions for a more astute transportation network where powerful information sets are available. Construction of these rich information sets requires cooperation between vehicles and the roadside infrastructure. Vehicles in an ITS environment act as localized distributed data stores carrying potentially vital pieces of information, which, when knitted together with knowledge held by the infrastructure, can provide a wealth of information of varying levels of detail and significance. The existence of these fused data stores prompts and supports the development of services which address the goals of ITS, namely safety, efficiency and environmental considerations and will also have commercial benefits. Such collaboration among vehicles and roadside infrastructure is fundamentally dependent on information and communication technologies. Consequently, many studies in the ITS domain concentrate on application development and underlying communications platform issues.

Cooperative Vehicle-Infrastructure Systems (CVIS) is a FP6 EU funded integrated project under the eSafety initiative whose goal is to create a cooperative vehicle infrastructure environment, empowering information exchange among vehicles and between vehicles and roadside infrastructure in an effort to seamlessly provide a number of intelligent transportation services. In this vision for a cooperative vehicle infrastructure environment, CVIS proposes a heterogeneous cooperative communications framework based on the Continuous Air Interface for Long to Medium range (CALM) standard to provide user transparent, continuous communication in support of emerging ITS applications in the public transport, urban, interurban and freight and fleet domains. Envisaged applications include an enhanced driver awareness service (EDA) which keeps drivers informed of current local road and traffic conditions, and a cooperative traveler assistance service (CTA), which provides pre- and on-trip route planning support. A number of traffic planning, control and management services are also proposed. A number of safety services ranging from intersection safety to pre-crash protection are proposed by PReVENT, while VII is working on systems for congestion relief and collision prevention. In addition to these applications, the European Telecommunications Standards Institute (ETSI) have outlined a number of safety and non-safety services for the intelligent transportation sector [7,8]. With much effort being spent on standardizing and optimizing the communication technologies and investigating service deployment possibilities in the vast ITS application space, it is clear that some form of management is necessary to ensure correct operation and satisfactory performance of all projected cooperative services.

Based on work carried out in the CVIS project, this paper introduces a generic service management architecture for cooperative vehicular systems whose goal is to fulfill the temporal and data based operational service requirements, ensuring an acceptable system performance.

As previously explained, much of the work being carried out in the ITS domain focuses on the development of driver assisting service and the underlying communications technologies. Few papers address the service management challenge for a vehicular environment. In [9] a peer to peer based semantic approach to service management is taken for mobile ad-hoc networks (MANETs). This work investigates the fulfillment of service requirements from a mobile network perspective. By unifying semantic service description, service request caching and service request dissemination via P2P-multicast a solution is provided to the problem of efficient discovery and use of application-level services in MANETS. This view is fundamentally different from service management in a vehicular environment. For example the MANETS discussed in [9,10,11] need not consider safety critical applications where the fulfillment of hard real time requirements is crucial. In [12] an architecture is outlined for a middleware platform that would serve applications running in vehicular ad hoc networks (VANETs). This paper takes a very communication centric approach. The proposed middleware concerns itself with the maintenance of connections between communicating entities. It essentially manages the connection between communicating services rather than the services themselves. The architecture proposed in this paper will allow a more service orientated framework within which more effort will be focused on the fulfillment of real time service requirements. In [16], a service orientated architecture (SOA) and the Open Services Grid Infrastructure (OSGi) is used to demonstrate that end-to-end communication and service provision is theoretically possible over a multitude of network types. The paper gives a broad overview of technologies that could be used to construct a framework to support this end-to-end communication. Aside from commenting that a vehicular environment is both "noisy" and "unstable", the paper provides little technical information regarding the implementation of this proposed framework from a VANET perspective, other than to say it is possible. Our proposed framework has been tailored with the ultimate goal of providing requirements management to a plethora of services that will reside on the vehicular network.

The rest of the paper is organized as follows. The service oriented architecture paradigm and motivation for its utilization is outlined in the next section. Section 3 introduces the cooperative service management challenge, describing a typical cooperative service scenario and identifying the need for service management. The proposed architectural solution and its envisaged operation is described in section 4 while section 6 discusses the design constraints and middleware issues. Finally, section 7 describes future work in the area of cooperative service management for the vehicular environment.

2. SERVICE ORIENTATED ARCHITECTURE

One of the goals of Intelligent Transport Systems (ITS) is to provide a standardised array of services to disparate entities in a vehicular environment. To facilitate the provision of these services, entities in the network must have a way of finding and invoking these facilities. One method of achieving this is to provide a framework in which all potential services are registered in a repository which can be interrogated by potential clients. Essentially the system architecture is built around the services that are provided. For the cooperative vehicular environment, this service based approach will have to be designed for effective operation on one of three platforms, namely the Traffic Management Centre (TMC), Road Side Unit (RSU) and the vehicle. The TMC is the control centre of the network and performs traffic monitoring and control on a system-wide level. The RSUs are present at strategic locations in the network and contain information at a regional level. The vehicles themselves generate and report on information at a local level.

SOA is defined as an application architecture within which all functions are defined as independent services with well defined invokable interfaces which can be called in defined sequences to form business process. A service within SOA can be defined as an application function packaged as a reusable component for use in a business process. In essence, services within the SOA are self-contained, modular, interoperable, loosely coupled, location transparent, composite entities that can share formal contacts with other services in the form of Service Level Agreements (SLAs)[13]. Adopting a SOA approach within a network gives both service and network operators the ability to view their underlying network infrastructure more as a commodity and allows infrastructure development to become more consistent. As shown in Figure 1 a large community of potential customers can browse, select and enact services based on their needs. Simple services can be composed into high level business processes.

One key benefit of adopting an SOA approach is that it allows for a more process centric approach to service provisioning, enabling faster time to market. Another key benefit within SOA is that it facilitates dynamic service composition procedures where atomic service elements can be composed/recomposed depending on its demand characteristics and performance of its atomic service elements.

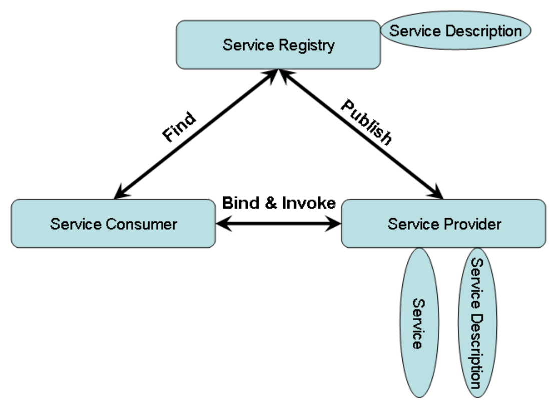


Figure 1 : SOA Collaborations

For the reasons stated above a Service Orientated Architecture ideally suites both the vehicular environment and the provision and management of the services therein.

3. COOPERATIVE SERVICE MANAGEMENT

Cooperation between entities in the network is one of the core principles of both a SOA and ITS. Multiple services running on each instance of a network element will need to communicate with one or more other network elements. Coupled with this demand for inter-element communication are the temporal and information requirements of the individual services/applications which have to be managed to ensure the correct operation of the applications. Without some form administration of these transmissions the system would degenerate into chaos. The following depicts a service management scenario with properly illustrates the above points.

Figure 2 shows a typical vehicular transport situation. Here an application running on the vehicle (i.e. route management) is providing a service to the end user. This service determines the density of traffic along a given path to a given destination. If traffic volume exceeds a tolerance level, a new route to the destination is calculated and the end user is informed. In order for this to become a reality, multiple interactions between services must occur in an orderly manner. As will be clarified in the next scenario, the infrastructure will be aware of traffic congestion. The application residing on the vehicle will have provided its route and request to be notified should high levels of traffic accumulate. The infrastructure, having received this information will begin monitoring the path. As can be seen in Figure 3 congestion has occurred, triggering the infrastructure to notify the route management application on the vehicle. The application will then calculate an alternative route to its destination and inform the end user. [**Error! Bookmark not defined.**]

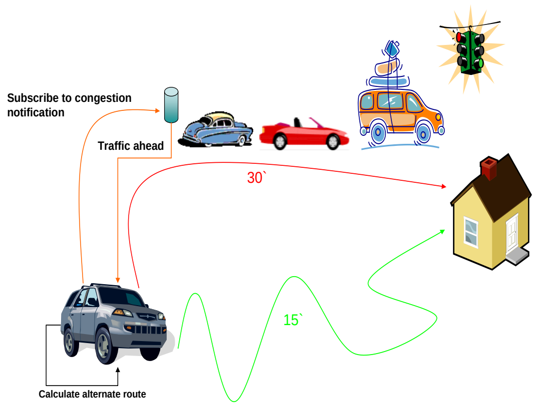


Figure 2 : Remote Service Management

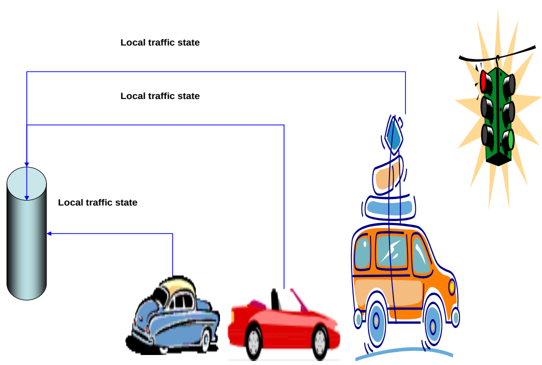


Figure 3 : Local Service Management

The scenario illustrated in Figure 3 is dependant on the actions shown in Figure 2. In order for the infrastructure to know when traffic has built up on a stretch of road, it must receive information from local vehicular entities. In this case, regular polling of the vehicles speed is occurring. The transition from low movement, to stationary and back again is interpreted as dense traffic by the vehicle. A service on the vehicle will then inform networked units in its vicinity. Similarly, other entities in the locality experiencing comparable traffic levels will do the same.

These aggregated traffic density messages will eventually make their way to the infrastructure which will interpret them as congestion along a specific path. Concurrently it will be checking for blockages along the route specified by the vehicle in Figure 2. Having found a match it then informs the application on that car of its findings. [**Error! Bookmark not defined.**]

Both of the above situations highlight the need for some form of management of the information and temporal requirements of the services resident on the network entities. This is what our proposed architecture hopes to achieve, however in order to develop this generic architecture it is important to classify the types of notification messages that exist within the network. Broadly these messages fall into two main categories:

3.1 Event Driven Notification

In general an event is an occurrence in one application or component that others may be interested in knowing about. A source posts an announcement of an event and handlers receive notification of it. These announcements are transmitted as event messages. In this architecture, entities in the network will subscribe to a given circumstance (e.g. Figure 2). When this event is realised, a message is transmitted to the requesting entity, which then will act accordingly.

3.2 Periodic Notification

Figure 3 illustrates a periodic notification scenario. The vehicle itself checks its own speed on a regular basis. The subsequent discovery of transitions from low movement to a stationary position and back again is interpreted as traffic congestion by the vehicle, which will then send an update to entities in its locality

4. COMPONENT ARCHITECTURE

Taking into account the requirements outlined in the previous section, Figure 4 shows a generic service orientated architecture for co-operative service management in a vehicular environment. This architecture will have the ability to manage the temporal and information requirements of the applications allowing them to fulfil their commitments. This management is achieved with the aid of three main components: the Message Manger, the LDM and the Subscription Manager. The Message Manager disseminates data to the corresponding entities in the Service Manger component of the architecture. The LDM (Local Dynamic Map) is responsible for maintaining an overall physical representation of the vehicular environment and supplying this raw data to the Subscription Manger when instructed to do so. The Subscription Manager fills the gap between the needs of the vehicular applications and the raw data being gathered. It receives requests for information from these applications, interprets them and ensures that the requesting application is informed if and when it becomes necessary.

These three components are described in more detail in the following sections.

4.1. MESSAGE MANAGER

The Message Manager is the point of entry and exit for data into the system. It is responsible for reformatting external messages for internal manipulation and comprehension and reformatting internal messages for transmission to external entities.

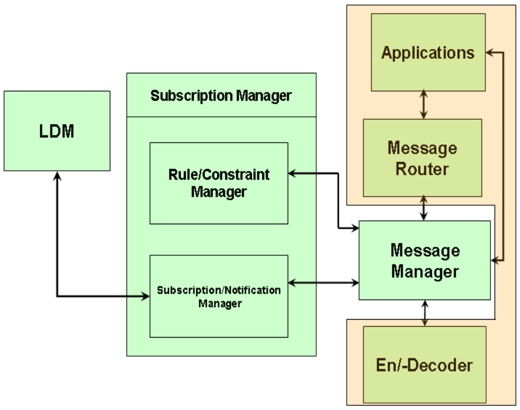
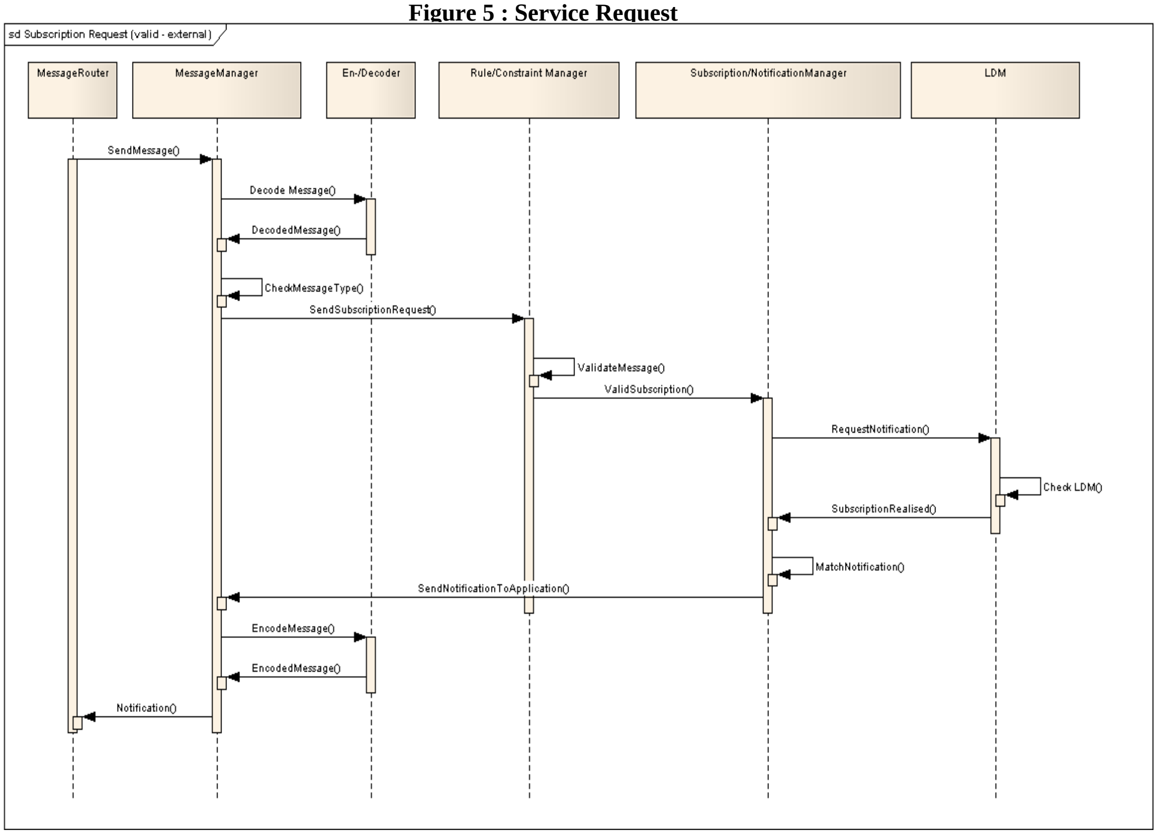


Figure 4 : Component Architecture

Depicted in Figure 5 is a subscription message from an external source. The message is transmitted to the Message Manager from the message router. The message that is received has been previously encoded for transmission. The Message Manager will decode all messages it receives from external sources by default. This newly decoded message is examined and in this case is revealed to be a subscription request. A superficial check is also performed at this point for conformance to certain basic messaging rules. If the message is valid it is forwarded to the Subscription Manager for further processing. The Message Manager may also be contacted by internal applications that require a subscription notification. In this case the procedure that is followed is identical to the one above, with the exception that the message will not have been encoded for external transmission, thus will not need to be decoded.

Figure 5 : Service Request



Upon realisation of a subscription the Message Manager will be contacted by the Subscription and Notification Manager. The requesting application now needs to be notified that the conditions they had specified have become true. The Subscription Manager will have associated the correct return address with the realised subscription. The Message Manager will determine the origin of the application. If the message is destined for an external source the message will have to be encoded for transmission. If this is not the case the message can be sent directly to the local application.

4.2 LDM (LOCAL DYNAMIC MAP)

The architecture proposed in this paper is targeted at managing cooperative applications that are part of second generation driving systems. These systems are largely dependent on the distribution of geo-referenced information which will be contained in some form of data store. The suggested framework is not dependent on any particular incarnation of a data store. While the majority of the infrastructural platforms (i.e. the centre and RSU) will have identical implementations of a data store, it is highly likely that the vehicular platforms will have varying proprietary instances. Not placing a restriction on the type of data store in each vehicle allows the architecture to be more generic and, it is hoped, more widely adopted. Safeguards have however been put in place to ensure that the system can accommodate varying implementations of a data store.. This safeguard is in the form of an encoding and decoding mechanism which will be explained in more detail in section 6.

The LDM is the subsystem that is responsible for the management of the database which contains all geo-referenced data. The LDM is present in all three platform types: centre, infrastructure and vehicle. It is conceived that it will contain a digital map, which will act as the framework for the integration of sensor data, where sensors are in this case other vehicles and/or the infrastructure, and the information will concern observations done by those other vehicles as well as the position or trajectory data of those vehicles themselves. It is from this static data (i.e. the digital map) and the dynamic information (i.e. other vehicles, RSU's etc) that the Local Dynamic Map can be constructed. This map represents nearby information and is dynamic as its content is rapidly changing, especially as vehicle and other objects are rapidly changing positions. [14]

4.3 SUBSCRIPTION MANAGER

The Subscription Manager component of the architecture receives subscription requests from applications either from either local or remote entities. The request is interpreted by the Subscription Manager which uses the environmental information at its disposal to ascertain when and if the request is realised. It will then instruct the Message Manager to notify the appropriate application. It is the entity of most importance within the architecture as it facilitates service and application provisioning across the network. The Subscription Manger itself is sub-divided into two components, the Rule and Constraint Manager and the Subscription and Notification Manager.

4.3.1 Rule & Constraint Manager

As seen in Figure 5 the Rule and Constraint Manager is the recipient of a subscription request from the services/applications via the Message Manager. This request is checked against a series of rules and conditions for conformity. For example if an application wishes to know about all traffic on all routes, this message would be rejected as it is too general, with a request for retransmission with more specific information. If the message is determined to be a valid subscription request it is forwarded to the Subscription/Notification Manager. If the subscription request is determined to be invalid, the message is returned to the service/application that initiated the query (via the Message Manager) with an appeal to review/change its request and an accompanying explanation e.g. data unavailable.

4.3.2 Subscription & Notification Manager

There are a plethora of services available to users of a service orientated vehicular network, everything from route management to alerts from emergency vehicles in the vicinity to reserving parking. The temporal and information requirements of these services will be fulfilled by this element of the architecture. The Subscription & Notification Manager will receive valid requests from the Rule & Notification Manager. It is the responsibility of the manager to disassemble this request into one or more information requests, and instruct the LDM that it is to be informed should one/all of the values change. The Subscription & Notification Manager will then aggregate the alerts it receives from the LDM and deduce when and if the event(s) specified by the application/service have been realised. It may be the case where a request from an application will mandate the retrieval of information from the LDM. It is only when the event(s) specified by the application have occurred, that the Subscription & Notification Manager will obtain this data from the LDM.

As well as interpreting requests made of it by the applications on the network, the Subscription & Notification Manager will also record addressing information for the applications. Subsequent to decomposing the request into its component events, the manager will associate the cumulative realisation of these events with a service/application on the network.

If and when an application needs to be contacted, the Subscription & Notification Manager will perform a lookup, extracting the addressing information. It now has the means to notify the service of the results of its request. If the request also mandated the return of data from the LDM, it is at this phase of the notification process that this information will be sent.

5. SESSION INITIATION PROTOCOL

It is clear that a framework must be put in place for inter-entity communication in this architecture. Transmission of the message itself is not of importance as that aspect of the process is handled by the message router. It is the protocol used to encapsulate this message for transmission that is of interest. The criteria that must be met by a potential protocol are as follows. It must:

- be open source
- have well established protocols
- be device neutral
- have the ability to carry a large volume of data

Many communication protocols were considered with the above requirements in mind. Both ICE Internet Communication Engine) and DCOM (Domain Component Object Model) were initially considered, however their proprietary nature inhibited their adoption. CORBA (Common Object Request Broker Architecture) and SOAP (formerly the Simple Object Access Protocol) were also considered for the framework. These protocols, while open source in nature are too inherently complex to cleanly insert into the already multifarious architecture proposed here. It would have the effect of adding a superfluous layer of complexity.

It was ultimately decided to utilize the Session Initiation Protocol for encapsulation of inter entity communication [15]. This protocol fulfils all of the requirements listed above and can be extended to incorporate the necessary functionality. The payload of the SIP message will contain XML (or a derivative). [16]

Figure 6 depicts a typical SIP subscription conversation scenario.

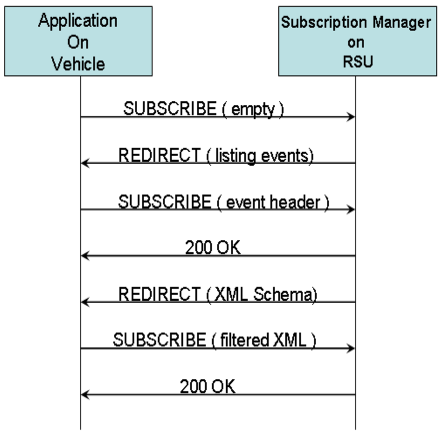


Figure 6 : Generic SIP Subscription

The initial message that passes from the requesting party (in this case a vehicle) to the receiving party (in this case a road side unit) is an empty SIP SUBSCRIBE message. This has the effect of alerting the RSU to the presence of an external entity wishing to request information. The RSU responds with a SIP REDIRECT message which will contain a list of the events that this unit can provide information for (e.g. traffic alerts, emergency vehicle alerts etc.). It is then the responsibility of the requestor to parse this list for the appropriate event and generate another SUBSCRIBE message containing the event of interest. This message is acknowledged by the RSU using the SIP 200 OK message. The RSU now knows what type of event that the querying application/service on the vehicle is interested in and is in a position to supply the corresponding XML schema it needs completed. This schema will contain the information and data format necessary to enable the RSU to complete the request. This schema is transmitted using another SIP REDIRECT message. The application/service running on the vehicle will either have the necessary information or the ability to obtain it. When the necessary data is ready it is packaged in an XML filter and sends it out encapsulated in a SUBSCRIBE message. This transmission will be recognized as a filtered subscribe by the RSU, where it will follow the process described earlier.

If the Rule & Constraint Manager detects a flaw in the filtered XML schema, a final 200 OK message will not be returned. Instead a SUBSCRIPTION_FAIL message is sent, the payload of which will contain a rational for the failure.

Figure 7 gives an example of one of the schema exchange messages that will take place as part of the SIP conversation sequences described above. The message has two main components: the SIP addressing information, and the XML payload. As previously discussed, XML will play a large role in the implementation of the publish/subscribe aspect of this architecture. The selection of an XML payload type is due to its simplicity, flexibility, platform independence and ready integration with the SIP standard. It will be used in the following areas:

- Message exchange
- Data structure & Semantics
- Subscription rules

Message Exchange: The messages exchanged between the services in this architecture will be based on a modified SIP standard. These messages will have an XML payload. This payload will contain the data needed to enable the Service Manager to fulfill the requirements of the application.

Data Structure & Semantics: In order for the XML data exchanged between the entities to make any sense, a common understanding of the meaning of the data must be achieved. An XML schema is used to achieve this. The XML schema for an event will consist of a set of typed elements. Each element will contain a type and a name. As part of the conversation between a service and the Service Manager, an XML schema is exchanged. This schema will contain the data types needed to fulfill the subscription and their interpretation. Figure 7 shows a typical example of one of these schemas.

Subscription Rules: The final XML based element is concerned with the rules and constraints language necessary for the service to specify its requirements. With proper and efficient construction, XPath expressions can be adapted to query the XML datasets.

```
SUBSCRIBE sip:alert@rsu.com SIP/2.0
EVENT: traffic_congestion
EXPIRES: 12345
FROM: sip:route_management@vehicle.com
TO: sip:alert@rsu.com
CONTENT:

<?xml version="1.0" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:complexType name="congestion">
   <xs:sequence>
     <xs:element name="location"type="string"/>
     <xs:element name="level" type="string"/>
     <xs:element name=”time” type=”integer”/>
</xs:sequence>
 </xs:complexType>
</xs:schema>
```

Figure 7 : SIP Message

6. DESIGN CONSTRAINTS

As stated in the introduction, adopting a service orientated approach to this architecture is the optimal method managing the needs of services and applications in a vehicular environment. The implication of taking this approach is that all facilities/applications in the system must be registered in a central repository as a service. One of the most reliable and efficient methods for achieving this in a vehicular environment is to “bundle” all services (including the Service Manager) and place them in an OSGi repository.

Another aspect of the vehicular environment that places a constraint of the system is, as previously mentioned, the diversity of LDM implementations that will exist. In order to mitigate against this, a common understanding of geographic information must be put in place. The optimal method of implementing this is to have an encoding/decoding component present on all entities in the network. This component will effectively encode the geographic information obtained from the LDM into a form that is intelligible by the decoder on the receiving end.

6.1 OSGI

Among the unique requirements of this environment is that manufactures must have the ability to remotely and dynamically upgrade the automotive software quickly, economically and securely. Due to the diversity and lack of standardization when it comes to types of vehicle and vehicular software of the automobiles populating the network, a service platform must be selected that can run on almost any operating system and processor in existence.

The OSGi specifications define a standardized, component orientated computing environment for networked services that is the foundation of an enhanced service-orientated architecture. The addition of this service platform adds the capability to manage the lifecycle of software components in the vehicle from anywhere in the network, while simultaneously reducing the overall complexity of building, maintaining and deploying applications. Components can be installed, updated or removed on the fly without ever having to disrupt the operation of the device. The Service Platform itself has two components, namely the OGSi framework and a set of standard service definitions. The OGSi framework which sits on top of a JVM (Java Virtual Machine) is the execution environment for services. [17]

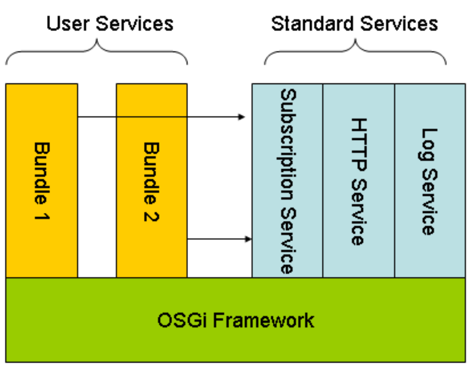


Figure 8 : OSGi Framework

In OSGi, service providers and requestors are part of an entity called a bundle that is both a logical as well as a physical entity. Service interfaces are implemented by objects created by the bundle. In standard OSGi, the bundle is responsible for run-time service dependency management activities which include publication, discovery and binding as well as adapting to changes resulting from dynamic availability of services that are bound to that bundle. [18]

For the purposes of service management within this architecture, it is foreseen that the Subscription and Message Management elements of the architecture will be bundled and presented as separate OSGi services within the platform.

6.2 ENCODING/DECODING

All geo-referenced messages in the system sent to or received from an external source must be encoded. AGORA-C is the en-/decoding method that has been selected to achieve this transmission of geographic information. This is a relatively new, generic method for location referencing. It has been designed for traffic telematic systems that use a digital map on both ends of the communication chain.

AGORA-C is a map-based on-the-fly location referencing method. The method concept is designed to compensate for differences that may exist between the map used at the sending system (the encoding side) and the map onboard the receiving system (the decoding side). The encoding rules provide the necessary semantics both for creating the location code at the sending system and for interpreting this code in the end terminal. Thus, the role of the encoding rules is both to provide constraints for selecting and creating this set of information elements at the sending system, and to provide a consistent interpretation basis for the receiving system to reconstruct the location reference as intended by the sending system. Its main advantage resides in the fact that it is not based on pre-coded location tables, but is coded and decoded using solely the map database. Any location or route portion of the database can therefore be coded. Pre-coding of locations, maintenance, and dissemination of location tables are no longer necessary.

7. CONCLUSION & FUTURE WORK

This paper presented a template for a generic service management architecture that will facilitate the fulfilment of the temporal and information requirements of services resident on a co-operative vehicular network. It was crucial in the development of this architecture that it not only has the ability to efficiently and effectively manage the needs of the ITS applications, but also that it be forward compatible. With this in mind, standard technologies were chosen which were both proven and extensible.

In terms of future work, the type(s) of information filtration mechanisms still need to be ascertained. It is clear that this proposed architecture lends itself perfectly to a publish/subscribe paradigm. A service/application expresses its interest in receiving certain types of events by submitting a predicate defined on the content of the event (i.e. the applications subscription). When a new event is generated and published, the infrastructure is responsible for checking the event against all current subscriptions and delivering it to all applications whose subscriptions match the event.

The proposed architecture has been designed with the intent of managing subscriptions across diverse sizes of network. It will be capable of processing subscriptions & notification messages that number in the hundreds of thousands. However in order to reduce the volume of unnecessary information being sent and received, some form of filtration will be necessary. The following are the options available:

Group Based: With this method of publish/subscribe, a set of groups (or channels) are designated by the system. Each event is published to one of these groups by its publisher. A service subscribes to one or more groups, and will receive all events published to the subscribed groups. For example, an emergency services scenario in the vehicular environment. In the event of a major fire, all vehicles on the route from the fire station to the incident could be notified well in advance that an emergency vehicle is approaching and to move out of the way. This type of notification does not distinguish individuals, but merely categorizes them based on some common criteria.

Topic Based: A system based on this type would be more flexible. When the service is publishing its event, it is tagged with short subject (or topic). This tagging is representative of the content of the event. The information contained within this subject can be either arbitrary or taken from a predefined schema. When a subscription is taking place the service will base their subscription on the contents of this subject header. It is also possible to subscribe to a subset of the information contained in this subject header. For example a service could subscribe to congestion notification for an entire road or just a section of it.

Content Based: A content based system is endowed with a much greater level of flexibility. With this approach, the service is able to more accurately express their query. Instead of relying on the publisher to classify the events into groups or subjects, the subscriber is able to construct their own subscriptions. The greater flexibility offered by this system is somewhat offset by the complexity involved in its implementation. Due to the arbitrary composition of the subscription, interpretation and lookup become difficult. [19]

Each of the approaches defined above have merits and drawbacks. It is hoped that with proper time and effort the appropriate option or combination of options can be applied to the system.

At present the architecture proposed in this paper is being implemented in the COMO (Co-operative Monitoring) subproject of CVIS. It is expected that a finalised incarnation of this framework will be evaluated in the CVIS test sites as part of the integration and requirements based testing, the results of which will be published at a later date.

8. ACKNOWLEDGEMENTS

The authors wish to acknowledge the support of the European Commission for financial support of this work under the EU IST FP6 CVIS project.

9. REFERENCES

[1] http://www.prevent-ip.org
[2] http://www.safespot-eu.org
[3] http://www.cvisproject.org
[4] http://www.coopers-ip.eu
[5] http://www.its.dot.gov/vii/htm
[6] http://www.utms.or.jp
[7] ETSI TR 102 492-1, "*Electromagnetic compatibility and Radio spectrum Matters (ERM); Intelligent Transport Systems(ITS); Part 1: Technical characteristics for pan-European harmonized communication equipment operating in the 5 GHz frequency range and intended for critical road-safety applications.*"
[8] ETSI TR 102 492-2 *"Electromagnetic compatibility and Radio spectrum Matters (ERM); Intelligent Transport Systems (ITS); Part 2: Technical characteristics for pan-European harmonized communications equipment operating in the 5 GHz frequency range and intended for road safety and traffic management, and for non-safety applications related ITS applications"*
[9]P. Baumung, S. Penz, M. Klein "*P2P-Based Semantic Service Management in Mobile Ad-hoc Networks*", 7th International Conference on Mobile Data Management, Nara, Japan, May 2006
[10] O. V. Drugan, T. Plagemann, E. Munthe-Kaas "*Building Resource Aware Middleware Services over MANET for Rescue and Emergency Applications*" 16th IEEE Personal, Indoor and Mobile Radio Communications (PIMRC), Berlin, Germany, 2005
[11]K. Farkas, L. Ruf, B. Plattner, "*Framework for Service Provisioning Framework forSelf-Organized Networks*", 1st International Conference on Telecommunications and Computer Networks (IADAT-tcn 2004), San Sebastian, Spain, 2004
[12] R. Meier, B. Hughes, R. Cunningham, V. Cahill, "*Towards Real-Time Middleware for Applications of Vehicular Ad Hoc Networks* ", 5th IFIP International Conference on Distributed Applications and Interoperable Systems (DAIS), Athens, Greece, 2005
[13] Channabasavaiah, Holley, Tuggle, *Migrating to a service-orientated architecture,* IBM DeveloperWorks, 16 Dec. 2003.
[14] Wevers, Lu , "*Digital Maps, driving assistance systems and traffic safety: the data chain for in-vehicle map databases*".
[15] J. Rosenberg, H. Schulrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler, "*SIP: Session Initiation Protocol"*, RFC 3261, Internet Engineering Task Force, June 2002
[16] Arabshian, Schulzrinne , "*A Generic Event notification System using XML and SIP*." , NYMAN Workshop, Sept 2003.
[17] OSGi Alliance, *"About the OSGi Service Platform Technical Whitepaper",* 7 June 2007
[18]http://www.cvisproject.org/download/Deliverables/DEL_CVIS_3.3_Architecture_and_System_Specifications_v1.2.pdf
[19]Huang, Garcia-Molina, "*Publish subscribe in a mobile environment, department of computer science",* Wireless Networks (Journal), Jan. 2005.