Disruption tolerance in vehicle to infrastructure communication: 
Making a Case for Intelligent Roadside Infrastructure

Sandford Bessler¹, Eva Kühn², Thomas Paulin¹
¹FTW Telecommunications Research Center, Austria
²Vienna University of Technology, Institute of Computer Languages, Austria

Abstract
The realization of novel vehicular services requires a distributed computing and data management architecture in the roadside nodes. We illustrate the proposed architecture with a data transport service that can cope with connectivity disruptions, inherent to this kind of mobility. We discuss the results obtained by running the disruption tolerant service in a field trial on Austrian motorways.

Keywords: IEEE 802.11p, roadside overlay network, space based computing, disruption tolerance.

1. Introduction
Intelligent transportation systems have to meet the big challenges of transportation people and goods to; a) reduce the number of casualties and personal injuries, and b) reduce the number of traffic jam hours. ITS experts controversially argue for the superiority of autonomous in-car systems or cooperative solutions based on V2X communications [9], [10]. Meanwhile, it becomes clear that the vehicle will be networked and both in-car and networking approaches will coexist. ETSI defined an ITS station architecture [8] to be realized both in the On-Board Unit (OBU) and in the Roadside Unit (RSU). The ITS station has besides communication functionality, middleware functionality (facilities layer) such as local dynamic map, geo-routing, etc. In this work we argue that more intelligence is needed in the ITS station to address the challenges of disrupted connectivity, massive floating car data collection, intelligent dissemination of alert messages and of many other future mobility applications.

We show that distributed computing solutions such as peer-to-peer networking and
space-based middleware [2] are required in the RSU design to realize storage in the network. The main contributions of this paper are twofold:

- To review the system and protocol design for disruption tolerant networking (DTN).
- To report performance based on real-world measurements of interactive V2I sessions in presence of interruptions.

The rest of the paper is structured as follows; in Section 2 we discuss the special requirements for the design of a roadside service infrastructure. Section 3 describes the design of disruption tolerant V2I communications. Section 4 discusses road tests performed to validate the designs described above. Finally, Section 5 concludes and discusses future work.

2 Architectural Aspects in Creating Service Intelligence

When considering ITS protocols such as IEEE 802.11p (ITS G5), intermittent connectivity is given by design; namely when vehicles move out of RSU range. The first generation of vehicular platforms deployed in large field trials during the last four years are mainly result of international projects and standardization activities: Car-2-Car Communication Consortium, IntelliDrive program in US and the three large IPs in FP7: CVIS, SAFESPOT and COOPERS. However, the only approach for insuring connectivity of fast moving OBUs has been Communications Access for Land Mobiles (CALM), standardized by ISO TC204, an approach based on the availability of several wireless technologies and seamless, vertical, handover to maintain connectivity.

Another approach, such as the concept of disruption tolerance [7] that is becoming main stream in future Internet research, has until now received little attention in the context of ITS services. To realize this approach, we heavily use the temporary storage of the content of vehicular messages in the roadside nodes.

However, not only DTN requires intelligent message handling; in vehicular applications special requirements arise from the type of information exchanged through the network, namely geo-located, short-lived status messages. For efficient processing, it is required to keep the data in collections according to certain criteria such as service type, geographical origin location or destination, vehicle, etc. This requirement is met by the so-called containers of extensible virtual shared memory (XVSM) [5]. XVSM is a middleware that allows storing data in a shared space, structured into containers. It is based on the concepts of Linda [3] and JavaSpaces [4], and allows for a flexible and extensible coordination of arbitrary data. Therefore, in the proposed roadside node architecture (see Figure 1), the messages (strings, files, tuples, etc.) are stored, processed and routed between space containers. The interactions between the applications and the containers are mostly asynchronous, using a
subscribe-notification pattern. Intelligent queries in the containers are possible using filters and ordering selectors.

Figure 1: Space based architecture of a roadside node

The RSU node is part of the infrastructure, a wired, provisioned and reliable network. It also has a WAVE connection with the vehicles situated in its range. Service applications may start instances on every RSU node or a subset of the nodes. Each application may require a number of application specific space containers. Their deployment on nodes can be optimized to share the load or minimize resource usage (storage, CPU, etc.).

3. Disruption tolerance in ITS G5 environment

The disruption tolerance service, realized in the ROADSAFE project, is the enabling technology for the operation of various data information services, such as retrieval of images from traffic cameras.

In general, enabling disruption tolerance requires the following key components:

1. Storage at network intermediaries.
2. Forwarding strategies: Pull or proactive (e.g. multiple destinations).
3. Fragmentation of application payload data units (PDUs) into packets because of short coverage periods.
Another requirement is to trigger the DTN mechanism only when the physical channel quality rapidly deteriorates. Therefore we made the following assumptions in the design:

- Application dialogues may choose to use DTN or not if deemed necessary.
- DTN enabled requests have to specify a DTN-service access point that has been previously announced by a RSU in a broadcast message on the control channel.
- A simple DTN enabled dialog, defined also as session in this paper, consists of two asynchronous messages; a request and a response, where each message is segmented in multiple packets and thus may be interrupted.

The environment in which DTN is designed is the IEEE 802.11p communication stack. The service is advertised like other ITS services during the 50 ms control channel slots. Each individual RSU, which supports DTN, has to specify a service access point (DTN SAP) in the announcement messages. An OBU that wants to use DTN for its service transactions has to include the SAP in the packet headers, because at this address the data storage and processing takes place. Specifically, the recombination of request packets, the fragmentation of response packets, the storage and the final service call occur at the SAP, which may be instantiated in the local RSU, or any other host, as long as it is known by the OBU.

3.1 Protocol Overview

We proceed with the analysis and assume that the request PDU, which consists of \( n_{\text{req}} \) packets, is sent to the first RSU but might be interrupted after the OBU sends \( k_{\text{req}} < n_{\text{req}} \) packets. The OBU modem detects the interruption instantly if no ACK is received on the MAC layer after the packet has been sent a maximum \( t \) number of times. In the same time the link quality is monitored at the OBU by listening to beacon signals from the RSU. As soon as the link to the RSU is good, the OBU continues to send packet number \( k_{\text{req}} + 1 \), using the initial SAP address. Note that after the connectivity disruption, the vehicle may still be in range of the initial RSU or in range of a new RSU. This is illustrated in Figure 2a.

The second part of the protocol analysis considers that a disruption occurs during the reception of the requested payload. Figure 2b illustrates the case in which the OBU has successfully sent the request, but the response is disrupted after \( k_{\text{resp}} \) packets. The DTN-SAP, which completed the request, forwards the fragmented payload to the OBU, via the RSU. Only if a packet \( k_{\text{resp}} \) in the sequence has not been successfully acknowledged by the OBU on the MAC layer, the DTN mechanism is activated; the RSU forwards the negative report, indicating the lack of acknowledgment, to the DTN-SAP, which in turn stores the remaining \( n_{\text{resp}} - k_{\text{resp}} \) packets in its local container.
Figure 2: a) Message flow for request interruption, b) Message flow for response interruption

The OBU has received and acknowledged the first $k_{\text{resp}}$ packets and checks now the link quality based on successful beacon receptions. When the link is up, the OBU sends a DTN-query message to the RSU in range. The query contains the SAP ID, session ID and sequence number of the last received packet. The query causes that the packets, $k_{\text{resp}} + 1, k_{\text{resp}} + 2, \ldots, n_{\text{resp}}$, are retrieved by the RSU link manager and transmitted to the OBU in range. The mechanism is seamless as it does not distinguish between a 100 ms connectivity drop within the range of a RSU and a 100 second disruption between RSUs, the handling is the same.

4. Performance of the DTN Mechanism on the Road

For the evaluation we installed five IEEE 802.11p RSUs on highway gantries in the vicinity of Vienna, in cooperation with ASFINAG and Kapsch TrafficCom. The schematic of the test setup is illustrated in Figure 3. Additionally a vehicle was equipped with a DTN enabled OBU and a DTN relevant file transfer application, which requests pictures from an upcoming traffic camera. Each RSU and OBU consists of an 802.11p enabled modem connected over Ethernet to an embedded computer or a standard laptop, respectively, on which the test software was running on. During the measurements the OBU was driving in the test area and the DTN application was requesting a new file as soon as the previous one has been successfully downloaded.
The goals of the DTN measurements are;

1. to quantify DTN dialogue completion time and how the time is spent,
2. to explore the spatial and temporal distribution of interruptions,
3. to determine request and query processing overhead in terms of time and
4. to evaluate the average throughput during connectivity periods.

Table 1 shows an overview of the parameters used during the experiment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame size (includes DTN header)</td>
<td>1400 B</td>
</tr>
<tr>
<td>Data rate</td>
<td>6 Mbit/s</td>
</tr>
<tr>
<td>Power</td>
<td>17 dBm</td>
</tr>
<tr>
<td>Number of MAC layer tries</td>
<td>11 tries</td>
</tr>
<tr>
<td>Size of test images</td>
<td>200, 500, 1000 kB</td>
</tr>
<tr>
<td>DTN timeout trigger</td>
<td>2 seconds</td>
</tr>
<tr>
<td>Vehicle speed</td>
<td>100 km/h</td>
</tr>
</tbody>
</table>

4.1 DTN Dialogue completion time
The first set of results considers the time budget of DTN dialogs. Here we investigate how long it takes to finish a DTN dialog, given protocol and the setting described above. It is important to note here that the geographic density of the RSUs play a major role on the completion time, if the dialog
spans several RSUs. As illustrated in Figure 4 we consider two metrics; the completion time, defined as the time from sending a request to the RSU and until the dialogue has been completed, and the actual communication time, defined by the time where messages are actively exchanged between RSU and OBU, excluding interruptions but including the 50 ms control channel interval, where no application data is exchanged. In Figure 4, the large majority of the completion time is spent on either waiting for coverage or recovering from interruptions, namely between 89% and 94%. Worth noting is that the completion time consists of the actual communication time, the time needed to identify interruptions (through a timeout of 2 seconds), and the processing time of both requests and queries at the RSU.

![Figure 4: Average dialogue completion time for various payload sizes. The completion time includes the driving time between RSUs, while the communication time grows linearly with payload size.]

### 4.2 Spatial and temporal distribution of interruptions

In this series of measurements we verify previous lab experiments [1] based on real-world communication traces. The interruptions occur not only between RSUs, but mainly within the RSU range. In [1], the number of recorded interruptions was up to 31. Figure 5 shows a sample of the measurements, showing the completion of a 1000kB file request; the y-axis shows the progress in the amount of received frames, as the test vehicle passes three RSUs. The Query messages (red vertical lines) are issued whenever the OBU detects an interruption, as described in the protocol in Section 2. The highlighted green area shows the coverage as it is perceived by the OBU modem (as a function of the received RSU announcements). Note that the RSU has a different view on the situation, as all but the first Query in each green field indicates that the RSU cannot deliver a message to the OBU.
Figure 5: Cumulative data received indicates the distribution of connectivity interruptions within the RSU range and between RSUs (large horizontal segments).

4.3 Internal processing time overhead

In addition to the time spent with request processing\(^1\), interruptions and timeouts, the time to move data from or to containers must be considered whenever the RSU receives a negative MAC layer acknowledgement or when receiving a query from the OBU (see Section 3.1). In this situation the software process on the RSU has to either write the remaining data into the container or to read stored session data from the container and send it on the wireless link. Figure 5 shows the time required for read and write operations for various message lengths, where each frame consists of 1400 bytes. This time overhead occurs at each interrupted session. Assumed the total connectivity time is around 20-25 seconds at 100 km/h, an interruption which requires an internal data transfer time of up to 1.4 second (write and read operation) are quite. The writing time is covered at the OBU side by the timeout duration that makes sure that the query is not sent before all write operations are finished. A solution to this problem could be a division of large payloads in smaller chunks that are written or read at a time.

Additionally, a more reliable way of detecting whether vehicles are indeed out of coverage (e.g. location-based) would allow the RSU to buffer messages until they either can be sent or written to the container. The peak at 900 frames in Figure 6 can be explained due to the low sample and therefore high uncertainty, as only a few of the 1000 kB session were interrupted within the first 100 frames.

---

\(^1\)Request processing is considered constant, independent of which communication approach is used, and is therefore not considered further.
Effective Throughput of DTN operation
The measured throughput is influenced mainly by four factors: the data rate, the response time of the middleware implementation, the switching between control and service channel and the UDP-based interface between the modem and the computer.

Table 2 shows the measured throughput for each of the three payload sizes.

<table>
<thead>
<tr>
<th>Payload (KB)</th>
<th>Actual Throughput (Kbit/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>611.66</td>
</tr>
<tr>
<td>500</td>
<td>591.23</td>
</tr>
<tr>
<td>1000</td>
<td>600.71</td>
</tr>
<tr>
<td>Combined</td>
<td>601.20</td>
</tr>
</tbody>
</table>

Table 2 – Throughput times for different payloads

A throughput reduction is caused by the Ethernet interface between the computer on which the software is running and the 802.11p modem, increasing the time between individual frames are sent. Additionally, the current implementation buffers only one message from each session to the modem and waits with the next message until the modem replies. Therefore the channel utilization can be significantly improved if multiple messages are buffered in the modem, thereby reducing inter-frame time over the air, as well if multiple sessions are
running at the same time. Finally, the current implementation only utilizes 35-40 ms out of each WAVE service channel interval (of 50 ms), additionally reducing the measured throughput.

**Conclusions and further work**

In this paper we presented and evaluated a disruption tolerant networking mechanism, which is based on a self-organizing, peer-2-peer container infrastructure. The results show that by using the infrastructure it is possible to provide a reliable service, even when the communication is suddenly interrupted, either by bad channel characteristics or due to a vehicle leaving the coverage area of a RSU.

Road tests have confirmed the simulations from previous work and proven the concept of DTN in a vehicular environment using ITS communication technology and it seems to be increasingly useful when dealing with small payload and long respond times. The tests have shown that WAVE is a powerful technology that can be used for interactive communications, provided a reliable transport layer is implemented on top of WSM.

As the results also showed that there is room for improvement, future work will focus on investigating optimal retry strategies combined with probability calculations used to determine whether a vehicle has left coverage, used to reduce unnecessary container operation overhead. Additionally strategies that are able to better utilize the wireless channel will be developed.

**Acknowledgement**

This work was performed in the project ROADSAFE, a scientific cooperation between FTW, TU Wien, ASFINAG MSG, KapschTrafficCom AG, and Fluidtime GmbH, to improve vehicular communication systems. The project is co-funded by the industry partners, by the Austrian Government, and by the city of Vienna within the competence center Program COMET.
References


