Optimizing Bluetooth Wireless Technology as the Ideal Interface for Car Diagnostics

Lars-Berno Fredriksson, Kvaser AB, Sweden

Executive summary

Theoretically, a Bluetooth link could transfer messages to or from a CAN network using the full bandwidth of 1 Mbit/s and with less than a 3 ms message delay. However, due to mandatory higher layers in the Bluetooth protocol stack, far lower performance should be expected. Transfer through the RFCOMM would result in less than 42 kbit/s. Applications using the Ericsson baseband directly on the L2CAP layer has shown to cope with a 250 kbit/s CAN network unidirectionally and 125 kbit/s bidirectionally. The average latency at low message rates is in the order of 10 ms from master to slave and 20 ms in the reverse direction but the jitter is extensive and some messages show delays of 35 ms or more. For non-realtime critical car diagnostics, like OBDII and KWP2000, this is OK, but not for diagnostic tasks requiring realtime responses on specific messages.

The currently proposed architecture for Bluetooth in cars is a single access point. Any bus in the vehicle shares this Bluetooth resource. The vehicle access point will include not only existing Bluetooth services but also car specific ones, e.g., diagnostics, comfort adjustments, locks etc. and an extended temperature range. The main reason for a single point solution is cost. This might be disputed as the gateway will be very complex and the throughput low.

An alternative architecture would be one Bluetooth access point for each bus. The access point can then be optimized to the needs of respective bus and the inter-bus traffic can be handled by the ordinary gateways. Existing diagnostic software running on the respective bus can be used with minor modifications.

An optimal solution would be to restructure Bluetooth in a low-level mandatory part, including what everyone needs and leave the remaining parts as optional to be included higher layer protocols. CAN is an example on such a structure. We would then have a "Bluetooth engine" were anything not needed by every application is stripped off, connected to another module, optimized for its tasks, via a byte oriented interface. With such an architecture, the mandatory part can be implemented in a small and cheap module, used by any application. The proposed single access point can still be accomplished, but other solutions requiring higher performance or lower price could be embraced under the Bluetooth umbrella. "Bluetooth engines" would have a vast market and then a low price.

Optimizing Bluetooth Wireless Technology as the Ideal Interface for Car Diagnostics

Lars-Berno Fredriksson, Kvaser AB, Sweden

Introduction

Bluetooth is intended for Personal Area Networks, i.e., interconnecting any microprocessor device the average person uses. As the initiative came from Ericsson, a phone company, the first target was headsets followed by fax machines, PCs, PDAs, etc. As the average person uses his cellular phone in his car, the Bluetooth SIG gained a vested interest in car applications, especially the hands-free use of phones. To get a broader reference group than the promoting members of the SIG, the Automotive Expert Group (AEG) was established. The task of the AEG is to paint scenarios on how Bluetooth will be used in cars. Due to more and more countries and states ban the use of mobile phones in cars unless they are not operated hands free, this application is a high priority one. Looking simple, it turns out to be a quite complex task in a modern car, involving coordination between different networks in the car. It was also soon realized that there are several other applications a person would like to run in a car. Virtually anything he could do with Bluetooth in his home or at office, could be done in the car. On top of that there are several car specific applications as customized settings like seats, mirrors, radio, etc. and car commands like door locks, trunk opening, temperature adjustments, etc., not to forget vehicle recognition and connection establishment.

Facing the complexity of the Bluetooth use in a car environment, it was suggested that a specific Vehicle Interface should be added to the Bluetooth standard. This should include not only end user requirements but also car requirements as extended temperature range and service related ones. Diagnostics is here a high priority task.

Diagnostics requirements

Diagnostic tools can be grouped in three classes: OBDII, Third-Party and OEM tools. The OBDII tools are used for testing that vehicles meet legislated emission limits. Third-party tools are based on diagnostic standards and should then be capable of servicing cars of different makes. OEM tools are car manufacturer specific and capable of making realtime critical analysis. The three different groups have different needs regarding message latency, bandwidth and standardized procedures.

Message Latency

ISO 15765 “Road Vehicle - Diagnostics on CAN” shows a response time requirement within 100 ms for some OBDII related messages. This can be seen as an acceptable figure for general diagnostic applications. Specific OBDII instruments and third party tools for ECU diagnostics, relying on standards, have then to have a message latency well below 100 ms form bus to application and vice versa. OEM specific applications may require responses below 10 ms.

Bandwidth

OBDII is a small subset of car diagnostics and does not need a high bandwidth. Around 100 messages per second, corresponding to some 15 kbit/s, would be sufficient. Third-party general tools would need more as they might upload or download data from the ECUs. 400 messages per second, i.e., 60 kbit/s, would be considered appropriate. OEM specific diagnostic tool may not only aim for the ECUs but also diagnose the CAN traffic and then 3000 messages per second (approx. 450 kbit/s) would not be considered an unreasonable requirement.

Bluetooth Access Procedure

The Bluetooth access procedure is required by all diagnostics tools and it needs to be enhanced. The current Bluetooth standard lacks the possibility to identify a car. The first step in a diagnostic procedure is to establish connection with the right vehicle and thus, the first thing to standardize specifically for vehicles would be how to identify them. The Vehicle Identification Number (VIN) is widely used for this purpose by other standards and would be a good choice for Bluetooth.

Bluetooth Profile and SDP

OBDII and third party tools would rely upon interoperability. This will require a profile for diagnostics and a set of services. OEM specific tools do not have these requirements as they will only work with manufacture specific networks.

Bluetooth Authentication

As OBDII only reads environmental data, there is no need for authentication. Third-party tools will require a standardized authentication procedure as parameters are set in the vehicle. OEM tools have access to very sensitive settings as engine tuning, drive-line parameters, etc. but authentication may better be handled on the application level than on the Bluetooth level.

Tool Class	Bandwidth	Latency	BT Access Procedure	BT Profile & SDP	BT Authentication
OBDII	15 kbit/s 100 msg/s	100 ms	YES	YES	YES
Third Party	60 kbit/s 400 msg/s	100 ms	YES	YES	YES
OEM	450 kbit/s 3000 msg/s	<10 ms	YES	NO	NO

Fig. 1: Requirements for diagnostic tools

Bluetooth performance

Reading the basic specification, Bluetooth shows an impressive performance of more than 700 kbit/s and a short latency. Fig. 2 below shows the expected performance of a transparent Bluetooth link between two CAN networks. Diagnostics on a CAN bus should easily be linked by Bluetooth, meeting even the high requirements of OEM tools. With DH 3 messages, even CAN messages with extended 29 bit identifiers, back-to-back at CAN at 1 Mbit/s, would be transferred within 2.5 ms.

Package type	Number of full CAN messages 1 Mbit/s		Max Latency time (ms)	% of CAN bandwidth		Direction
	Std	Ext		Std	Ext	
HV1 (SCO)	1	0	1.25	11	0	Bidirectional
HV2 (SCO)	2	1	5.0	5	3	Bidirectional
DM1 (ACL)	1	1	1.25	11	14	Bidirectional
DH1 (ACL)	2	2	1.25	22	28	Bidirectional
DM3 (ACL)	12	10	2.5	67	67	One-way
DH3 (ACL)	**19**	**15**	**2.5**	**100**	**100**	**One-way**
DM5 (ACL)	23	19	3.75	85	82	One-way
DH5 (ACL)	36	29	3.75	133	126	One-way

Fig. 2: CAN-Bluetooth-CAN, theoretical bandwidth and latency

However, theory and reality are not always the same. The figures above could only be achieved by a dedicated low level and transparent CAN- Bluetooth-CAN connection where Bluetooth just transports bytes from one side to the other as quickly as possible. Such a mode is not specified in Bluetooth. Any byte transportation from one application to another has to pass through a protocol stack using at least the L2CAP, but as it is data, preferably also RFCOMM. Great efforts have been made to make it simple for the application designer to use Bluetooth. He should not have to be concerned about the lower protocol layers. The price payed for this convenience is realtime performance. At Kvaser, we have used the Ericsson baseband code and worked directly on the L2CAP (Fig. 3). At Kvaser, we have used the Ericsson baseband code and worked directly on the L2CAP. The achieved bandwidth is about a quarter of the theoretical one. The figure using RFCOMM is estimated and found to agree with figures in the Bluetooth literature[1] and shows only some 10% of the expected bandwidth. This is acceptable for OBDII and a majority of

Theory and Practice
are not always the same

Achieved Bandwidth using Ericsson Baseband directly on L2CAP
125 kbit/s bidirectional CAN
(approx. 800 messages/sec)

Expected bandwidth using RFCOMM
42 kbit/s bidirectional CAN
(approx. 300 messages/sec)

Fig. 3: Achieved and expected bandwidth

[1]"Bluetooth: Connect Without Cables" by Jennifer Bray Charles Sturman, pg 348
ISBN: 0130898406

third party tools, but not for advanced OEM tools.

Although seeming disappointing, the bandwidth is not the very limiting factor: It is the latency. The protocol stack introduces latencies around 10 ms. The chain from one CAN network to another is shown in figure 4.

This does not seem to bad. Most diagnostic tools could live with that. However, the operating system in the Ericsson baseband application introduces a nasty jitter depending on the bus load. Test results for some different message rates are shown in figure 5.

Transmitting side

Delay µs	Σ delay µs	The process causing the delay
0	0	CAN message received from the CAN bus
10	10	Interrupt latency from CAN receive interrupt
40	50	Transfer from CAN to the Bluetooth stack
500	550	CAN profile equivalent
1000	1550	L2CAP
400	1950	HCI transfer
3750	5700	Baseband transmit processing
400	6100	Wait for slot in the air
625	6725	Broadcast on bluetooth

Receiving side

Delay µs	Σ delay µs	The process causing the delay
0	6725	Bluetooth message received
1250	7975	Baseband receive process
400	8375	HCI transfer
1000	9375	L2CAP
200	9575	CAN profile equivalent
100	9675	Store in CAN Controller for arbitration
325	10000	CAN transfer
0	10000	CAN message received at the remote module

Fig. 4: Approximate CAN latency chain using Ericsson baseband application and L2CAP

At low message rates, the downlink from the master gives a result around 10ms at low message rates, but already at 250 messages per second, we face an increased latency and a statistical jitter. The uplink latency from the slave is longer and even more unpredictable. It shows signs of some resonance problems with a bigger jitter at specific message rates already at low levels. Great concerns have to be taken in the choice of a protocol stack, especially for high end diagnostic tools.

Latency ms / Message rate msg/s	% messages master - slave slave-master						
	6-9	9-12	12-15	15-18	18-21	21-24	max
250	**32.5** 2.7	**60.7** 36.4	**6.6** 48.8	**0.2** 11.3	 .9	 .1	28.4 36.9
62.5	95.5 0	4.5 93.3	6.7				12.2 18.8
32.25	97.9 0	2.1 50.0	 0.3	 49.6			12.2 30.5
2	98.1 0	1.9 22.5	 31.6	 30.4	 15.3	 .1	11.3 30.3
.1	97.6 0	2.4 25.0	 29.1	 34.2	 11.6		10.1 25.2

Fig. 5: Measured latency at different message rates

Architectures

Bluetooth can be integrated into a vehicle in several ways. Three possible architectures are depicted below: One Bluetooth access point interfaced to all relevant buses in the vehicle, one access point per bus or a detachable access point, attached when needed.

Central access point

This is the architecture currently proposed by the Automotive Expert Group. The one point solution is mainly embraced for cost reasons. The car industry would not like to pay for

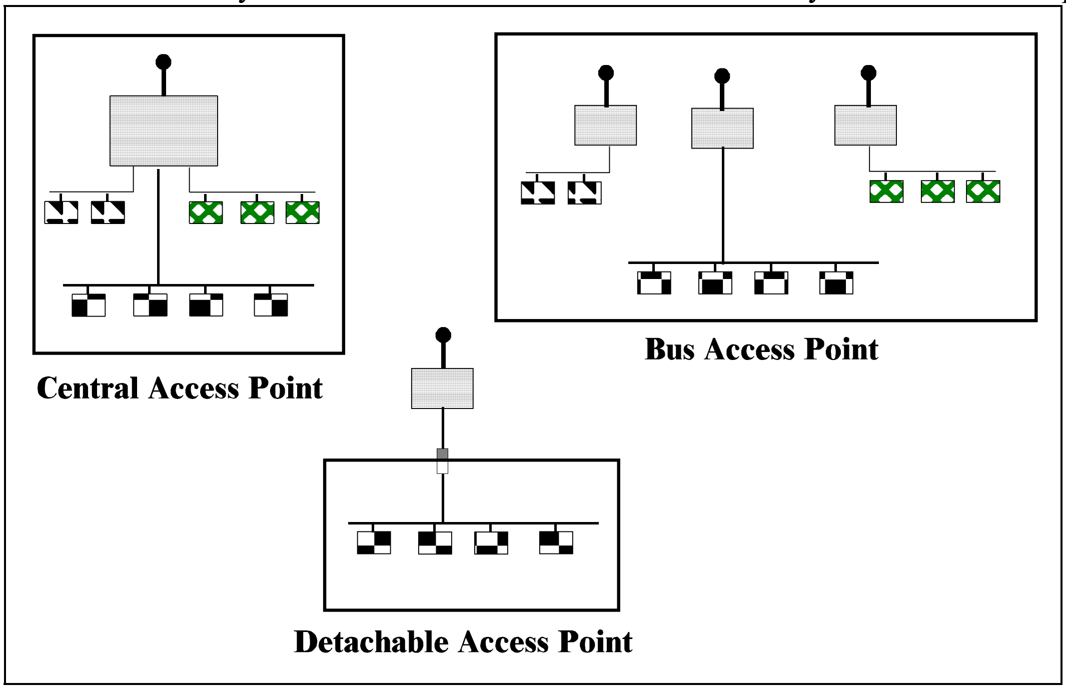

Fig. 6: Some possible architectures for Bluetooth vehicle integration

more than one Bluetooth unit, it should be standardized and cope with any aspect of the use in a vehicle. The main tasks can be classified into three groups: Infotainment, diagnostics and car specific tasks. Infotainment would include hands free operation of phones, audio, navigation, etc. and example on car specific tasks are adjustments of mirrors, seats, comfort equipment, operating locks, etc. Diagnostics has a high priority and a Bluetooth specific and bus independent protocol will be included. The bus independence is achieved by using ISO 14229 as an intermediate protocol.

There are two major drawbacks with this proposal. Firstly, it will require an advanced realtime operating system, among other things capable of arbitrating the service requests from the different buses and secondly, some required standards are still not in place, among them ISO 14229. Further, we at Kvaser, specialized in CAN, have faced big problems meeting the realtime demands for a simple CAN link and we assume that specialists of other realtime buses face similar problems. It will take a long time before this solution is ready for the market and it can be questioned if it ever will be cheap.

Bus access point

This is a simpler approach. The interface can be optimized to the needs of each bus. Existing solutions, in many cases already including diagnostic tasks, for respective buses can be used. Bluetooth acts as a transparent layer during application sessions. Communication between the buses is taken care of by existing gateways. This is a faster way to the market and it might show to be cost effective as the gateway functionality is much simpler due to a larger bandwidth, lower latency and no need for bus arbitration, compared with the central approach.

Detachable access point

This is the simplest approach, just a wire replacement. The diagnostic tool itself provides two Bluetooth units that are already paired. No specific Bluetooth standard for diagnostics is required. As soon as the Bluetooth unit is connected to the bus, a connection between the bus

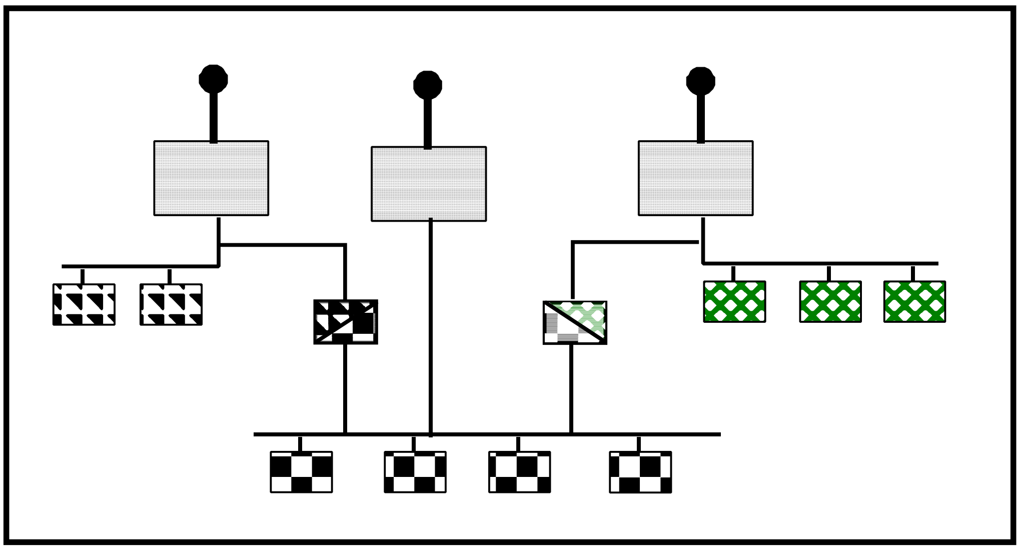
Fig. 7: Bluetooth bus access point and bus gateways

and the tool is established. During runtime, the Bluetooth connection is transparent to the bus communication. However, additional latency is introduced and the bandwidth might be lower than for the replaced wire connection. Such a solution ought to be acceptable for diagnostics during production and for service by manufacturer approved service stations. This is the current WAVEcan approach by Kvaser.

The ideal Bluetooth concept for diagnostics

The ideal Bluetooth concept for diagnostics should show the following qualities:

1. Low cost
2. Performance as needed
3. Standards as needed

To reach these goals we must have a low cost, high performance Bluetooth unit that can be used for any application. Then it has to be stripped from everything not needed by everyone, i.e., a concept like CAN. CAN provides the basic features needed by any controller network, but remaining features required have to be provided by an additional higher layer protocol. The basic Bluetooth block should take care of the RF part and the essential part of the baseband protocol that is of no or minor interest for any application but essential for the Bluetooth functionality. Examples on that are hop synchronisation, whitening, error recognition and correction, signal strength measurement, the clock, some basic handshaking etc. Some information should be readable from the application side as the SSI value, the clock, the BER value, etc. and the output power should be possible to control. The main task is to take bytes as quickly as possible from one side to the other through a byte oriented interface. Any other features as L2CAP, RFCOMM, authentication, master/slave decision, piconet and scatternet establishment, etc. should be taken care of by higher layers. The specification should be layered with optional features (fig 8.) to simplify the implementation. Those who need them, implement them. When full interoperability is required, the whole stack is implemented. The proposed centralized architecture as well as bus specific and detachable architectures could make use of the very same "Bluetooth engine" (fig. 9).

For a diagnostic tool a lot of simplifications can be made. The tool can always be the

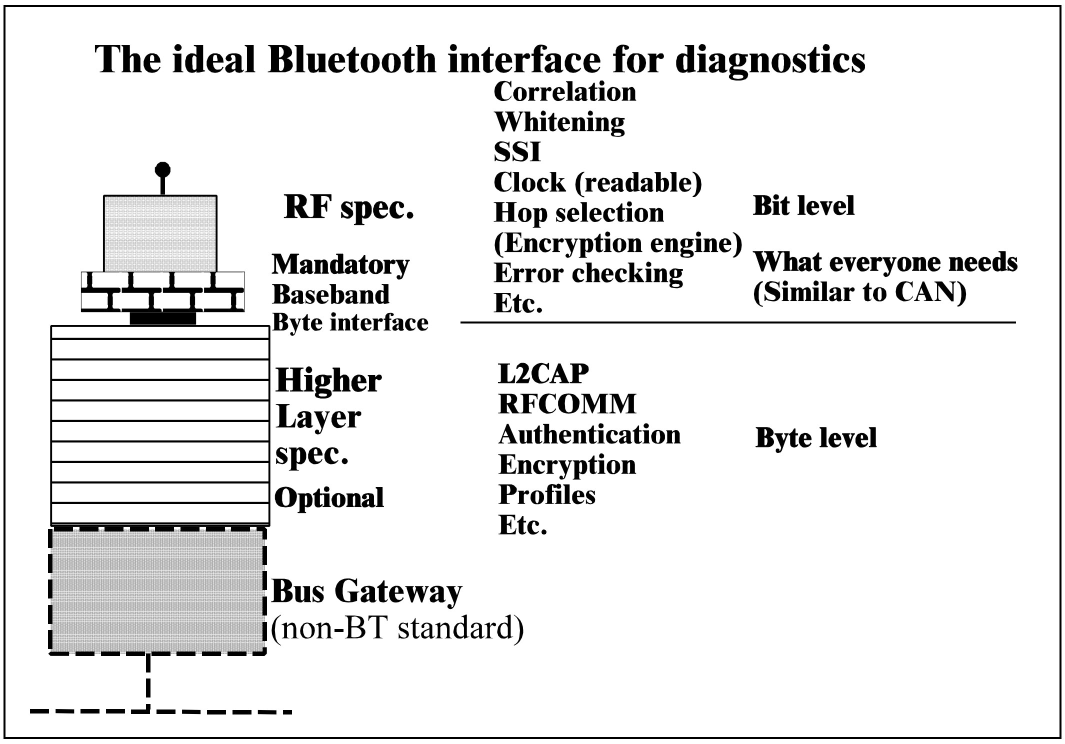

Fig. 8: Concept for an ideal Bluetooth interface for diagnostics

master and when the connection is established, no voice channels are needed and there is no need to look for other units until the session is over. After the connection is established, it should be possible to switch to a transparent mode where the applications take over the responsibility for safety and security. In a sense, we have these qualities in the voice channels, why should we not have them for a data channel?

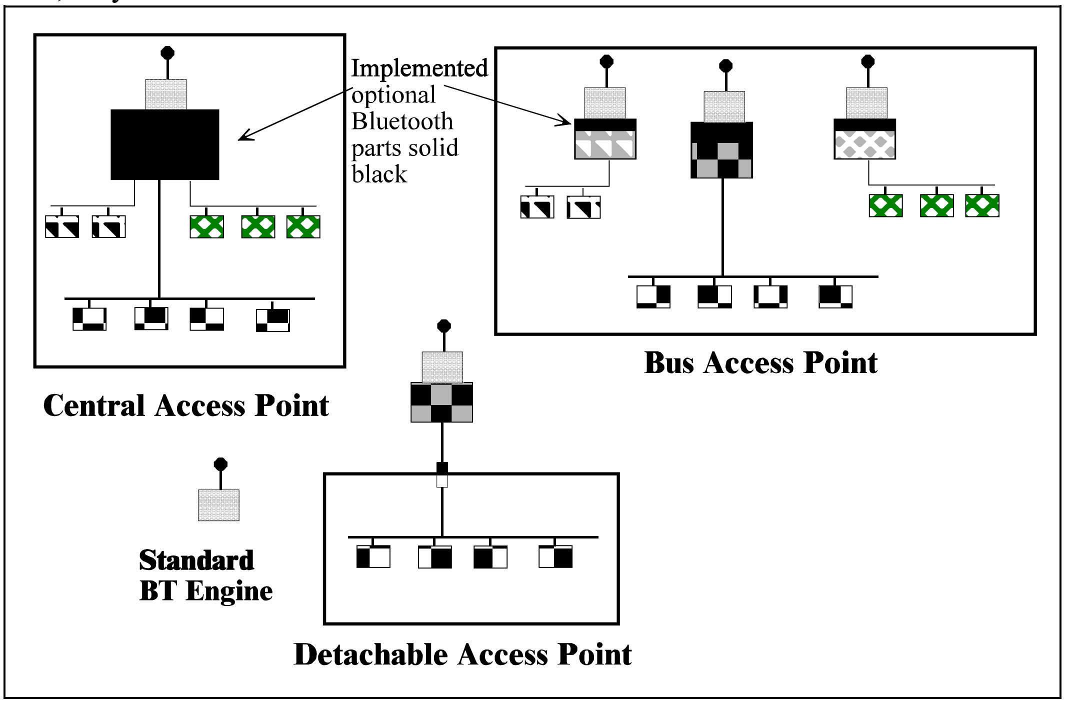

Fig. 9: Ideal architectures for diagnostics.

A solution like the proposed one would open up for other lowcost devices that are specific to a particular vehicle. The most obvious one is the key, but there are several other units where it would be good to have a wireless connection, e.g., seat adjustments, door units, etc. The design would be simplified if all units in a car would use the RF environment in the same way. It will be easier to control the available RF bandwidth in the car, and the bandwidth will soon be a headache for the designers.

One essential thing has to be added to the Bluetooth standard: How to identify a vehicle. The AEG has proposed to use the VIN (Vehicle Identification Number) and this looks as a good solution. Most other things are already covered in the Bluetooth specification or in existing wired solutions.

Summary

Theoretically, a Bluetooth link could transfer messages to or from a CAN network using the full bandwidth of 1 Mbit/s and with less than a 3 ms message delay. However, due to mandatory higher layer in the Bluetooth protocol stack, far lower performance should be expected. Transfer through the RFCOMM would result in less than 42 kbit/s. Applications using the Ericsson baseband directly on the L2CAP layer has shown to cope with a 250 kbit/s CAN network unidirectionally and 125 kbit bidirectionally. At low message rates, the average latency is in the order of 10 ms from master to slave and 20 ms in the reverse direction but the jitter is extensive and some messages show delays of 35 ms or more. For non-realtime critical car diagnostics, like OBDII and KWP2000, this is OK, but not for diagnostic tasks requiring realtime responses on specific messages.

The currently proposed architecture for Bluetooth in cars is a single access point. Any bus in the vehicle shares this Bluetooth resource. The vehicle access point will include not only existing Bluetooth services but also car specific ones, e.g., diagnostics, comfort adjustments, locks, etc. and an extended temperature range. The main reason for a single point solution is cost. This may be disputed as the gateway will be very complex and the throughput low.

An alternative architecture would be one Bluetooth access point for each bus. The access point can then be optimized to the needs of the respective bus and the inter-bus traffic can be handled by the ordinary gateways. Existing diagnostic software running on the respective bus can be used with minor modifications.

An optimal solution would be to restructure Bluetooth in a low-level mandatory part including what everyone needs and leave the rest to higher layer protocols. CAN is an example on such a structure. We would then have a “Bluetooth engine” were anything not needed by every application is stripped off, connected to a module optimized for its tasks by a simple byte oriented interface. With such an architecture, the mandatory part can be implemented in a small and cheap module, used by any application. The proposed single access point can still be accomplished, but other solutions requiring higher performance or lower price could be embraced under the Bluetooth umbrella. “Bluetooth engines” would have a vast market and then a low price.