Traffic Shaping to Reduce Jitter in Controller Area Network (CAN)

Robert Davis[1] and Nicolas Navet[2]

[1]*Real-Time Systems Research Group, Department of Computer Science, University of York, UK*

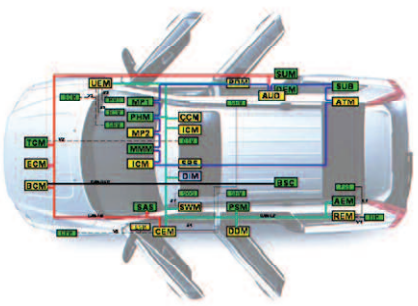
[2] *INRIA / RTaW, Nancy, France*

Controller Area Network (CAN)

- Simple, robust serial communications bus for in-vehicle networks
- Originally developed by BOSCH in 1983 standardised in 1993
- Average family car now has approx 25-35 Electronic Control Units (ECUs) connected via CAN
- Sales of microprocessors with CAN capability – approx 750 million in 2010
- Today almost every new car sold in Europe uses CAN

Typical use

- High speed (500 Kbit/sec) network connecting chassis and power train ECUs
- Low speed (100-125 Kbit/sec) network(s) connecting body and comfort ECUs
- Data required on different networks is **gatewayed** between them by an ECU connected to both



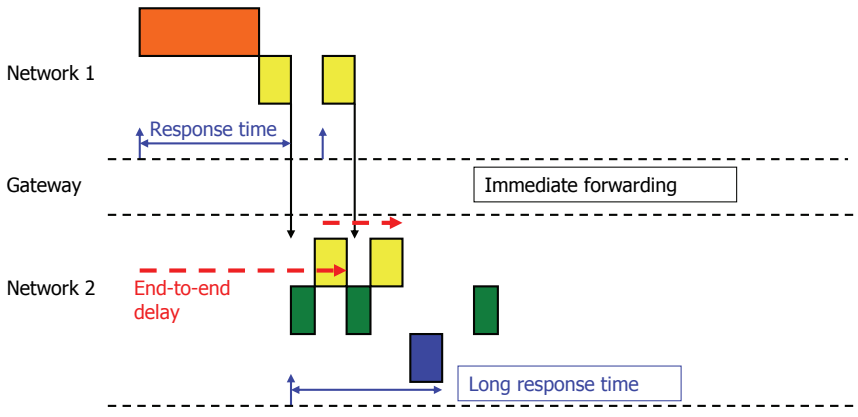
Requirements

- Eliminate or greatly reduce jitter on messages forwarded by the gateway
  - and hence reduce interference on lower priority messages on the destination network
- Do this for free!
  - No increase in the end-to-end response times of gatewayed messages

Practical Considerations for CAN

- No global time
- No time-stamping of when messages are actually received over CAN (some CAN devices have this facility, but it is not part of the standard)
- Efficient implementation needed
  - Single comms task handling all messages gatewayed between two networks
  - Compatibility with OSEK BCC1 Tasks (No blocking calls, single stack OS)

Gateway with immediate forwarding



Non-blocking Jitter Reduction (NJR) Policy

Comms task runs periodically. Period T_{COM} response time R_{COM}

NJR policy assumes a delay of $\Delta = T_{COM} + R_{COM}$ may have happened in detecting there is a message instance to process. ($\Delta \ll$ message period, T_m)

X_m is the earliest permitted time for queuing the next instance of message m

When an instance of message m is queued with a local time t, the next instance may only be queued at a time $X_m = t - \Delta + T_m$ or later

If the next instance arrives before this time, it waits in the message m FIFO buffer

The k-th subsequent instance of message m may only be sent at or after $X_m + kT_m$

No requirement to timestamp messages, reading local time in the comms task is enough

```
1  if the message m FIFO buffer is empty then return
2  get local time t
3  if t < X_m then return // too early to process message m
4  get instance of message m from buffer
5  queue instance on destination network
6  X_m = max(X_m, t − Δ) + T_m
```

Immediate forwarding

Comms task runs periodically

Max. time of Δ from receiving a message until it is queued on the destination network

Δ is typically the period T_{COM} plus the response time R_{COM} of the comms task

The queuing jitter for gatewayed messages on the destination network is:

$$J_m^{DEST} = J_m^{SRC} + R_m^{SRC} + \Delta$$

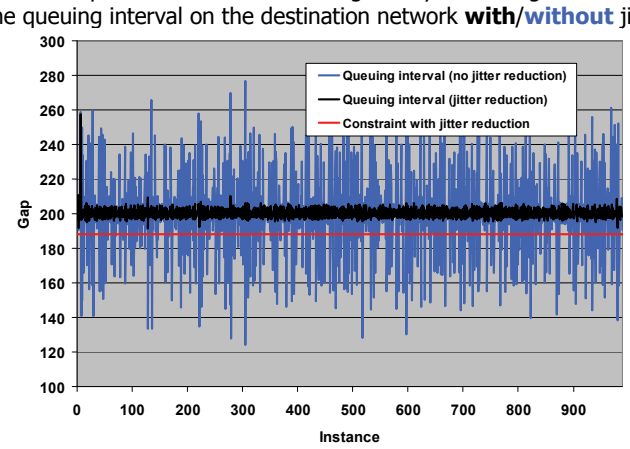
where J_m^{SRC} is the queuing jitter of the message on the source network and R_m^{SRC} is the worst-case response time of the message on the source network

The large inherited queuing jitter J_m^{DEST} can negatively impact the real-time performance of the destination network

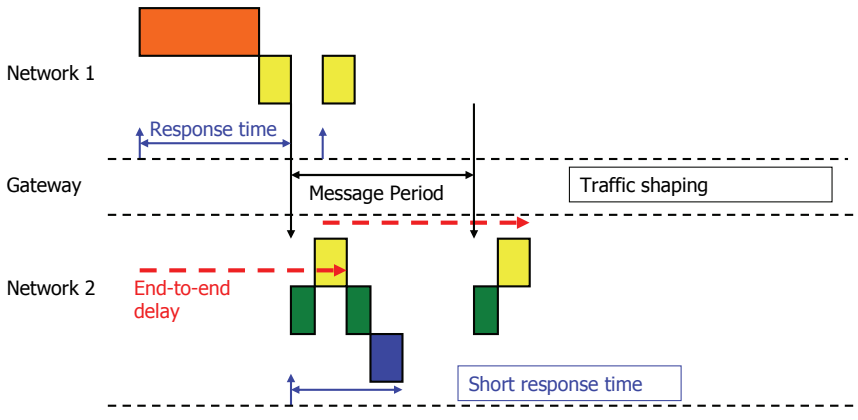
Jitter Reduction

NETCARBENCH data (125 Kbit/sec bus, 145 messages, Periods from 50 to 2000ms)

Graph shows the separation of instances of a gatewayed message with a 200ms period in terms of the queuing interval on the destination network **with**/**without** jitter reduction



Gateway with traffic shaping



NJR Policy Performance v. Immediate Forwarding

Longest time before a message instance is queued on the destination network

- Unchanged at $J_m^{SRC} + R_m^{SRC} + \Delta$

Queuing jitter on destination network

- Reduced from $J_m^{SRC} + R_m^{SRC} + \Delta$ to just $\Delta + R_{COM}$

NJR Policy

Reduces jitter and hence interference on lower priority messages

Simple efficient implementation

- Requires only a single free-running timer counter for all gatewayed messages
- Can be implemented in a non-blocking periodic task (OSEK BCC1)
- Compatible with single stack RTOS

Can be adapted to account for clock drift between networks

Traffic Shaping to Reduce Jitter in Controller Area Network (CAN)

Robert Davis[1] and Nicolas Navet[2]

[1]*Real-Time Systems Research Group, Department of Computer Science, University of York, UK*

[2] *INRIA / RTaW, Nancy, France*