Addressing the hard problems of automotive Linux: networking and IPC

Alison Chaiken
Mentor Embedded Software
Fremont CA USA
alison_chaiken@mentor.com

April 2, 2013
Outline

- Status of Linux in automotive
- Automotive InterProcess Comms (Intranet)
- V2V and V2I networking (Ravi)
- Summary
# Current Public Status of Automotive Linux

<table>
<thead>
<tr>
<th>Carmaker</th>
<th>Confirmed Operating system</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fiat-Chrysler Blue&amp;Me, Kia Uvo</td>
<td>Microsoft Windows Embedded Automotive</td>
</tr>
<tr>
<td>Ford (all?)</td>
<td>Microsoft MyTouch/Sync (+ OpenXC Android dongle and SmartPhoneLink)</td>
</tr>
<tr>
<td>General Motors “Cadillac User Experience”</td>
<td>Linux</td>
</tr>
<tr>
<td>Geely (China); Hawtai (China)</td>
<td>Linux: Moblin (MeeGo-Tizen precursor)</td>
</tr>
<tr>
<td>Renault R-Link</td>
<td>native Android</td>
</tr>
<tr>
<td>Jaguar Land-Rover, Tata, Toyota</td>
<td>Tizen (GNU/Linux)</td>
</tr>
<tr>
<td>Honda (Accord, Odyssey, Pilot), Audi (A8L, Q5, A6), BMW (7-series and M models), Chrysler, Daewoo, GM (OnStar), Hyundai, Land Rover, Porsche, Saab (9-3), Renault (SM7), Mercedes (S- and C-class)</td>
<td>QNX</td>
</tr>
</tbody>
</table>

**Linux Foundation** members: Toyota, Pelagicore, Symbio, Tieto

**GENIVI Alliance**: 160+ members including 11 automakers
GENIVI Alliance

- **Goals:**
  - reduce lock-in by suppliers.
  - reduce cost and TTM of new models.

- **Methods:**
  - Promote code reuse and attract outside contributors.
  - Focus on middleware: not a distro.

- *Mailing lists and #genivi on FreeNode.*

- **10 projects with released code** including AUTOSAR (Diagnostic Log and Trace) and IPC (AF_BUS D-Bus Optimization)
# OSI Networking Stack

<table>
<thead>
<tr>
<th>Layer</th>
<th>Function</th>
<th>Questions for this layer</th>
<th>On a PC's Ethernet</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Physical</td>
<td>Provides the electrical and physical specification</td>
<td>How many wires connect your processor to a peripheral? At what voltage? At what speed?</td>
<td>Ethernet cable</td>
</tr>
<tr>
<td>2. Data link</td>
<td>Describes how bytes flow over the physical wires</td>
<td>Do the bytes have parity checking? How many bits are sent and received in each frame?</td>
<td>Ethernet (802.xx)</td>
</tr>
<tr>
<td>3. Network</td>
<td>Gets a variable length of information (packets) from one place to another</td>
<td>How is each system addressed? How does the layer break up (and re-form) big blocks of data into amounts that can go over the communication pathway?</td>
<td>IP</td>
</tr>
<tr>
<td>4. Transport</td>
<td>Moves blocks of data in a reliable manner, even if those blocks are larger than the lower levels can handle</td>
<td>How do you count on data being received even when there is a glitch in the wires? How are errors recovered from?</td>
<td>TCP</td>
</tr>
<tr>
<td>5. Session</td>
<td>Manages a connection between the local and remote application</td>
<td>How to send this data from here to there?</td>
<td>Sockets</td>
</tr>
<tr>
<td>6. Presentation</td>
<td>Provides structure to the data, possibly encryption</td>
<td>How is the data organized when it is sent?</td>
<td>TLS and SSL</td>
</tr>
<tr>
<td>7. Application</td>
<td>Takes user interaction with the software and formulates a communication request</td>
<td>What command to send when a button is pressed?</td>
<td>HTTP</td>
</tr>
</tbody>
</table>

From *Making Embedded Systems* by Elecia White, with permission
Perlman’s View of ISO Layers

- 1: Physical
- 2: Data link: (neighbor to neighbor)
- 3: Network: create path, forward data (e.g., IP)
- 4: Transport: end-to-end (e.g., TCP, UDP)
- 5 and above: .... boring
Mixture of time-critical and best-effort networks

Copyright Renesas, "Introduction to CAN", with permission.
Diverse IPC mechanisms, Legacy Protocols

- TCP/IP?
- UDP/IP?
- RemoteProc?
- D-Bus?
- FlexRay
- EthernetAVB
- EtherCAT
- J1939...

or maybe

- Linux?
- RTOS
- AMBA/AXP
- SoC
- platform devices

Board

- Sensor
- I2C
- MCU
- SPI
- PWM
- Actuator
- Backup or "Mirror" video
- Driver webcam

Electronic Control Unit

- FPD-Link
- I2C

rear-seat display (LVDS)
Controller Area Network is 2-wire serial like I2C

Adapted from *Making Embedded Systems* by Elecia White, with permission
CAN Bus has security problems at Data Link Layer

Remote wireless exploit against stock vehicle: U. Wash. and UCSD collaboration

CAN will (eventually) phase out.

Linux security (SELinux, LXC, cgroups) offer limited help.
Challenges for Linux InterProcess Communication

• New for Linux: systems like cruise control incorporate both safety-critical and driver-facing components.

• Event-driven and timer-based traffic coexist on same network.
  – Will asynchronous networking provide QoS?

• “IP is the narrow waist of the Internet” BUT
  – TCP/UDP, even IP headers are too large for AUTOSAR.
    – Header compression (6LoWPAN) offers a solution?

• Many SAE, IETF, IEEE standards lack Linux support.
Linux (Ångstrom) CAN-Ethernet Gateway

Conclusions

- Linux ships in a small fraction of vehicles.
- Growing rapidly, helped by GENIVI.
- Problem: integration of legacy and modern networks.
- Security, safety and reliability remain a challenge.
- Headed incrementally towards autonomous vehicles:
  - much work left to do.
  - field is expanding rapidly.
Resources

- GENIVI open-source projects, mailing lists, #genivi IRC
- *Making Embedded Systems* by Elecia White
- IETF-ITS mailing list, Telematics News, Wired Autopia
- autosec.org and Ruggeri SAE paper
- LWN and H-Online (as always!)
- Mentor's AUTOSAR and Embedded Linux platforms
Automotive extravehicular networking: Ravi Puvvala, Savari Networks
Vehicles are a “network of networks”
IEEE: DSRC and Basic Safety Message

- Collision avoidance is primary motivation.
- USDot had RFC on PKE for V2X in 2012.
- How to issue revocable keys w/o trackability?

802.11p & 1609: DSRC, WAVE and WSMP

- 802.11p has **dedicated spectrum** at 5.9 GHz.
- Unlike other 802.11, **no BSS**.
- **New protocols**, e.g. DNS Geocasting
- **New use cases**, e.g. mobile routers
- VIN == MAC? or is VIN private?
- Jouni Malinen, 2012 Linux Wireless Summit, "Not yet implemented."

802.11p V2X routers

Linux-based

StreetWAVE™ Roadside Unit: Supports V2X Safety and Mobility applications using DSRC, 3G

BSD-based?

Commsignia
Anyone reminded of this Babel?
Housekeeping: IVI Jargon

- “OEM”: a car manufacturer
- “Tier 1”: a vendor who sells directly to OEMs
- “Tier 2”: a vendor to Tier 1s, who bundle components
- “ECU”: electronic control unit, 32- or 16-bit MCU running an RTOS
- “AUTOSAR”: ECU protocol incl. design methodology
- “ADAS”: advanced driver assistance system
**Linux won:**
- on servers and on handsets.

**Linux lost:**
- on desktops.

**Linux could lose in automotive:**
- QNX and Windows have the largest automotive base.
- QNX has fast IPC and works well on smaller MCUs.
- Most car CPUs run proprietary RTOSes.
CAN-Ethernet Gateway Demo


- ARM Cortex A8
- Linux Ångström v 2.6.28
- Stack “SocketCAN”
- Can and Ethernet on chip

Two independent threads. Uses socket paradigm as an abstraction to transparently copy messages on different physical layers.
Another view of automotive networks

courtesy AUTOSAR
SPI used as communication stack

Impact:
SPI interface module needed

Note:
two or more MCUs

Courtesy of AUTOSAR Consortium
## High-Bandwidth feasible buses overview

<table>
<thead>
<tr>
<th></th>
<th>IEEE 802.3</th>
<th>Ethernet Field busses</th>
<th>CAN-FD</th>
<th>Flexray</th>
<th>MOST</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Openness</strong></td>
<td>Open standard, high availability of PHYs and MACs</td>
<td>Usually open standard, some have custom PHYs and MACs</td>
<td>Under ISO standardization ISO-11898-2/6 transceivers</td>
<td>Under ISO standardization (or flexray.com) Expensive controllers</td>
<td>Closed Standard. Expensive Fiber wiring</td>
</tr>
<tr>
<td><strong>Bandwidth</strong></td>
<td>1000/100 Mbit</td>
<td>100 Mbit (not every fieldbus Gigabit ready)</td>
<td>Up to 8x CAN datarate→2Mbit for ISOBUS (theoretical)</td>
<td>Up to 10 Mbit</td>
<td>Up to 138Mbit (MOST 150)</td>
</tr>
<tr>
<td><strong>Hotplug capability</strong></td>
<td>YES</td>
<td>Depends on the field bus</td>
<td>YES</td>
<td>NO (attempts were made to enable)</td>
<td>YES</td>
</tr>
<tr>
<td><strong>Topology</strong></td>
<td>Star, Logical Bus, daisy chain, ...</td>
<td>Ring, Daisy chain, Star in some topologies</td>
<td>Physical Bus</td>
<td>Star</td>
<td>Ring or doubled ring, star feasible</td>
</tr>
<tr>
<td><strong>Safety certified</strong></td>
<td>NO (OpenSafety)</td>
<td>Many fieldbuses have IEC-61508 SIL3 certified Layer</td>
<td>??</td>
<td>NO</td>
<td>??</td>
</tr>
</tbody>
</table>

*From “SAE J 1939 Over Real Time Ethernet: The Future of Heavy Duty Vehicle Networks,” Ruggeri et al., Imamoter*
**GENIVI meets kernel: AF_BUS**

- Problem: D-Bus scales poorly, is resource-intensive and slow.
- Recent history of contention around IPC: **binder** in 2009
- **AF_BUS** is created by Collabora with GENIVI-funding.
- Implements a **new socket interface** based on AF_UNIX but with multicast capability.
- Rejected from mainline with rationale that IP sockets can provide needed performance.
  - Real-time IPC guarantees possible with IP?
- **AF_BUS** subsequently merged in **LTSI kernel 3.4.21**.
Automotive Architecture (H.-J. Mantsch, Mentor Automotive Networking)
Feb 2013: **Gnome Hackfest**

- New *in-kernel* D-Bus-based IPC is broached.
  - Will support Binder userspace API as well as D-Bus.
  - And others (0MQ, RabbitMQ, etc.)?
- **Endorsed** by D-Bus (Pennington) and *kernel* (GKH) contributors.
- **Victory** for GENIVI: in-kernel D-Bus optimization is coming!
- Not viewed that way by everyone . . .
- **brcmsmac vs. b43** (Broadcom) redux?