



Connection Manager for embedded systems

Linux Collaboration Summit

Marcel Holtmann
Open Source Technology Center



About me

- BlueZ maintainer
- OpenOBEX and Gnokii project
- WiFi, Ultra-Wideband and WiMAX hacking
- RFID and NFC research
- Open Source CERT - www.ocert.org
- ConnMan project
- Open Source Technology Center at Intel



Agenda

- History and ideas behind ConnMan project
- Lessons learned so far

Doing things a little bit different

Connection management 101

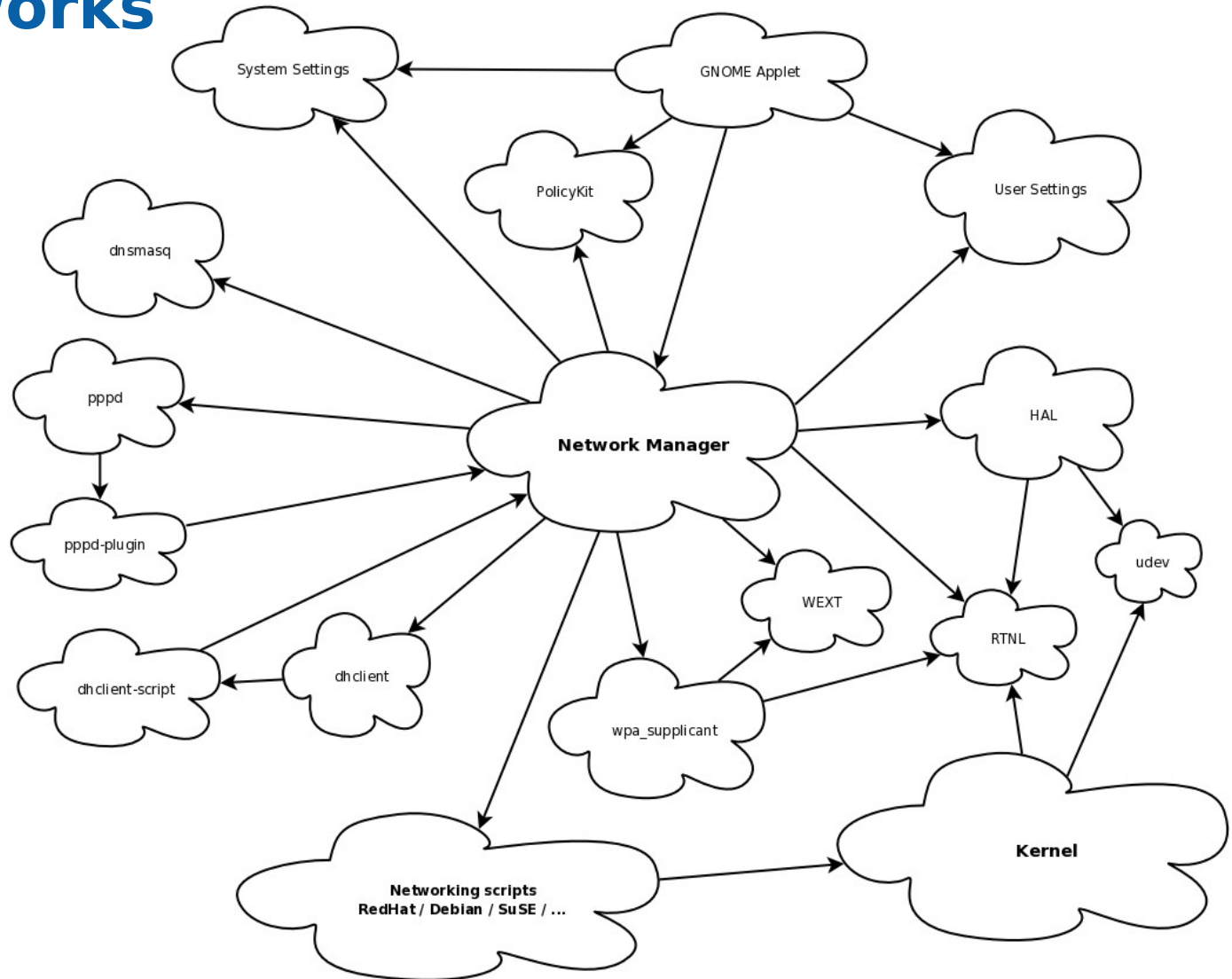
- What do you expect from a Connection Manager?
- The world is more complex than Ethernet and WiFi
- Linux networking sometimes still feels like the 80ties

- Welcome the 21st century, but don't forget your roots

Words about Network Manager

- Network Manager is a great project, but ...
- Very complex design
- Large dependency list
- Too much decision making depends on the UI
- Not easy to extend
- Tries to workaround Linux distributions problems
- Too much GNOME like source code

How it works



Starting from scratch

- UI should be easy to replace (branding/customization)
- Adding new technologies should be simple
- Replaceable common components
- Minimal number of dependencies
- Extendable via plugins
- Handle command line access gracefully
- Ready for embedded usage

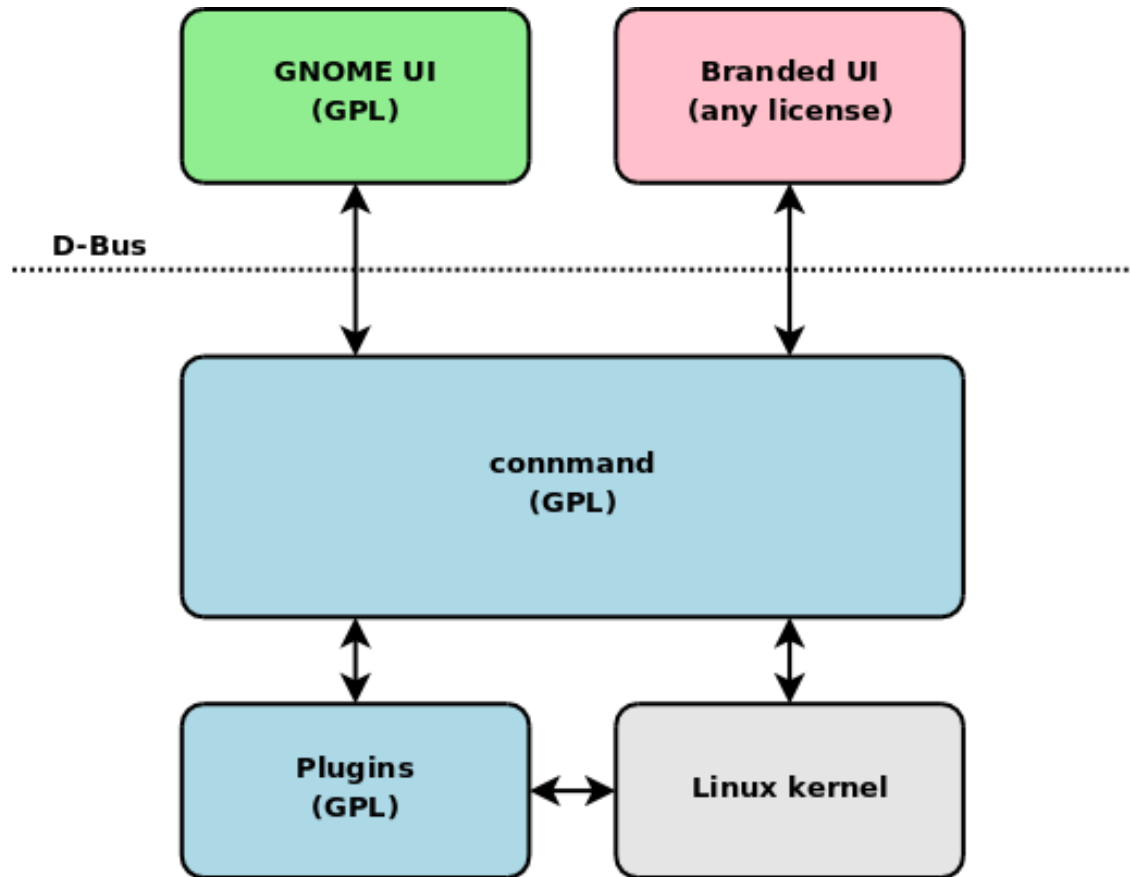
Mandatory dependencies

- C library with dynamic loader
- GLib core module
 - For mainloop, string handling etc.
 - No usage of GThreads and GModule
 - GThreads can be enabled if plugins require threading
- D-Bus low-level library
 - No GObject based D-Bus bindings
 - Copy of libgdbus helper library is included
- udev library for device detection and handling (optional)
- PolicyKit for D-Bus access authorizations (optional)

Full network control

- One system wide entity that controls all networks
- Tools like ifconfig and route are just for configuration
- No networking scripts or hacks
- Start as early as possible
- Setup of loopback device, hostname etc. by the daemon itself instead of init process
- Take ownership of DNS resolver
- Support restart of components and technologies

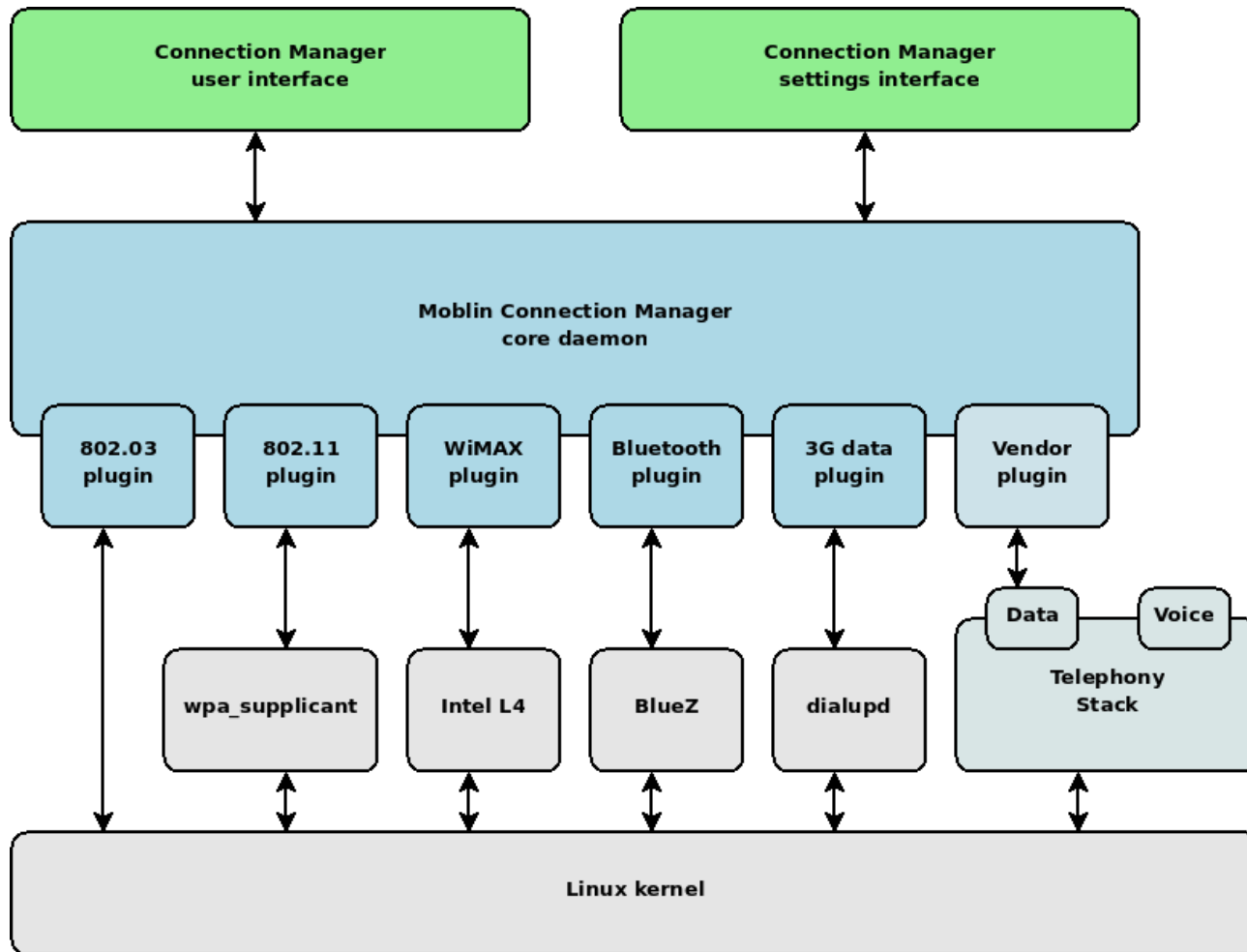
Architecture and licensing



User interfaces

- User interfaces communicate with the core daemon using the D-Bus system message bus
- Multiple interfaces can be running at the same time
- Possibility to split tasks into separate applications
- All state changes are propagated via D-Bus signals to support fully asynchronous applications

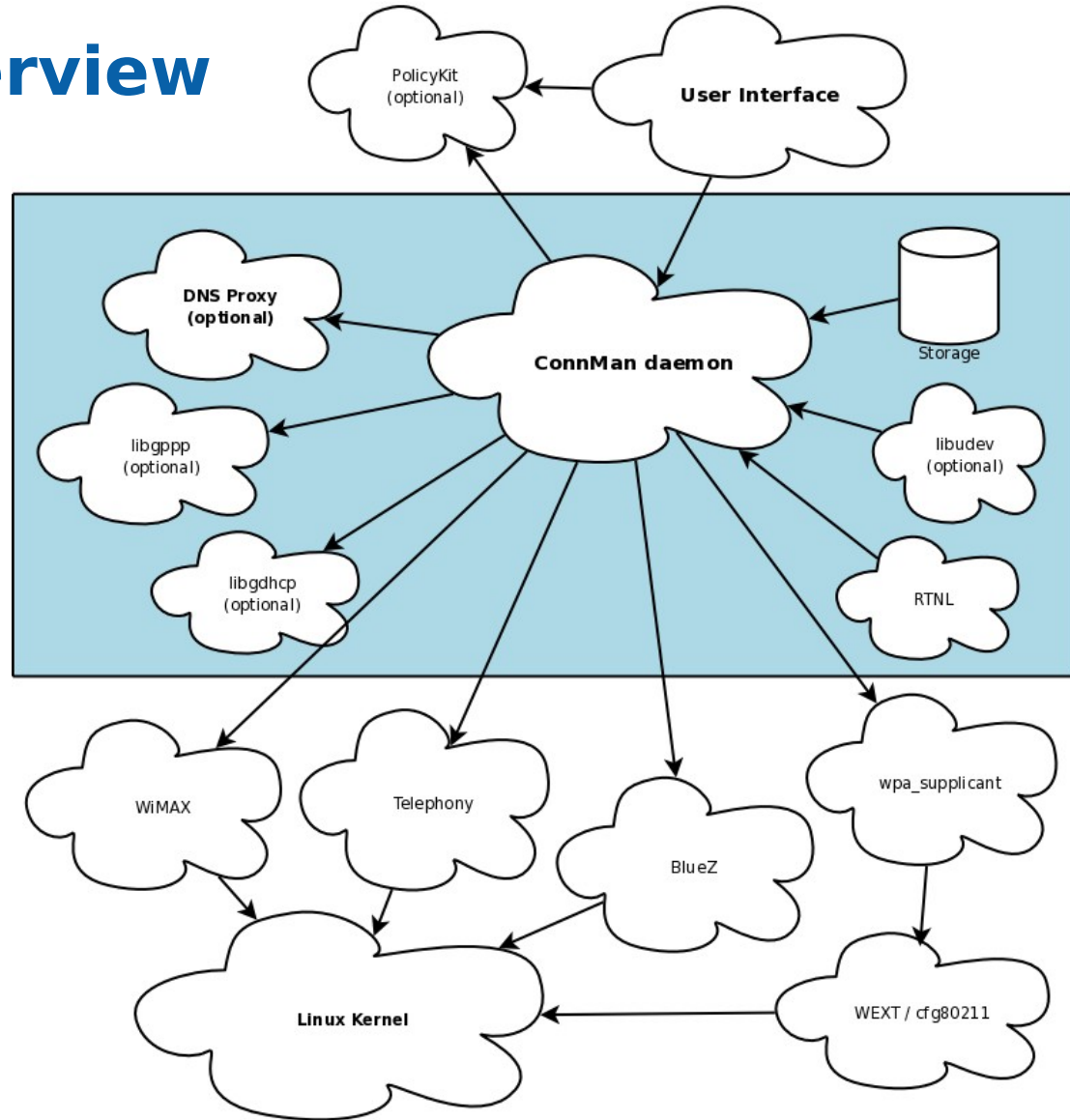
Support multiple technologies



New technologies

- Extending ConnMan with new technologies is simple
- Hardware detection can be done by RTNL and udev or also via vendor plugins if needed
- Plugins can register drivers for vendor devices
- Common elements like DHCP, IPv4 and resolver support can be easily re-used
- Multiple implementation of DHCP, resolvers etc. are encouraged

ConnMan overview



ConnMan internals

- Fully generic element based system
- Abstractions for Devices, Networks and Connections
- More advanced Service interface
- Helpers for Resolvers, Security Policies and Storage
- Implementation details are moved plugins
- Priority based assignment that allows easy overriding of default behavior

Fixing the DNS problem

- DNS is synchronous and one server at a time
- The whole concept of /etc/resolv.conf is broken
- Name Service Switch (NSS) helps, but has also its limits
- DNS caching services are mandatory
- Controlling DNS caching doesn't exist
- Solution is to include DNS proxy inside ConnMan
- Keep legacy applications working

Integrating DHCP and ZeroConf

- DHCP and link-local configuration needs to be merged to work without race conditions
- Most applications can't handle multiple addresses on an IPv4 interface (even if Linux does support it)
- Use Avahi for mDNS via the DNS proxy of ConnMan
- No more NSS tricks are needed and all applications are using the same call path for DNS request

Taking on the system time

- Time server information are delivered via DHCP
- Simple systems should just trigger a remote NTP update every now and then
- Update list of time sources on system that have ntpd or similar time-synchronization infrastructure
- Full control of time updates and ntpd is important for embedded system to save power
- Timezone information can also be used for regulatory compliance (mostly WiFi)

Dealing with HTTP proxy settings

- The `http_proxy` environment variable is not a viable solution for UI based systems
- Every application or desktop specific proxy settings is also not helpful
- Proxy details can be distributed via DHCP
- Handle these details inside ConnMan is a challenge, but it is the right way to go

New ways for the user interface

- Hierarchical and per technology interfaces are just too complex for the UI and the end user
- ConnMan does provide a low-level interface that is similar to Network Manager
- These interfaces are just not the solution
- User are not technology aware and they should not be

A flat service based interface

```
+-----+
| Ethernet |
+-----+
| Bluetooth phone |
+-----+
| Guest      (strength 90, none) |
+-----+
| My WiFi AP  (strength 80, wpa2) |
+-----+
| Clear WiMAX (strength 70) |
+-----+
| Other AP    (strength 70, wpa2) |
+-----+
| Friends AP  (strength 70, wep) |
+-----+
| Other WiMAX (strength 50) |
+-----+
```

Sorting the services

```
+-----+
| My WiFi AP      (strength 80, wpa2) | order=1 - favorite=yes
+-----+
| Ethernet        | order=0
+-----+
| Guest           (strength 90, none) | order=0
+-----+
|                 |
```

```
+-----+
| Ethernet with cable | order=1 - favorite=yes
+-----+
| Ethernet without cable | order=0 - favorite=no
+-----+
| Guest               (strength 90, none) | order=0
+-----+
|                 |
```





Connections and default route

```
+-----+
| Ethernet                               | order=2 - connected=yes
+-----+
| My WiFi AP      (strength 80, wpa2)   | order=1 - connected=yes
+-----+
| Guest           (strength 90, none)    | order=0
+-----+
|
```

```
+-----+
| My WiFi AP      (strength 80, wpa2)   | order=2 - connected=yes
+-----+
| Ethernet                               | order=1 - connected=yes
+-----+
| Guest           (strength 90, none)    | order=0
+-----+
|
```

Interface details

▼ Object Paths

- ▶  /
- ▶  /dev_00_1F_16_16_C6_D1
- ▶  /profile/default
- ▼  /profile/default/ethernet_2

▼ Interfaces

- ▼ org.moblin.connman.Service

▼ Methods

- Connect()
- Disconnect()
- GetProperties() → (Dict of {String, Variant})
- MoveAfter(Object Path)
- MoveBefore(Object Path)
- Remove()

▼ Signals

- △ PropertyChanged(String, Variant)

```
ActiveProfile
  /profile/default
Devices
  /dev_00_1F_16_16_C6_D1
  Name = Ethernet
  Powered = true
  Priority = 11
  Policy = auto
  IPv4.Method = dhcp
  Address = 00:1F:16:16:C6:D1
  Interface = eth0
  Type = ethernet
Connections
Policy
  single
State
  offline
Profiles
  /profile/default
  Name = Default
  Services = [ ethernet_2 ]
Services
  /profile/default/ethernet_2
  Favorite = 0
  State = idle
  Type = ethernet
OfflineMode
  false
```


The almost totally stupid UI

- ConnMan will do the sorting of services
- User choices will be stored in a central place
- Status updates and changes are propagated via signals
- The user doesn't have to care about technologies
- Services are either picked by name or logo
- UI is responsible for making the interface look sexy

Long term future work

- Command line interface
 - Tools like ip, iw and cm will be the future for admins
- Automatic web based login/authentication
 - Extra step before connection becomes available
 - Accepting terms and conditions in an unified way
- Connection sharing
 - Turn your phone or laptop into an access point
 - Bluetooth is possible, but a WiFi access point might be just simpler
- VPN support

Questions?

- Website
<http://www.moblin.org/projects/connection-manager>
- Snapshots
<http://ftp.moblin.org/connman/releases/>
<http://www.kernel.org/pub/linux/network/connman/>
- Mailing list
connman@moblin.org
- #connman on freenode.net

Legal information

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY RELATING TO SALE AND/OR USE OF INTEL PRODUCTS, INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT, OR OTHER INTELLECTUAL PROPERTY RIGHT.

Intel may make changes to specifications, product descriptions, and plans at any time, without notice.

All dates provided are subject to change without notice.

Intel is a trademark of Intel Corporation in the U.S. and other countries.

*Other names and brands may be claimed as the property of others.

Copyright © 2007-2009, Intel Corporation. All rights are protected.



