Profiling and Optimizing D-Bus APIs

Will Thompson
Collabora Ltd.
Profiling
Why?

- Why does it take so long to sign in/join #ubuntu with Empathy?
- Does some signal keep waking up lots of processes?
- See how an API works out in practice
- Getting your head around a new framework
How?

- Slow D-Bus calls won’t show up in generic profilers
  - (you are making async calls, right?)
- Need to study several processes’ interaction
- Timing of messages on the bus is handy
- Let’s use a D-Bus-specific tool.
Statistics

- *bustle-count* summarizes number of calls to methods and emissions of signals
- *bustle-time* shows total and average time spent in each method
- *bustle-dot* plus GraphViz shows component dependencies
Wishlist

- Equivalents of *bustle-count* and *bustle-time* in the UI
- Searching and filtering
- Spacing events in proportion to time
- Showing the contents of messages
/getinvolved

• Get it:
  • http://willthompson.co.uk/bustle/
  • git://git.collabora.co.uk/git/user/wjt/bustle.git
  • http://git.collabora.co.uk/?p=user/wjt/bustle.git

• “Bug tracker”:
  • will.thompson@collabora.co.uk
Optimizing
Reduce roundtrips

“M40 at Night”. by timo_w2s
Standard access pattern

• Fetch initial state
• Subscribe to change notifications
• Do whatever you wanted to do
Standard access pattern

- Fetch initial state
- Subscribe to change notifications
- Do whatever you wanted to do
Use D-Bus properties, not accessors

- Get(iface, "Foo") rather than GetFoo()
- GetAll(iface) rather than Get(), Get(), Get()
Telepathy’s Group interface

- Used for chatrooms, calls and contact lists.

- Three conceptual lists of participants:
  - Members
  - LocalPendingMembers
  - RemovePendingMembers

- Used to have lots of separate accessors.
- Now just GetAll("...Channel.Interface.Group")
ConnMan’s properties

- Reinvents DBus.Properties once per interface
- `GetProperties() → a{sv}`
- `SetProperty( s: name, v: value ) → ()`
- `PropertyChanged( s: name, v: value )`
Supply information up front

- FooAdded / NewFoo signals can contain a snapshot of properties
- Lets UI react immediately without needing to get properties first
NewChannel $\rightarrow$ NewChannels

- NewChannel contains:
  - the contact/chatroom’s integer handle (not name!);
  - whether it’s a call, text conversation, …

- NewChannels adds:
  - the contact/chatroom’s name;
  - whether you requested the channel;
  - who invited you;

- … and more, as appropriate:
  - the incoming file’s name, type, size, etc.;
  - whether the call is (initially) audio/video/both;
  - the application being shared over the tube;
  - …
foreach
descriptions = [ programme.Describe(talk) for talk in gcds_talks ]
descriptions = programme.Describe(gcds_talks)
Fetching metadata from Tracker

- Get( s: service_type, s: uri, as: keys ) → as: metadata
Fetching metadata from Tracker

- Get( s: service_type, s: uri, as: keys ) → as: metadata
- GetMultiple( s: service_type, as: uris, as: keys ) → aas: metadatas
- In extreme cases (thousands of items — your photo collection/mp3 library/mail archive?): 2.5× faster.
Telepathy contacts’ metadata

- InspectHandles( ..., au: handles ) → as: identifiers
Telepathy contacts’ metadata

- GetAliases( au: contacts ) → a{us}: aliases
- GetPresences( au ) → a{u(uss))}: presences
- GetKnownAvatarTokens( au ) → a{us}: tokens
Telepathy contacts’ metadata

- GetLocations
- GetContactCapabilities
- GetContactInfo
- ...
Contacts interface

GetContactAttributes(
  au: Handles,
  as: Interfaces )
→ a{ua{sv}}: Attributes
Contacts interface

```c
void
void tp_connection_get_contacts_by_handle (TpConnection *self,
    const TpHandle handles[],
    const TpContactFeature features[],
    TpConnectionContactsByHandleCb callback);

typedef void
(*TpConnectionContactsByHandleCb) (TpConnection *connection,
    TpContact *contacts[]);
```
Integer IDs?!

“You don’t need to invent a new kind of integer ID! That is what the object path is, it’s an id for an object.”

— Havoc Pennington, Best D-Bus Practices
Don’t just wrap your C/Java/… API
Awesome

- /
- org.naquadah.awesome.awful.Remote¹
  - ¹ formerly org.awesome.awful.Remote ...
- Eval( s: lua ) → ???
- Return type depends on what the Lua code returns!
Reducing wakeups
NameOwnerChanged

- Can wake you up on every name change, even if you don't care about the name
- Match \texttt{arg0=\texttt{name.you.care.about}}
- Maybe patch the daemon to warn you about unqualified matches?
Don't force information on clients

- Split information into conceptually self-contained interfaces
- Lets clients only monitor properties they care about
Client API

- Efficient D-Bus APIs can be inconvenient to use
- Good bindings help a lot
Tp::Proxy (and TpProxy)

- Create object; call_when_ready()
- Subsequent changes through normal Qt / GObject signals
Features

- Specify what you care about at construct time.
- Upgrade: “I now care about avatars”
Property change notification

- No standard mechanism
  ➔ no auto-generated bindings 😞
ConnMan

PropertyChanged( s: name, v: value )
Telepathy chatrooms

MembersChanged ( au: added,
    au: removed,
    au: local_pending,
    au: remote_pending )
Potential compromise

- PropertiesChanged( s: interface, a{sv}: values )
- IDL annotation for properties that use it.
Logging in with Empathy …

- Connecting
  - Network bound.
- Getting all the avatars
  - Free if they’re cached;
  - Network-bound if not.
- Getting the entire contact list
  - Hmm …
Contact list

- Separate objects for:
  - **subscribe**: contacts we’re subscribed to
  - **publish**: contacts subscribed to us
  - **deny**: contacts we’ve blocked
  - each named group

- Each one needs to become ready.
- Then, match up the members to existing Contact objects.
- Make any Contacts you don’t have (asynchronously).
- Sorry …
Slight improvement

- Separate objects for:
  - **subscribe**: contacts we’re subscribed to
  - **publish**: contacts subscribed to us
  - **deny**: contacts we’ve blocked
  - each named group
  - **stored**: superset of all the above
But we should fix it properly

- Roster object
- Property mapping contact handles to their state, with change notification
- Helper method to delegate to `GetContactAttributes`:
  - all contacts (for the contact list)
  - just those in state $x$
- (or similar)
Thanks!

- http://willthompson.co.uk/bustle/
- http://telepathy.freedesktop.org/
- #telepathy on Freenode
- http://www.collabora.co.uk/
Yes, Bustle is written in Haskell!

(What of it?)