IM-up your app!

Senko Rašić, Collabora

Facts

- Constant communication
- Multiple protocols and ways to contact people
- Multiple accounts for each person
- Rich communication (not just text)

Ideals

- Think about people, not accounts
- Don't care about implementation specifics
- Integrated with user workflow
- Invisible
- “With <user> do <action>”

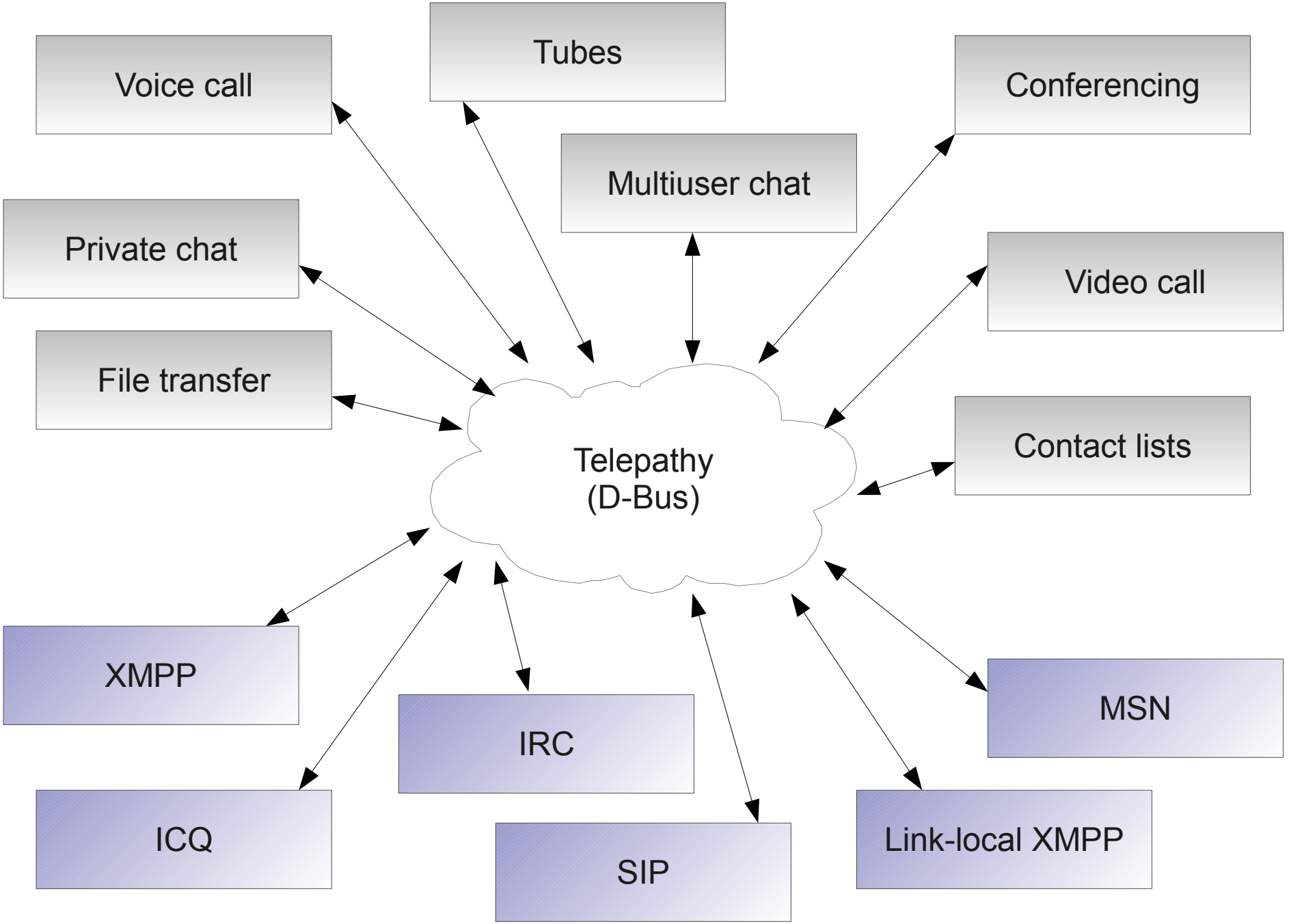
Solution

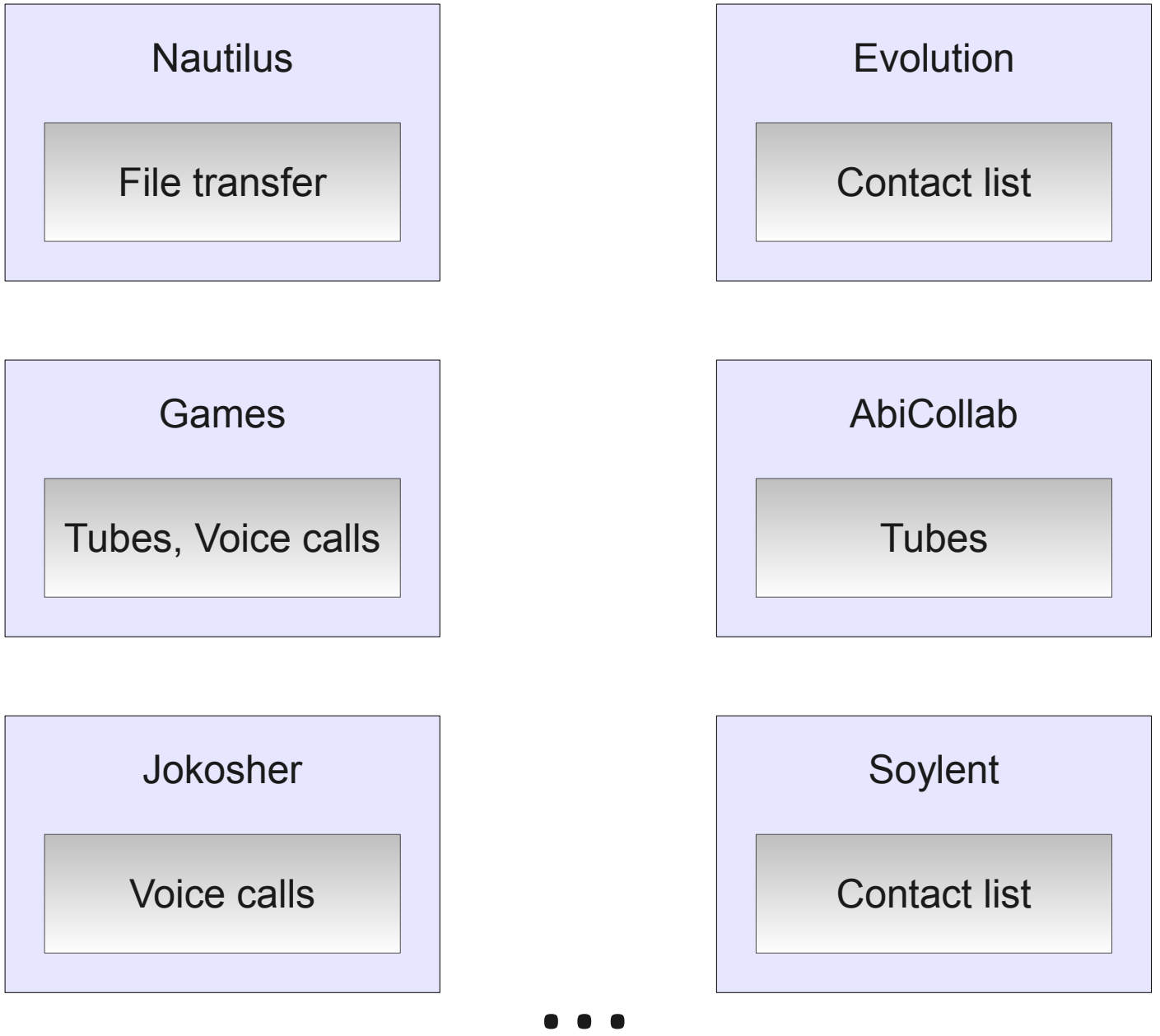
- Suport for communication as an integrated part of the desktop
- No separate IM app
- Any application can provide (related) communication features

E.g.

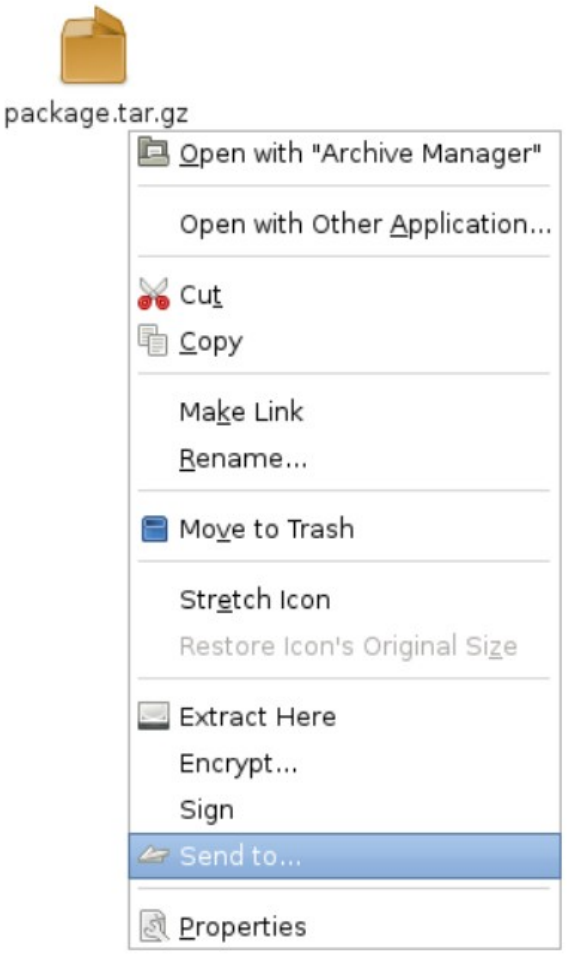
- Using user info from gnome-about-me
- Unified people browser instead of buddy lists
- Drag'n'drop files to users for file transfer
- Call people from audio recording application
- Collaborative work on an application

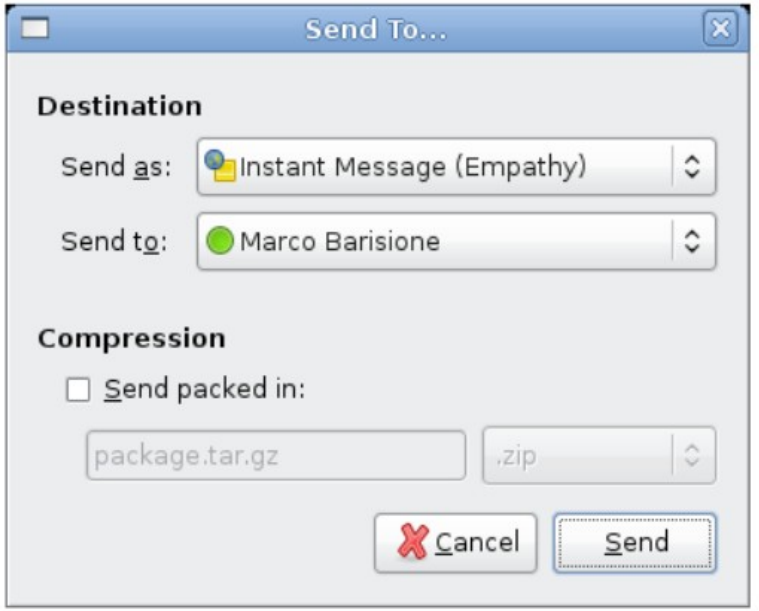
Telepathy

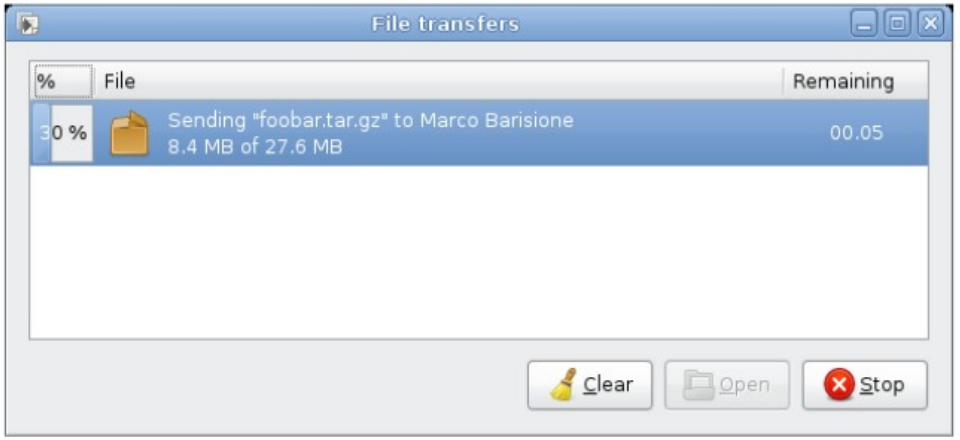

Voice call
Tubes
Conferencing
Multiuser chat
Private chat
Video call
File transfer
Telepathy
(D-Bus)
Contact lists
XMPP
MSN
IRC
ICQ
SIP
Link-local XMPP


Nautilus
File transfer
Evolution
Contact list
Games
Tubes, Voice calls
AbiCollab
Tubes
Jokosher
Voice calls
Soylent
Contact list

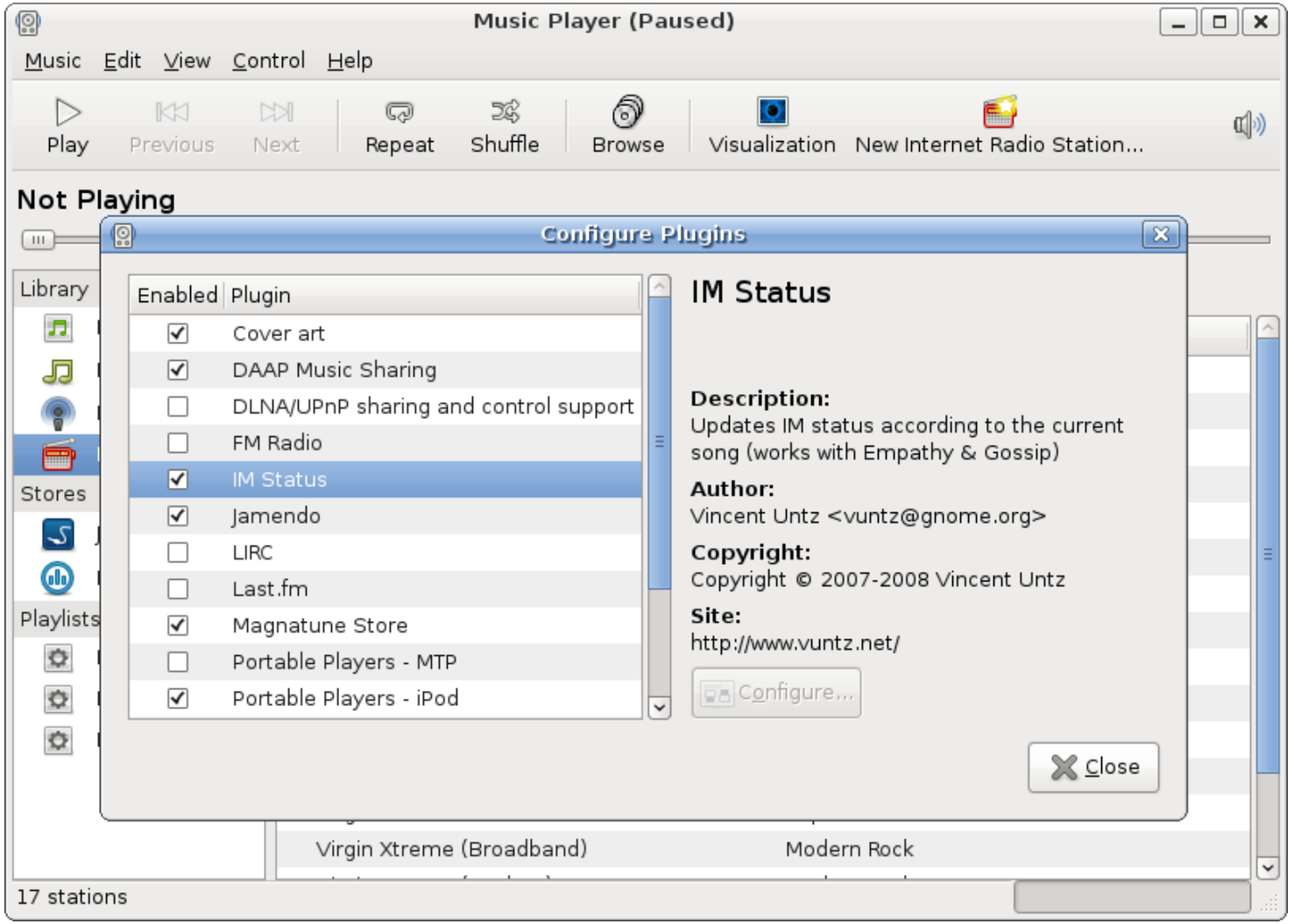
Telekinesis



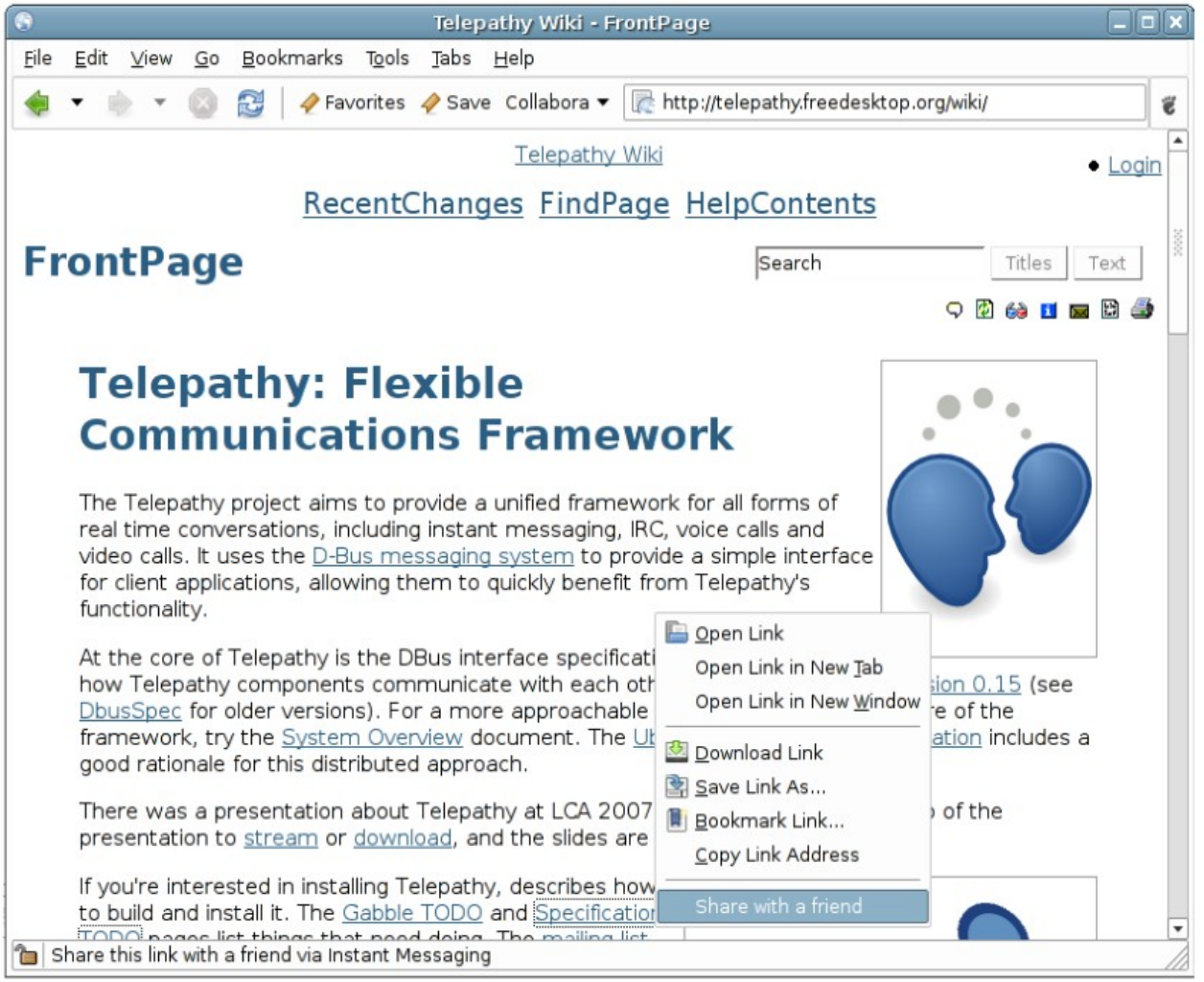




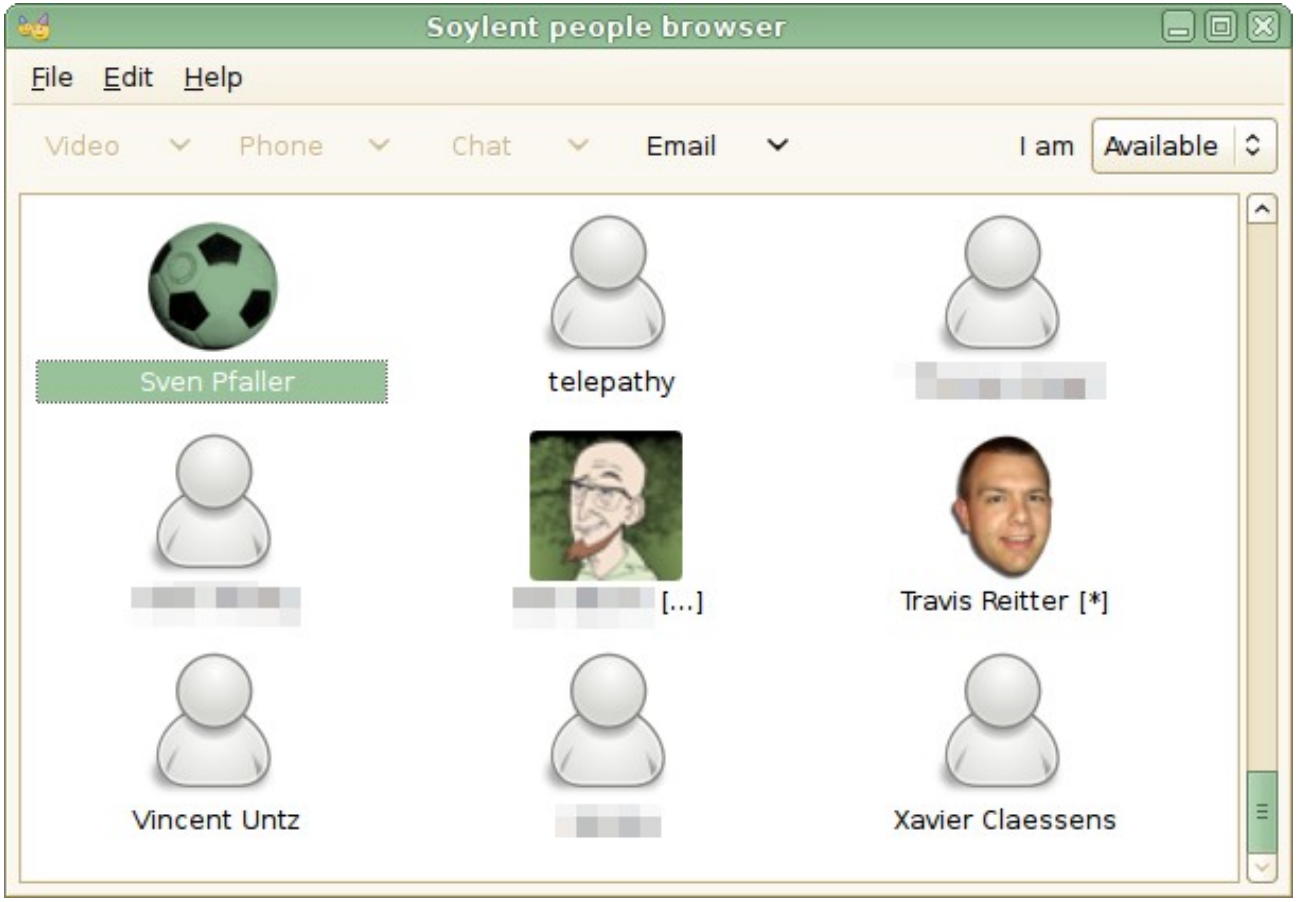
Rhythmbox status plugin



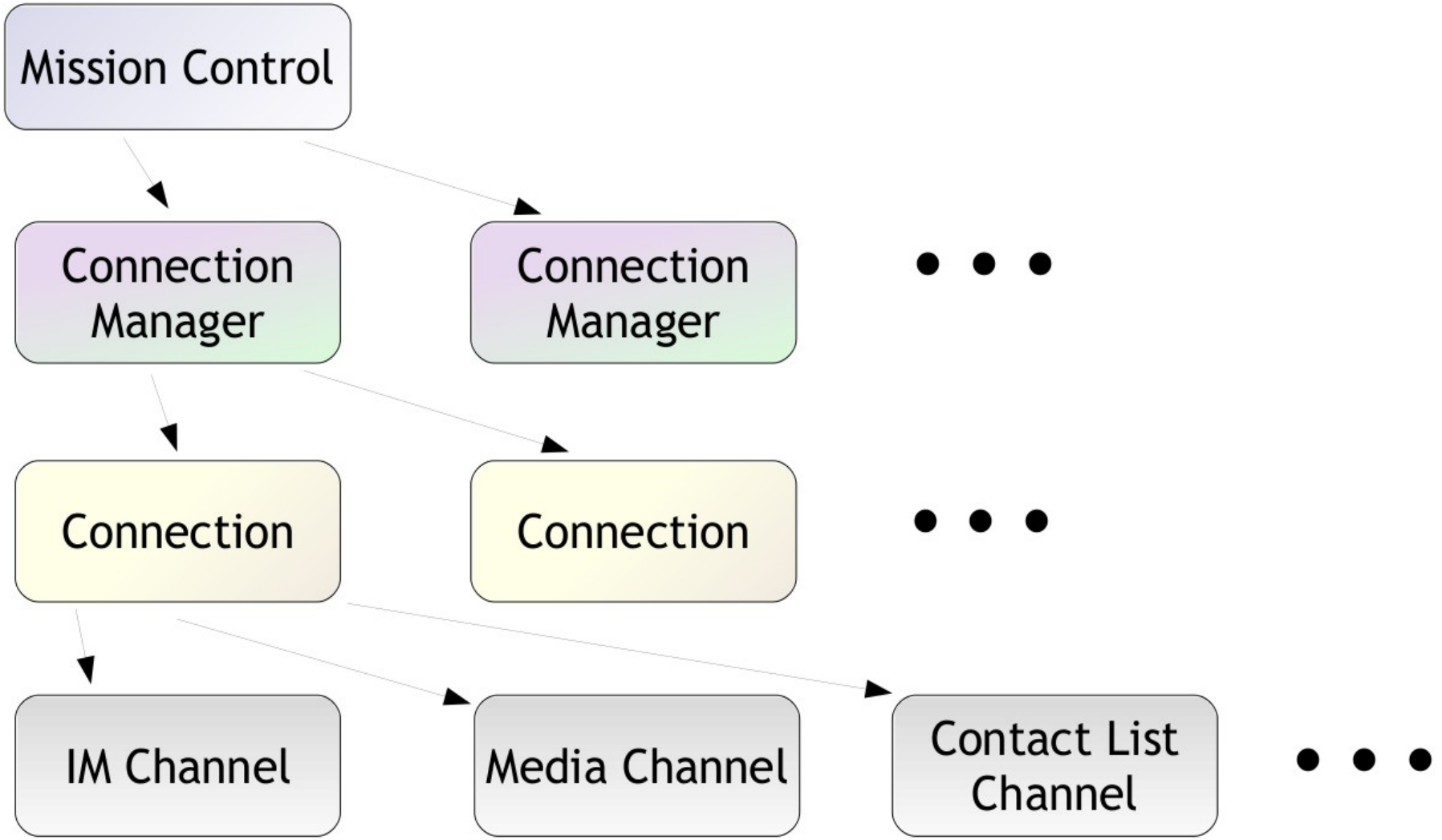
Epiphany link share



Soylent


Soylent people browser
File Edit Help
Video Phone Chat Email I am Available
Sven Pfaller
telepathy
[...]
Travis Reitter [*]
Vincent Untz
Xavier Claessens

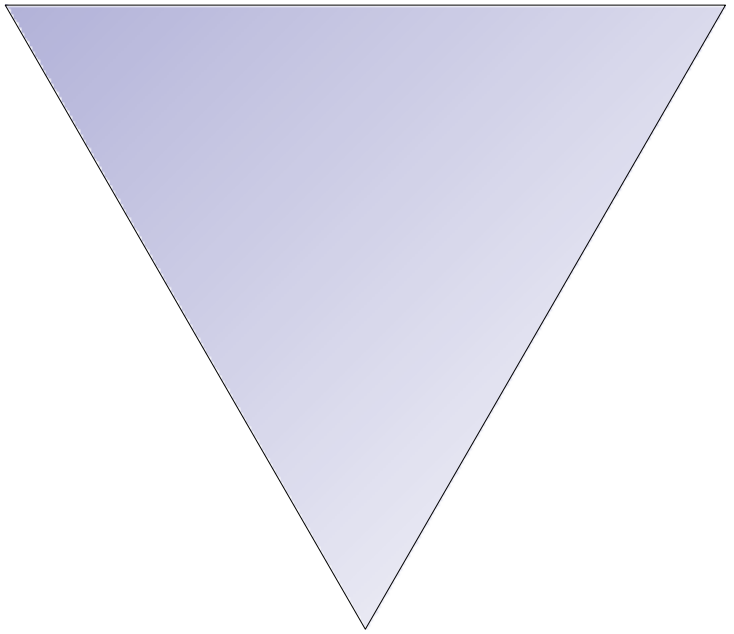
Telepathy objects



Ecosystem

libempathy-gtk
libempathy

libmissioncontrol



telepathy-glib

telepathy-glib

- Autogenerated from Telepathy spec
- Client and server-side bindings
- Sync/async operation support
- Extensive API docs in gtk-doc

libmissioncontrol

MissionControl

McAccount

libempathy

EmpathyIdle

EmpathyTpChat

EmpathyContact

EmpathyContactManager

EmpathyContactList

EmpathyTubeHandler

EmpathyTpTube

libempathy-gtk

EmpathyChat

EmpathyContactListStore

EmpathyContactListView

EmpathyPresenceChooser

pyempathy

getting and setting presence and status message

```
import empathy

# Get the status message
idle = empathy.Idle()

state = empathy.presence_to_str(idle.get_state())
msg = idle.get_status()

# Set new status message
idle.set_status('Custom status message')

# Go offline
idle.set_state(empathy.presence_from_str('offline'))
```

pyempathy (II)

getting contact list

```
import empathy

cm = empathy.ContactManager()

def on_contact(manager, contact):
  print “Alias: “, contact.get_name()
  print “Presence. “, contact.get_status()

cm.connect('member-added', on_contact)
```

pyempathy-gtk

selecting a contact from the list and sending a message

```
import gtk, empathy, pyempathy

manager = empathy.ContactManager()
store = empathygtk.ContactListStore(manager)
view = empathygtk.ContactListView(store, 'none')

def send(model, path, iter):
    (contact) = model.get(iter, 6)
    chat = empathygtk.tp_chat_new_with_contact(contact)
    chat.send(empathy.Message('Hi there!'))

d = gtk.Dialog()
d.get_child().add(view)
d.add_button(gtk.STOCK_CANCEL, 0)
d.add_button(gtk.STOCK_OK, 1)
d.show_all()

if gtk.run():
    view.get_selection().selected_foreach(send)
```

telepathy-python

going online and sending a message

```
import telepathy.client
from telepathy.client import interfaces, constants

reg = telepathy.client.ManagerRegistry()
reg.LoadManagers()

cm = reg.GetManager("gabble")
name, path = mgr[CONN_MGR_INTERFACE].RequestConnection("jabber",
    "senko.rasic@jabber.org")
conn = telepathy.client.Connection(name, path)
connection[CONN_INTERFACE].Connect()

handle = conn[CONN_INTERFACE].RequestHandles(CONNECTION_HANDLE_TYPE_CONTACT,
    ["someone@jabber.org"])[0]

path = conn[CONN_INTERFACE].RequestChannel(
    CHANNEL_TYPE_TEXT, CONNECTION_HANDLE_TYPE_CONTACT, handle, True)

chan = telepathy.client.Channel(conn.service_name, path)
chan.Send(CHANNEL_TEXT_TYPE_NORMAL, "Hi there!")
```

libempathy

going online / setting presence

```
static void
go_online ()
{
  EmpathyIdle *idle = empathy_idle_new ();
  empathy_idle_set_presence (idle, MC_PRESENCE_AVAILABLE, “I'm here”);
}
```

```
static void
set_message ()
{
  EmpathyIdle *idle = empathy_idle_new ();
  empathy_idle_set_status (idle, “Custom status message”);
}
```

libempathy (II)

opening a chat/call window

```
static void
chat_with (gchar *account_id, gchar *user_id)
{
  McAccount *acc = mc_account_lookup (account_id);
  empathy_dispatcher_chat_with_contact_id (acc, user_id);
}

static void
call (gchar *account_id, gchar *user_id)
{
  McAccount *acc = mc_account_lookup (account_id);
  empathy_dispatcher_call_with_contact_id (acc, user_id);
}
```

libempathy (III)

getting a contact

```
static EmpathyContact *
get_contact (gchar *account_id, gchar *contact_id)
{
  EmpathyContactFactory *cf;
  McAccount *acc;
  EmpathyContact *contact;
  EmpathyContactReady ready_flags;

  acc = mc_account_lookup (account_id);

  cf = empathy_contact_factory_new ();
  contact = empathy_contact_factory_get_from_id (cf, acc, contact_id);

  ready_flags = EMPATHY_CONTACT_READY_HANDLE | EMPATHY_CONTACT_READY_NAME;
  empathy_contact_run_until_ready (contact, ready_flags, NULL);

  return contact;
}
```

telepathy-glib

sync opening a text channel

```
static EmpathyTpChat *
open_text_channel (EmpathyContact *contact)
{
  TpChannel *chan;
  gchar *obj_path;
  GError *error = NULL;
  EmpathyTpChat *chat;
  MissionControl *mc = mission_control_new (tp_get_bus ());
  McAccount *acc = empathy_contact_get_account (contact);
  TpConnection *conn = mission_control_get_tpconnection (mc, acc, NULL);

  tp_connection_run_until_ready (conn, FALSE, NULL, NULL);

  if (!tp_cli_connection_run_request_channel (conn, -1,
      TP_IFACE_CHANNEL_TYPE_TEXT,
      TP_HANDLE_TYPE_CONTACT,
      empathy_contact_get_handle (contact),
      TRUE, &obj_path, &error, NULL))
        return NULL;
```

telepathy-glib

cont'd

```
  chan = tp_channel_new (conn, obj_path,
      TP_IFACE_CHANNEL_TYPE_TEXT,
      TP_HANDLE_TYPE_CONTACT,
      empathy_contact_get_handle (contact),
      NULL);

  tp_channel_run_until_ready (chan, NULL, NULL);

  tp_cli_channel_type_text_run_send (chan, -1,
    TP_CHANNEL_TEXT_MESSAGE_TYPE_NORMAL,
    "Hi there!", NULL, NULL);

  chat = empathy_tp_chat_new (chan);

  return chat;
}
```

libempathy-gtk

embedding a chat widget

```
static void
show_chat_window (EmpathyTpChat *chat)
{
  GtkWidget *win = gtk_window_new (GTK_WINDOW_TOPLEVEL);
  EmpathyChat *emchat = empathy_chat_new (chat);

  gtk_container_add (GTK_CONTAINER (win), GTK_WIDGET (emchat));
}
```

If you get stuck

or have a cool idea you want to talk about

- #telepathy on Freenode
- telepathy@lists.freedesktop.org

Future

- Mission Control D-Bus Interface standard
  - with indings in tp-glib
- libsoylent for accessing people
- A few GTK+ widgets (tp-gtk?)
- Better Python (, etc.) bindings

We have all the plumbing ready,

now let's build the kitchen.