

Easy Publish and Consume Library

Mathias Hasselmann

Openismus



Scope

- Publish information
- Announce that information
- Find that information
- And finally consume it



Techniques

- **DNS-SD** via **Avahi** for publication
- **HTTP(S)** via **libsoup** and **gnutls** for transport

Openismus



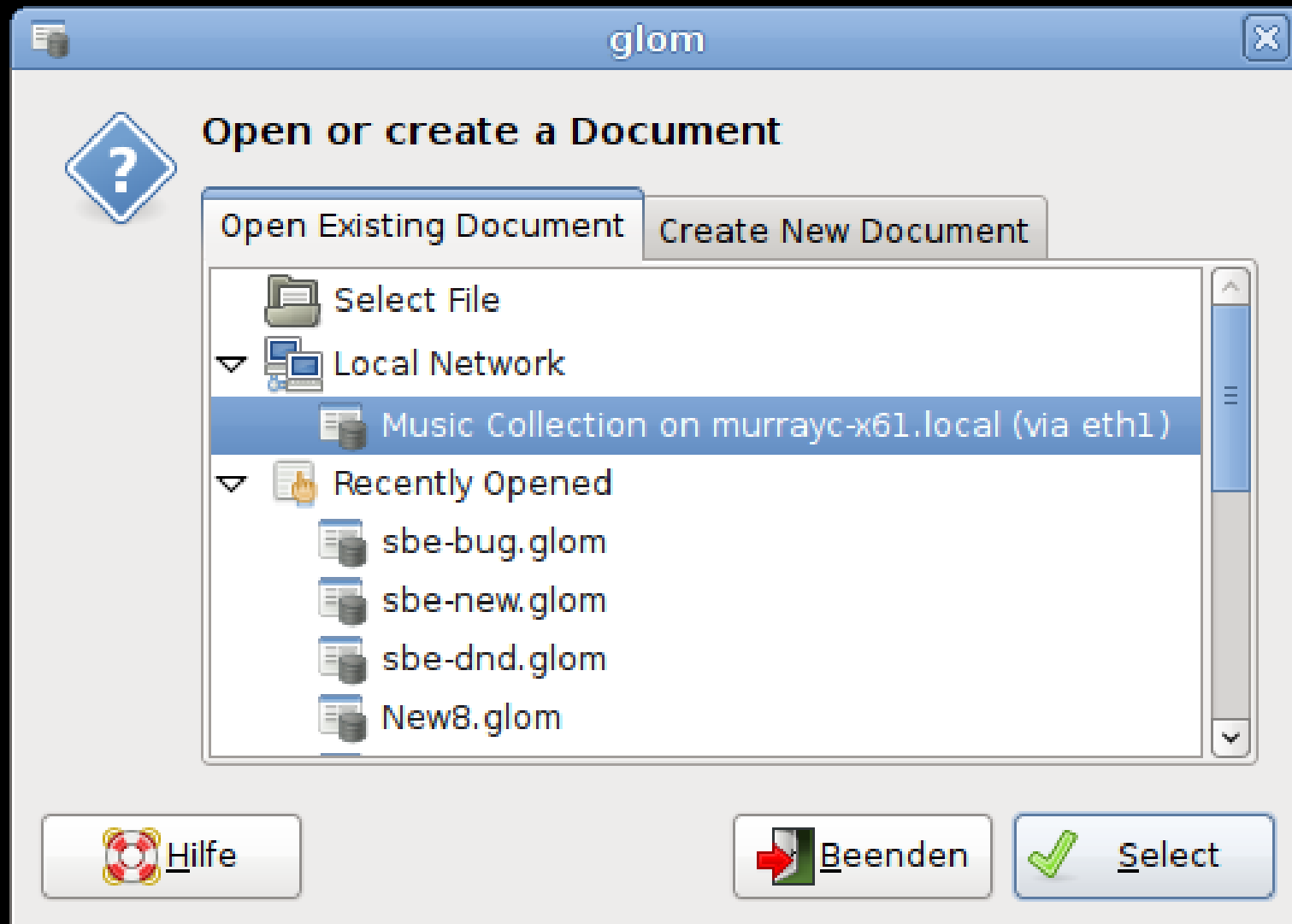
Motivation

- Make Glom network aware

Openismus



Database Chooser in Glom



Motivation

- Share things like Totem playlists

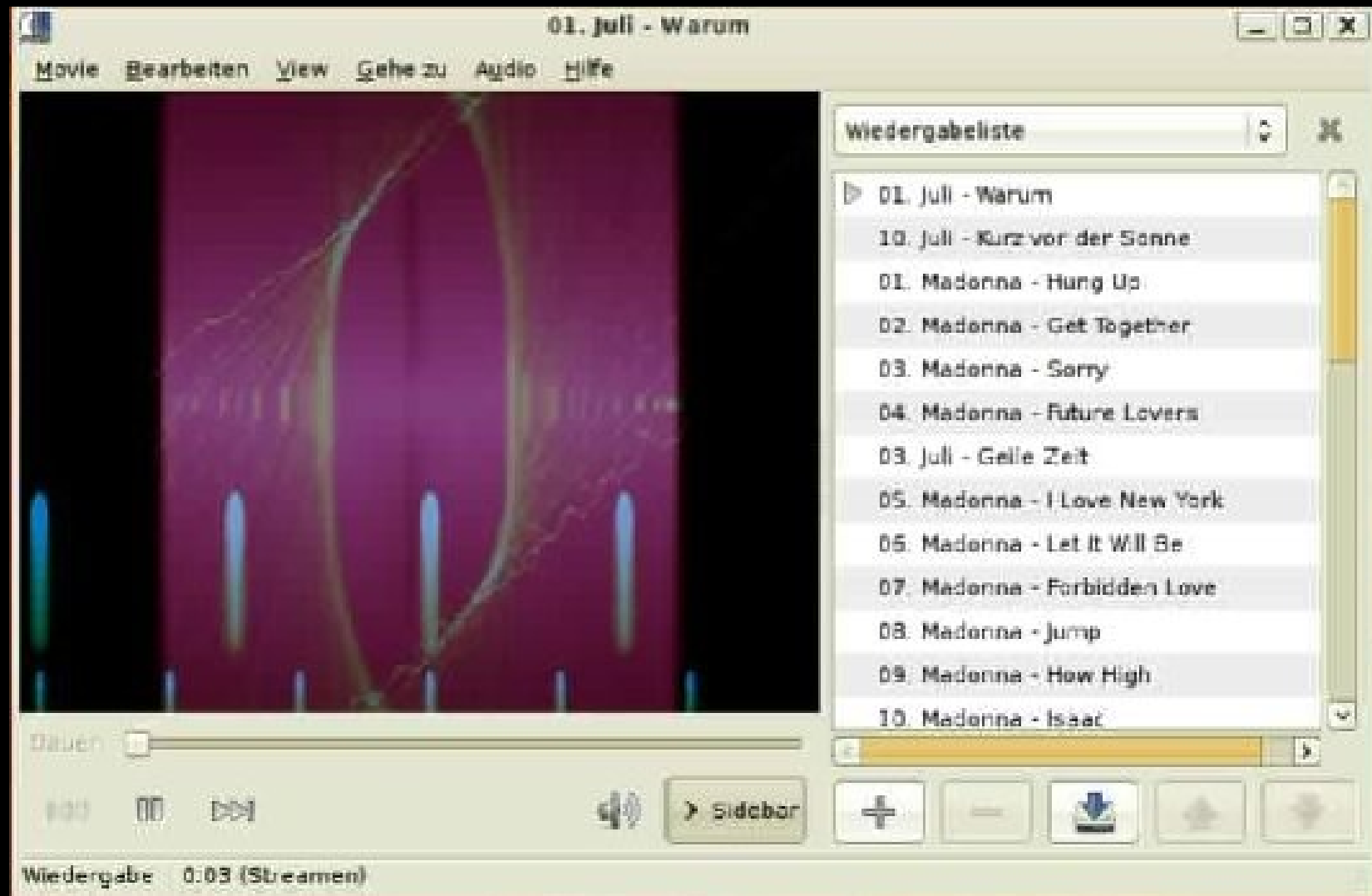
Openismus



Playlist Sharing in Totem



Playlist Sharing in Totem



Motivation

- We are Desktop hackers,
not networking experts.

Openismus



Motivation

- Want some **simple API**.

Openismus



Approach

- Let's create some simple **Network HashTable**.

Openismus



Consumer: Example

```
consumer =  
    epc_consumer_new_for_name ("kitchen");  
  
value = epc_consumer_lookup  
    (consumer, "cake", &length, &error);  
  
if (value)  
    g_print ("%s (%d)\n", value, length);  
else  
    handle_error (&error);
```



Publisher: Example

```
publisher = epc_publisher_new  
    ("Tasty Cakes", "kitchen", NULL);  
  
epc_publisher_add  
    (publisher, "cake", "lots of sugar", -1);  
  
epc_publisher_run (publisher);
```



Publisher: Example

```
publisher = epc_publisher_new  
    ("Tasty Cakes", "kitchen", NULL);
```

```
epc_publisher_add  
    (publisher, "cake-picture",  
     image_data, image_length);
```

```
epc_publisher_run (publisher);
```



Publisher: Example

```
publisher = epc_publisher_new  
    ("Tasty Cakes", "kitchen", NULL);
```

```
epc_publisher_add_file  
    (publisher, "cake-recipe",  
     "recipe.txt");
```

```
epc_publisher_run (publisher);
```



Publisher: Example

```
publisher = epc_publisher_new  
    ("Tasty Cakes", "kitchen", NULL);
```

```
epc_publisher_add_handler  
    (publisher, "cake-status",  
     cake_status_handler, self, NULL);
```

```
epc_publisher_run (publisher);
```



Publisher: Example

```
static EpcContents*
timestamp_handler (EpcPublisher *publisher,
                  const gchar *key,
                  gpointer      data)
{
    /* ... */

    return epc_contents_new
        ("text/plain", buffer, length, g_free);
}
```

A large, stylized, cursive logo with the word "Openismus" in a light gray color, positioned diagonally across the middle of the slide, partially overlapping the code.

Publisher: Authorization

- `epc_publisher_set_auth_handler(key, handler)`
- `epc_auth_context_get_key()`,
`epc_auth_context_check_password(password)`
- `epc_auth_context_get_password()`,
“auth_flags” property



Publisher: Collision Handling

- “collision-handling” property:
 - EPC_COLLISIONS_IGNORE
 - EPC_COLLISIONS_CHANGE_NAME
 - EPC_COLLISIONS_UNIQUE_SERVICE



Publisher: Certificates

- `epc_publisher_set_credentials (certfile, keyfile)`

Openismus



Publisher: Miscellaneous

- `epc_publisher_set_contents_path(path)`
- `epc_publisher_add_bookmark(key, label)`
- `epc_publisher_expand_name(name)`
- “protocol” property



Consumer: Lazy Publisher Lookup

- `bool resolve_publisher(timeout)`
- `bool is_publisher_resolved()`
- “publisher-resolved” signal

Openismus



Consumer: Authentication

- “username” and “password” property
- “authenticate” signal

Openismus



Consumer: Miscellaneous

- epc_consumer_list (pattern)
- “protocol” property

Openismus



Service Monitor

```
monitor = epc_service_monitor_new  
    ("kitchen", NULL, EPC_PROTOCOL_UNKNOWN);  
  
g_signal_connect  
    (monitor, "service-found",  
     G_CALLBACK (service_found_cb), NULL);  
  
g_signal_connect  
    (monitor, "service-removed",  
     G_CALLBACK (service_removed_cb), NULL);  
  
g_main_loop_run (loop);
```



Dispatcher

```
dispatcher = epc_dispatcher_new  
            ("Dead Tree Desecrator");
```

```
epc_dispatcher_add_service  
    (dispatcher, EPC_ADDRESS_IPV4,  
     "_ipp._tcp", NULL, NULL, 651,  
     "path=/printers", NULL);
```

```
epc_dispatcher_run (dispatcher);
```



Dispatcher

```
dispatcher = epc_dispatcher_new  
            ("Dead Tree Desecrator");
```

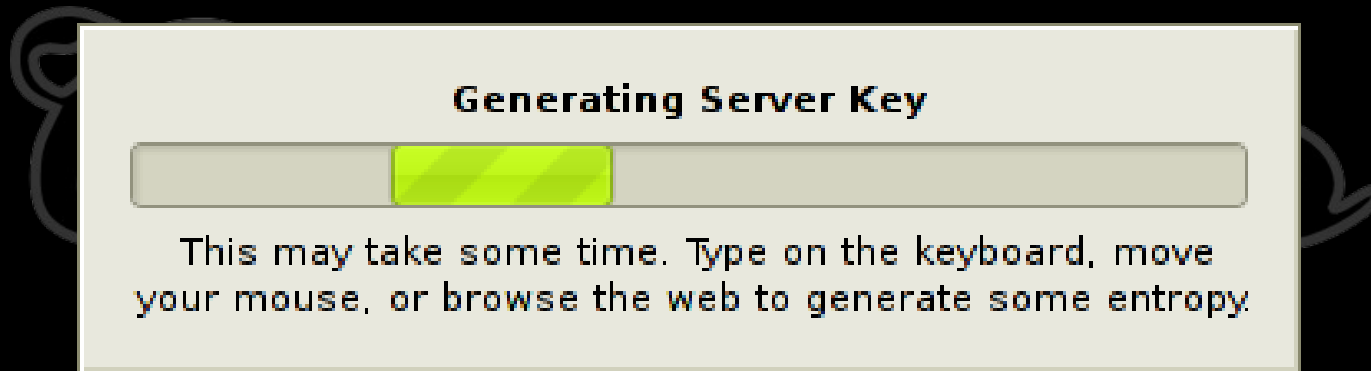
```
epc_dispatcher_add_service  
    (dispatcher, EPC_ADDRESS_UNSPEC,  
     "_duplex._sub._printer._tcp",  
     NULL, NULL, 515, NULL);
```

```
epc_dispatcher_run (dispatcher);
```

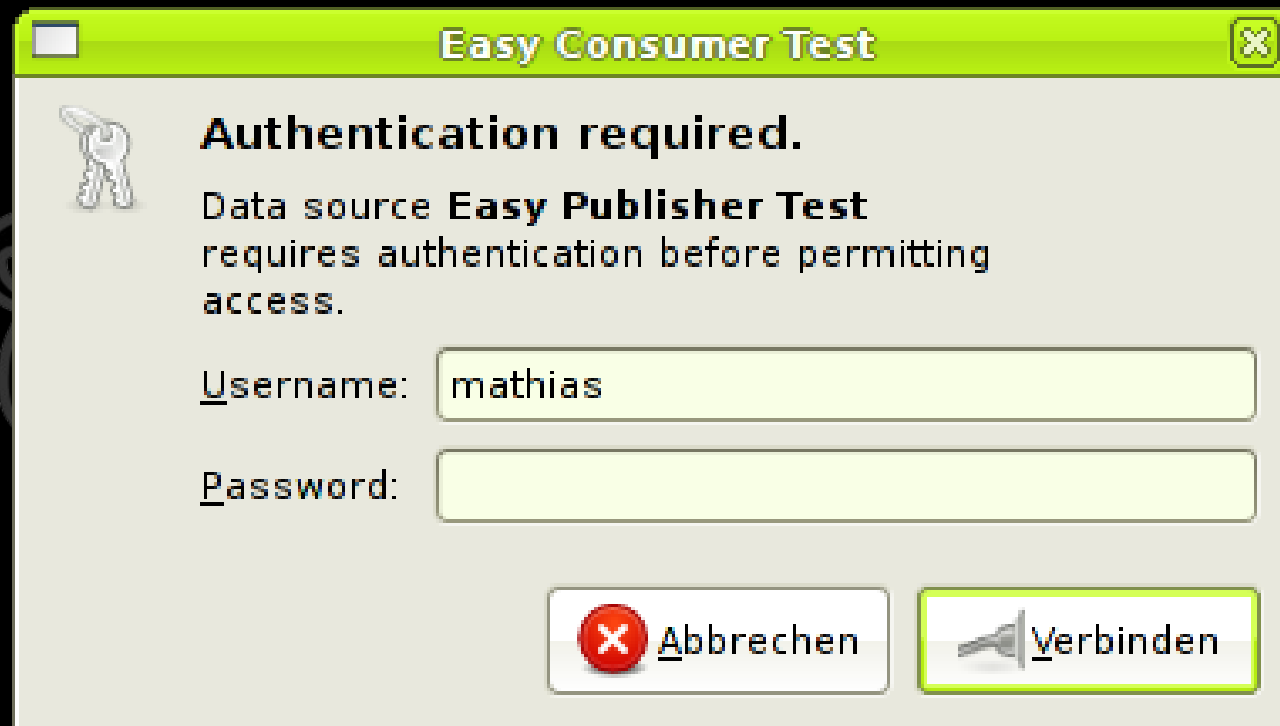


Widgets: Progress Window

- `epc_progress_window_install` (parent)



Widgets: Password Dialog



A screenshot of a Java Swing dialog box titled "Easy Consumer Test". The dialog has a yellow title bar with standard window controls. Inside, there is a key icon next to the text "Authentication required." followed by a message: "Data source **Easy Publisher Test** requires authentication before permitting access." Below this, there are two text input fields. The first is labeled "Username:" and contains the text "mathias". The second is labeled "Password:" and is empty. At the bottom, there are two buttons: "Abbrechen" (Cancel) with a red 'X' icon, and "Verbinden" (Connect) with a key icon. The "Verbinden" button is highlighted with a green border.

Easy Consumer Test

 **Authentication required.**
Data source **Easy Publisher Test** requires authentication before permitting access.

Username:

Password:

 Abbrechen  Verbinden



Future

- Improved certificate validation.
- Change Notifications?
- Allow the client to update the HashTable?

Openismus



Links

- More Information:
<http://live.gnome.org/libepc>
- Examples:
<http://svn.gnome.org/viewvc/libepc/trunk/examples/>
- Reference Manual:
<http://library.gnome.org/devel/libepc-1.0/unstable/>



Questions?

