Introduction to the Desktop Data Model

Owen Taylor

GUADEC 2008

10 July

Your information is not
(just) on your hard drive

Instant updates
make users happy

Instant updates
make users happy

Even if the information
is not on their hard drive

Web pages with
reload buttons?

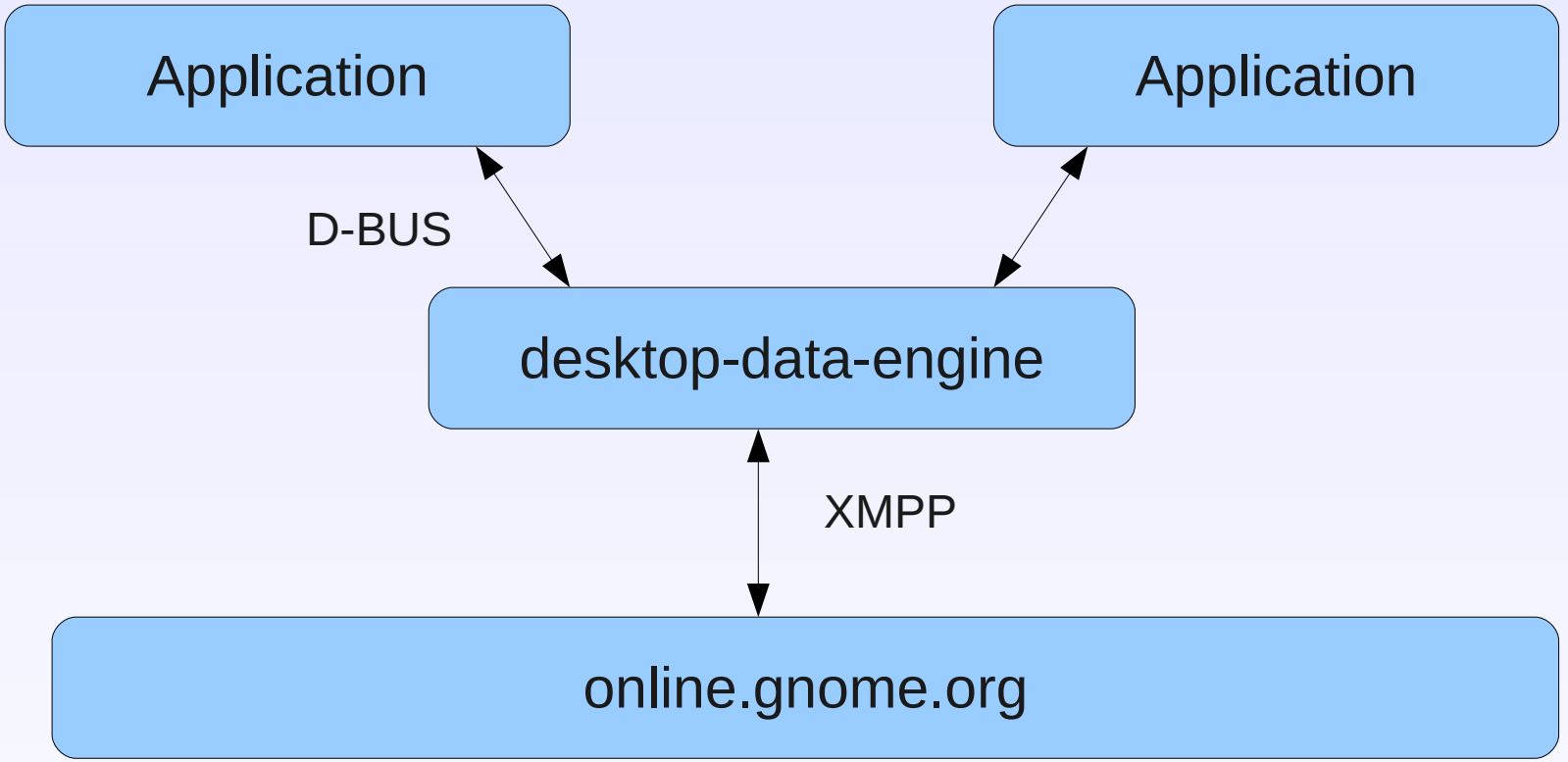
Always up-to-date
local applications

GNOME

Allow local applications to get:

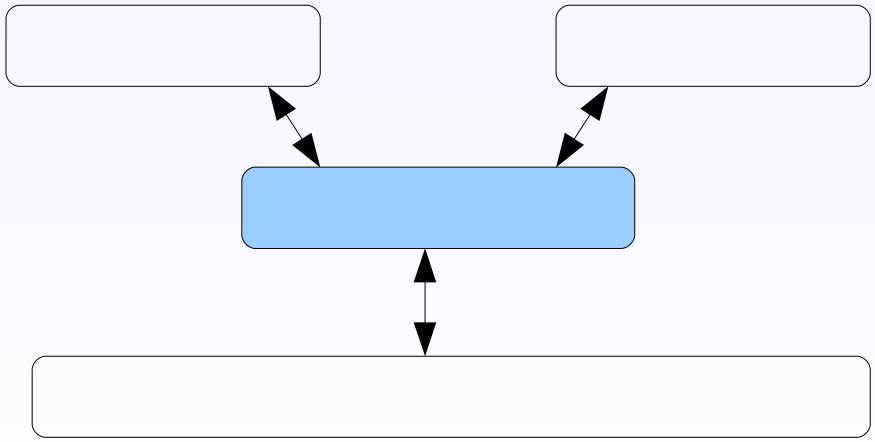
The information they need
&
Instant updates

The Big Picture



desktop-data-engine

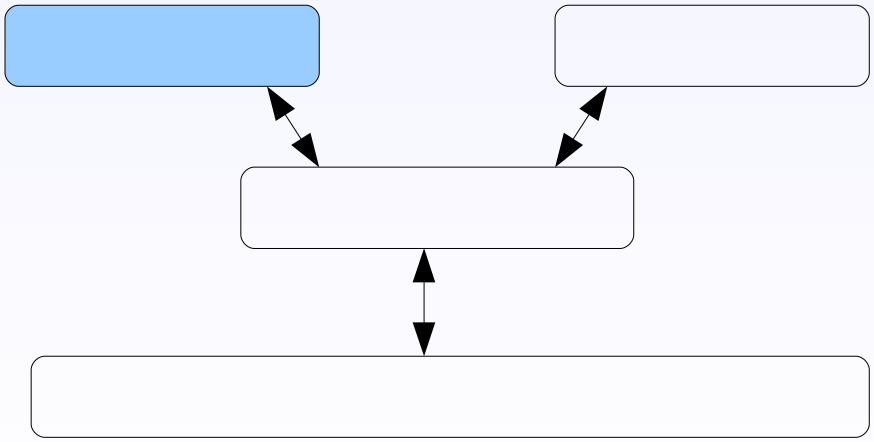
- D-BUS to apps
- XMPP to server
- Integrate local information (empathy IM contacts)
- Cache session data in memory
- Persistant caching via SQLite (offline mode)



GNOME

Application interface

- D-BUS not convenient to use directly
- Need client library
- Python and C (some work on C#)



GNOME

To the details...

Basic Ideas

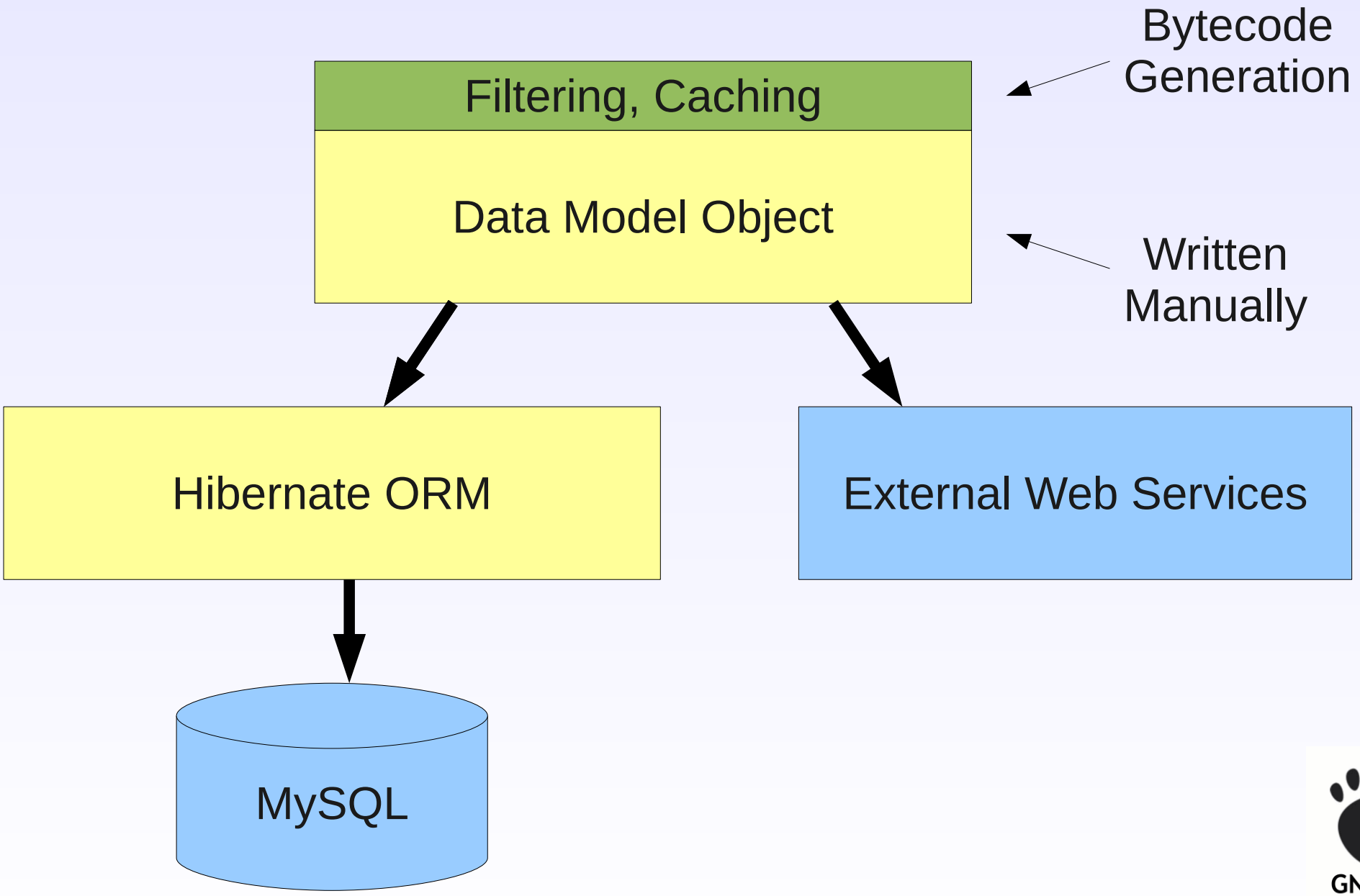
- Resources:
  - “Something”
  - Identified by URI
- Properties:
  - Numbers
  - Strings
  - *Resources*

Example

```
http://online.gnome.org/o/user/61m76k3hGbRRFS
   name: Owen
   photoUrl: /files/headshots/61m76k3hGbRRFS
   emails:
    [otaylor@redhat.com,otaylor@fishsoup.net]
   currentTrack:
    http://online.gnome.org/o/track/531234

http://online.gnome.org/o/track/531234
   artist: John Holloway
   name: Sonata No. 5 in C major: I. Largo
```

Resources on Server



Query / Update

- Query: Get data
  - Like HTTP GET, SQL SELECT
- Update: Change data
  - Like HTTP POST, SQL UPDATE

Query

- Identified by URI

```
http://online.gnome.org/p/apps#getPopularApplications
```

- Parameters: string key value pairs

```
category=games
```

- Fetch: what properties to get

```
id;iconUrl;name;description
```

- Returns:
  - List of resources
  - Properties for each resource (from fetch)

Query (Python)

```
def get_popular_apps(self):
  model = ddm.DataModel()
  query = model.query(
     "http://online.g.o/p/apps#getPopularApplications",
     category="games",
     fetch="id;iconUrl;name;description")
  query.addHandler(self.on_got_popular_apps)
  query.execute()


def on_got_popular_apps(self, resources):
  print "Got applications:"
  for application in resources:
    print "    ", application.name
```

getResource

`http://online.gnome.org/p/system#getResource`

- Standard query
- Already have a resource – want to get more information
- Special handlings in bindings:

```
query = model.query_resource(
            model.self_resource,
            fetch="currentTrack[artist;name]")
```

Self Resource

```
query = model.query_resource(
            model.self_resource,
            fetch="currentTrack[artist;name]")
```

- Automatically fetched by system
- Basic starting place
  - Accounts, Contacts, Applications, etc

Recursive Fetches

```
query = model.query_resource(
            model.self_resource,
            fetch="currentTrack[artist;name]")
```

- For resource valued properties,
- Descend and get properties of referenced resource
- Can get complex

```
aimBuddies [+;icon;statusMessage;contact +;user
  [+;contact]]; xmppBuddies [+;icon;statusMessage;contact
  +;user [+;contact]]; mugshotLocalBuddies [+;icon;user
  [+;contact]]
```

Default fetch

```
name;photoUrl;track[name;artist]
```

```
+;track[+]
```

GNOME

Updates

- Identified by URI

`http://mugshot.org/p/contacts#createContact`

- Parameters: string key value pairs

`addressType=email`

`address=otaylor@redhat.com`

- Fetch string if update returns results

`(createContact` Returns newly created contact object)

Notification

- Server remembers what it sent client
- If that changes, a notification is automatically sent
- *No Need Select For Notification*

Notification (Python)

```
[...]

def on_got_popular_apps(self, resources):
    print "Got applications:"
    for application in resources:
        print "    ", application.name
        application.connect(self.name_changed, "name")

def on_name_changed(self, application):
    print "App name is now", application.name
```

The Trick

- Server remembers what it sent the client
- Client keeps copy of the data
- Server notifies on changes

The Trick

- Server remembers what it sent the client
- Client keeps copy of the data
- Server notifies on changes

- No need to send duplicate information

Compression Example (1/2)

- Get my contacts

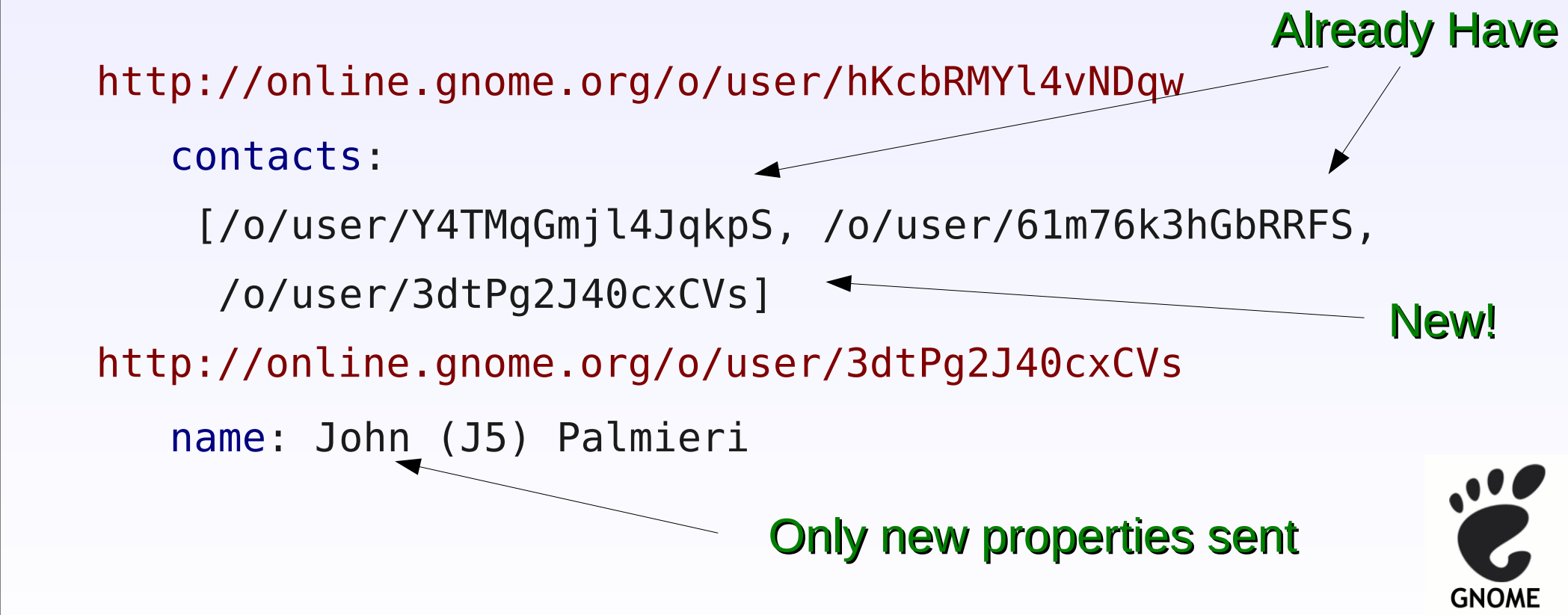
```
getResource(http://online.gnome.org/o/user/61m76k3hGbRRFS)
  fetch=userContacts[name]

http://online.gnome.org/o/user/61m76k3hGbRRFS
   contacts:
    [/o/user/hKcbRMYl4vNDqw, /o/user/Y4TMqGmjl4JqkpS]
http://online.gnome.org/o/user/hKcbRMYl4vNDqw
   name: Colin
http://online.gnome.org/o/user/Y4TMqGmjl4JqkpS
   name: Marina
```

Compression Example (2/2)

- Then get Colin's contacts

```
getResource(http://online.gnome.org/o/user/hKcbRMYl4vNDqw)
  fetch=userContacts[name]
```



GNOME

Comparison to LDAP

Data Model	LDAP
• Notification central	• Notification Add-on
• Merge in external information	• Single database
• Read-only with “Update Queries”	• Arbitrary Modification
• No search (but app defined queries)	• Searching

Comparison to RDF

- Resource Data Framework
- Triples
  - Subject (resource)
  - Predicate (property)
  - Object (property value)
- Can easily map data model data onto RDF

Comparison to RDF

```
@prefix r1: <http://online.gnome.org/p/o/user>.

<http://online.gnome.org/o/user/61m76k3hGbRRFS>
   r1:name "Owen"
   r1:emails otaylor@redhat.com
   r1:emails otaylor@fishsoup.net
   ...
```

Comparison to RDF

Data Model

- Defines protocol
- Canonical server for each piece of information

RDF

- No protocol
- Known information, not necessarily from a particular source

Extending: current

Hack on the online.gnome.org server

http://developer.mugshot.org/wiki/Server_Development_Setup

Extending: future

Better support for LDAP-style stuffing data into the server?

Server fetches data from from your via HTTP?

Conclusion

- Protocol works well
  - Complicated set of data
  - Instant updates
- Extensibility not really worked out yet

More information

These slides:

http://fishoup.net/istanbul-2008/data-model-intro.odp

Design document:

http://fishsoup.net/articles/mugshot-data-model

General information:

http://live.gnome.org/OnlineDesktop

Mailing list:

http://mail.gnome.org/mailman/listinfo/online-desktop-list

IRC:

irc.gnome.org:#online-desktop