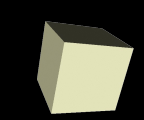
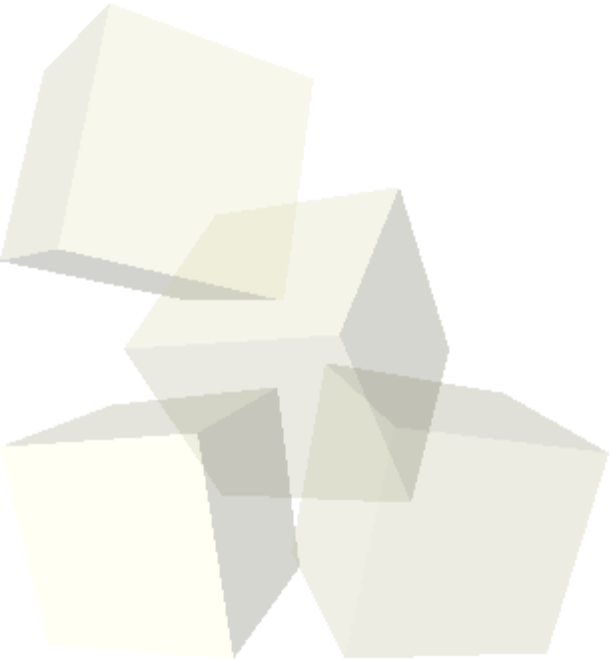


D-Bus

Teamwork für Software

Kevin Krammer

Grazer LinuxTage 2007



Teamwork

- Gemeinsames Ziel
- Zusammenarbeit
- Arbeitsteilung
- Kommunikation

Kommunikation – Chat

- Verbinden zu einem Chatserver
- Eindeutige Zuordnung (Socket)
- Persönliche Zuordung (Nickname)
- Nachrichten an alle
- Nachrichten an Einzelne

Kommunikation - D-Bus

- Verbinden zu einem D-Bus Server
- Eindeutige Zuordnung (unique name)
- Persönliche Zuordnung (requested name)
- Nachrichten an alle (Signals)
- Nachrichten an Einzelne (Methodcalls)

D-Bus - Begriffe

- “Der Bus”: der Nachrichtenkanal, d.h. der D-Bus Daemon
  - System Bus
  - Session Bus

- Service Name: Adressierung eines Service
  - unique name, z.B. “:1.5”
  - requested name, z.B. “org.freedesktop.DBus”

D-Bus – Begriffe 2

- Object Path: Adressierung innerhalb eines Service
  - wie ein Dateipfad, z.B. “/org/freedesktop/Hal/Manager”
- Interface: Gruppierung zusammengehöriger Methoden/Signals
  - ähnlich wie Service Name, z.B. “org.freedesktop.DBus.Introspectable”

D-Bus – Begriffe 3

- Message: Serialisierte Form eines Aufrufs, einer Antwort oder eines Signals. Der Bus verteilt nur Nachrichten zwischen Teilnehmern
- Activation: automatisches Starten eines Service durch den Bus, wenn ein an den Service gerichteter Aufruf erfolgt

Intropection

- Einfaches XML Format um Interfaces und seine Methoden/Signals zu beschreiben
- Services sollten für alle ihre Objekte das “Introspectable” Interface implementieren, um ihre Funktionalität entdeckbar zu machen

Introspection - Beispiel

```
<node>
   <interface name="org.freedesktop.DBus">
      <method name="ListNames">
         <arg direction="out" type="as" />
      </method>

      <method name="GetNameOwner">
         <arg direction="in"  type="s" />
         <arg direction="out" type="s" />
      </method>

      <signal name="NameOwnerChanged">
         <arg type="s" />
         <arg type="s" />
         <arg type="s" />
      </signal>
   </interface>
</node>
```

Programmierung

- libdbus: Low-Level Bibliothek
  - Connection Handling
  - Serialisierung/Deserialiserung
- Eigentliche Benutzung durch Bindings
  - Mapping von Datentypen
  - Eventloop Integration
  - Vereinfachung
  - Codegenerierung
  - C++, GLib, Qt, Java, Python, Ruby, Mono/C#, Pascal

Anwendungsfälle

- Rechtetrennung: Services mit erweiterten Rechten am Systembus, Clients mit normalen Benutzerrechten
  - HAL
  - NetworkManager
- Gemeinsame Resourcen: Zugriff nur durch den Service, zentrales Caching, kein Locking erforderlich, etc
  - Passwörter, Cookies, Einstellungen, PIM Daten, usw.

Anwendungsfälle 2

- Trennung von Verarbeitung und Visualisierung
  - Benachrichtigungen (Notifications)
  - Fortschrittsanzeige, usw.
- Desktop Integration
  - Bevorzugte Anwendungen
  - Dialoge
  - Benutzereinstellungen
  - usw.

TEAMWORK
FÜR
SOFTWARE

auch als Service Orientierte Architektur (SOA) bekannt

Vielen Dank für Ihre Aufmerksamkeit!

Gibt es noch Fragen?