the ezHub API

Mike Flynn
Idiap Research Institute

IM2 Summer Workshop 2009
overview

• hub recap
  – traditional architecture
  – hub architecture

• ezHub
  – data model
  – producers & consumers
  – using HubObjects
hub basics

• data distribution & storage
  – live meeting support
    • requires real-time distribution of data
    • historic data still required

• data: metadata & annotations
  – who, where, when…
    • people join late, change seats, leave early…
  – video/audio stream addresses
  – speech data
  – slides, focus of attention, gestures…
traditional architecture

problem: consumers must poll for updates, latency, database schema complicated
hub architecture

- Microphone connected to ASR
- ASR connected to hub
- Hub connected to browser(s)
- Hub connected to database

Arrows indicate flow of data:
- Red arrow labeled "live data" from microphone to browser(s)
- Red arrow labeled "past data" from database to browser(s)
levels of abstraction

• TCP/IP
• XML
• triples
• ezHub object model
ezHub data model
producers & consumers

• distributed object model…
  – producers set attributes & relationships
  – consumers get attributes & relationships

• …with temporal access
  – arbitrary past time
  – latest
  – monitor future changes

• simple API but very efficient
  – object data is cached locally
  – temporally indexed
  – sequential or random access
using HubObjects

HubObject.open();
HubObject meeting = HubObject.find("IS1008c");
HubObject location = meeting.getObject("Location");
Set<HubObject> attendees = meeting.getInverseObjects("Attended");
for(HubObject person : attendees) {
    String name = person.getValue("name");
    ...
    Collection<String> speech = person.getValues("says");
    ...
    person.getValues("says", myHandler, "myquery");
}
...
myHandler.update(TimedTriple tt, String value, String ref) { ... }

myHandler.upToDate(String ref) { ... }
...
HubObject.close();
ezHub summary

• distributed object model
• data distributed in real time
  – also held in database
  – producers set attributes/relationships
  – consumers get attributes/relationships
• past & present data in same API
  – ask for attributes/relationships
  – now or in the past
  – continuously in the future
• abstract data model
  – data model independent of the Hub
  – extensible
ezHub tutorial & downloads

http://www.idiap.ch/~flynn/Hub