Boggle Server: two views

- What are advantages of having a boggle server?
  - Multiple players/clients
  - Uniformity of client-side programs (??)

- What does a boggle server serve? (or an ooga server)
  - What are paths of communication between client/server?
    - > Board, ??
    - < ???

- What are scenarios/use-cases for server
  - What happens when client connects?
  - What happens as game progresses?
Sockets: lower-level view via Oracle

- Oracle here is the Oracle of Bacon, hosted at UVA
  - Uses IMDB to develop database
  - Constructs graph from database, uses graph for queries
  - Server responds to several kinds of queries, all done via a lower-level socket-based protocol

- What are sockets in this and other network contexts?
  - Two-way pipe between programs, slightly different for client and server
    - Server accepts client connections, establishes the socket
    - Socket provides input/output streams
  - Bytes go in both directions, protocol needed to exchange information
All actors N-steps from named actor

- What is server protocol, what code do you need to see?
  - Any obvious refactorings that would help and hurt?

```c
int fd = sock_connect(SERVER, PORT);
if (fd == -1) return -1;
wf = fdopen(fd, "w");
r = fdopen(fd, "r");

putc(CMD_BY_NUMBER, w);
putstring(argv[1], w);
putint(atoi(argv[2]), w);
putc(USE_MOVIES, w);
fflush(w);
while ((res = fgetc(r)) == RSP_LINK_ENT) {
    char *s = getstring(r);
    puts(s);
    free(s);
}
```
Oracle.java: sockets in Java

```java
mySocket = new Socket(host, port);
myInput =
    new DataInputStream(mySocket.getInputStream());
myOutput =
    new DataOutputStream(mySocket.getOutputStream());
try{
    myOutput.writeByte(BYNUMBER); // protocol driven
    write(actor); // write a string
    writeInt(1); // distance of 1
    myOutput.writeByte(USE_MOVIES); // protocol driven
    myOutput.flush();
}
catch (IOException e) {...}
```

- What should we see in this client program to understand it?
multi-platform, multi-os client/server

- Architectural issues impact client/server code
  - Little-endian/Big-endian issues
    - 0xabcd is a 32-bit value, which is MSB? How is this stored?
  - How big is an int? 32-bits, 64 bits, ...
  - Other issues

- Towards raising the level of discussion
  - Worrying about integer byte order is not fun and it can be frustrating
  - Let’s worry about sending objects back-and-forth, not bytes
  - How does an object on one machine communicate with another object on a different machine
Writing objects: towards COM/Corba/...

- The Java stream hierarchy is a rich source of options
  - Object streams, Data streams, Buffered Readers, ...
  - Often these convert between bytes and characters
    - What’s the story with Unicode? (e.g. compared to ASCII)
    - FileStream, BufferedReader, ...

- We can read and write objects over sockets
  - Advantages compared to lower-level protocols?
  - Disadvantages?

- Issues in understanding and implementing
  - Where do objects “live”, are classes different?
  - Subclass/Superclass issues
  - What about connection issues (where, how, knowledge)
OOGA Boggle server

- How do BoardServer.java and BoardClient.java differ from a working OOGA/Boggle server?
  - What are main issues with subclass/superclass?
  - What about .class files

- What about other games, is there a reason for a server?
  - What about bejeweled at MSN, what does MSN want out of this?
  - Other issues?

- What about scoring, multiple games, heavy hits (24K/day?)
  - Difference in scale? Worth worrying about now?