

- 2–4 people per group. E-mail me if you need a partner for the project.
- Develop a news server (two versions) and a text-based client.
- Write a report, hand in the report and your programs no later than Monday, April 18th.

A News Server and News Clients

The server keeps a database of newsgroups, containing articles. The clients connect to the server. Sample conversation:

```
news> list
1. comp.lang.java
2. camp.lang.c++
news> list comp.lang.c++
1. What is C++? From: xxx
2. Why C++?      From: yyy
news> read 2
Why C++?      From: xxx
... text ...
news>
```

A client can also create and delete newsgroups, and create and delete articles in newsgroups.

The Project: Write Server and Client

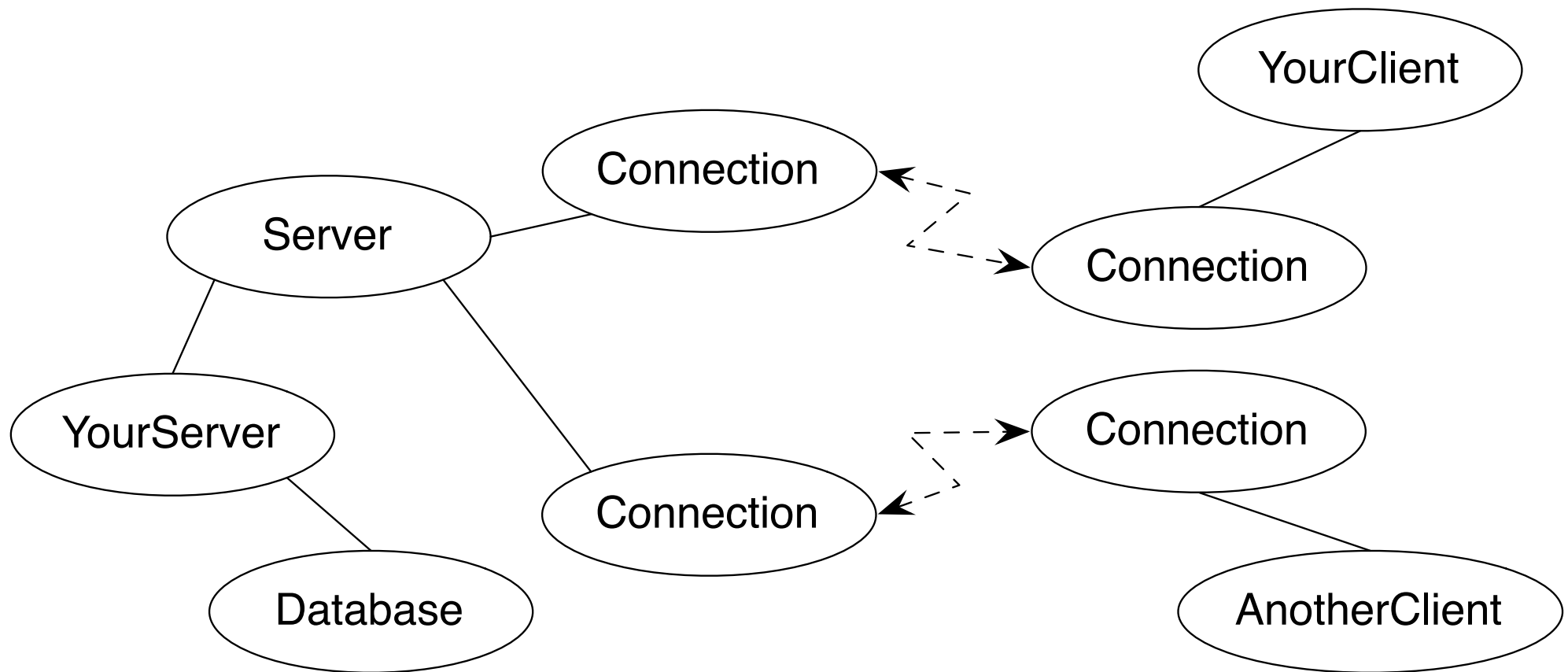
- You are to develop two versions of the server:
 - one in-memory server that forgets the data about newsgroups and articles between invocations (use the standard library containers for this database), and
 - one disk-based server that remembers the data between invocations (use files for this database)

These versions should be pluggable — the rest of the system shouldn't change when you change the database implementation.

- A single-threaded server is ok.
- You are to develop a client with a text-based interface. It shall read commands from the keyboard and present the replies from the server as text.

System Overview

The classes `Server` and `Connection` are pre-written.



Communication Protocol

A message is a sequence of bytes. Messages must follow a specified protocol, which specifies the message format.

List newsgroups (message to server and reply from server):

```
COM_LIST_NG COM_END
```

```
ANS_LIST_NG 2 13 comp.lang.java 15 comp.lang.c++ ANS_END
```

2 is the number of newsgroups, 13 and 15 are the unique identification numbers of the newsgroups `comp.lang.java` and `comp.lang.c++`.

Numbers and strings are coded according to the protocol. Hint: write a class to handle the communication on “low protocol level” (encoding and decoding of numbers and strings).

Class Connection

```
struct ConnectionClosedException {};  
  
/* A Connection object represents a socket */  
class Connection {  
public:  
    Connection(const char* host, int port);  
  
    Connection();  
  
    virtual ~Connection();  
  
    bool isConnected() const;  
  
    void write(unsigned char ch) const;  
  
    unsigned char read() const;  
};
```

Class Server

```
/* A server listens to a port and handles multiple connections */  
class Server {  
public:  
    explicit Server(int port);  
  
    virtual ~Server();  
  
    bool isReady() const;  
  
    std::shared_ptr<Connection> waitForActivity() const;  
  
    void registerConnection(const shared_ptr<Connection>& conn);  
  
    void deregisterConnection(const shared_ptr<Connection>& conn);  
};
```

Server Usage

```
while (true) {
    auto conn = server.waitForActivity();
    if (conn != nullptr) {
        try {
            /*
             * Communicate with a client, conn->read()
             * and conn->write(c)
             */
        } catch (ConnectionClosedException&) {
            server.deregisterConnection(conn);
            cout << "Client closed connection" << endl;
        }
    } else {
        conn = make_shared<Connection>();
        server.registerConnection(conn);
        cout << "New client connects" << endl;
    }
}
```


- Write the report, preferably in English, follow the instructions.
- Create a directory with your programs (only source) and a Makefile.
- Write a README file (text) with instructions on how to build and test your system.
- Submission:
 - 1 The report in PDF format.
 - 2 The README file.
 - 3 The program directory (tar-ed and gzip-ped). Don't bury the readme file or the report inside the gzip file.
 - 4 E-mail these files to `eda031@cs.lth.se`. Subject line:

`news by id1 id2 id3`