CS 4500 Assignment D

Labyrinth TCP Client / Reflection on Implementing Specifications

Due: Tuesday, October 8, 11:59pm

Submission: You must submit the following artifacts in a directory called `D` in your repository's master branch:

Task 1 An executable named `D`.

Task 2 A file `traversal-integration-report.md`

Optional A PDF `new-project-programming-language.pdf`.

As before, all auxiliary files are to be placed in a subdirectory of `D`, named `Other`.

Task 1

Implement *a client program* for the TCP Labyrinth server according to the protocol specification, which will be made available on Wednesday, 10/2 before noon as an update to this assignment.

After starting up and connecting to the server via TCP, the client enters an interactive loop, reading JSON requests for the labyrinth from STDIN, processing them and passing the corresponding TCP requests to the server, and rendering responses to STDOUT. The client accepts well-formed JSON labyrinth requests following Task 3 in Assignment C. An interactive session is ended when the user sends a `^D` (Ctrl-D) to STDIN.

The client, `D`, should take the following arguments, in this order:

1. the IP address of the server,[1]
2. the destination port, and
3. the user's name.

[1] It does not have to accept a hostname. However, I will be pleased if it does.

If the name is missing, use `John Doe` as default. If the port is missing, use 8000 as the default. If the IP adderess is also missing, use 127.0.0.1 (localhost). That is, if `D` is called without arguments, it should connect to 127.0.0.1 on port 8000 and use `John Doe` as the user's name. If only one argument is given, it is the IP address. If two, they are the address and port.

For this task, you only need to implement a client that follows the given protocol. To test your client, you can implement a *mock server*. The Wikipedia page on Mock Objects is a good starting point on this concept.

Task 2

In a directory `D` of your master branch, you will receive an implementation of your specification for a `Labyrinth` server module. Alternatively, there might be a memo explaining why the specification could not be implemented.

If you received an implementation, write a short (1-2 pages) memo addressing the following questions:

1. How well did the other team implement your specification? Did they follow it truthfully? If they deviated from it, was it well justified?
2. Were you or would you be able to integrate the received implementation with your client module from Task 3 of Assignment C? What was the actual or what is estimated effort required?
    - The implementation might not be in the language you requested. In that case, you think about whether you would be able to integrate the module through a foreign function interface or a similar mechanism. Note, *your* language does not actually have to support FFI – just assume you have a mechanism for calling foreign functions and interpreting foreign values.
3. Based on the artifact you received and the above two questions, how could you improve your specification to make it more amenable for implementation as you intended?

If you received an explanation of why the specification could not be implemented, or why it is incomplete, instead of answering 1 and 2 above, write a reply to the explanation and include an answer to 3.

Optional: Switching Languages for the Semester Project

Complete this task if and only if you wish to switch your chosen programming language.

As this is the last warm-up assignment, you now have the option to choose a different language to use for the remainder of the semester.

1. Confirm (for yourself) that you can comfortably use your language to

(a) process command-line arguments;
(b) work with STDIN and STDOUT;
(c) work with TCP sockets: create and listen for connections on a particular port, as well as connect to a specified IP address and port
(d) write, manage and run unit tests
(e) read, parse, and write JSON

The above points form a minimum, but not an exhaustive list of concepts you will acquire on the side for software system construction. Over the course of the semester, your chosen language will have to support other essential concepts from software systems building.

2. Write a memo containing two paragraphs: one that explains why you are abandoning your originally chosen language, and another explaining how you have checked that you are familiar with the above concepts in your new language.

Protocol for Task 1

Protocol for a Labyrinth Server-Client Interaction

The first three sections explain the interaction arrangements between the user, the client program (`D`), and the `server` component. The remaining sections specify the format of the messages.

Start Up Steps

```
---------------------------------------------------------------------------------------------
   labyrinth server                                      +------- user launches `./D'
                            ||                           |              |
           |                ||          lab client <----+               |
           |                ||            |                             |
           |<-----------------------------| tcp connect                 |
           |                ||            |                             |
           |<~~~~~~~~~~~~~~~~~~~~~~~~~~~~~| sign-up name                |
           |~~~~~~~~~~~~~~~~~~~~~~~~~~~~~>| receive session id          |
           |                ||            |                             |
           |                ||            |<----------------------------| create labyrinth
```

The client program should be prepared to consume up to three arguments:

- the TCP address of the server, default: `127.0.0.1`
- the port number at the server; default: `8000`
- the name of the user; default: `John Doe`

Processing Phase

```
----------------------------------------------------------------------------------------------
   labyrinth server         ||       lab client                      user enters requests
                            ||                                           |
           |                ||            |                              |
           |                ||            |<-----------------------------| add token requests
           |                ||            |                              |
          ...              ....          ...                            ...
           |                ||            |                              |
           |                ||            |<-----------------------------| reachable? query
           |<~~~~~~~~~~~~~~~~~~~~~~~~~~~~~| send batch                   |
           |~~~~~~~~~~~~~~~~~~~~~~~~~~~~~>| receive at response          |
           |                ||            |----------------------------->| render response
           |                ||            |                              | ...
           |                ||            |<-----------------------------| add token requests
           |                ||            |                              |
          ...              ....          ...                            ...
           |                ||            |                              |
           |                ||            |<-----------------------------| reachable? query
           |<~~~~~~~~~~~~~~~~~~~~~~~~~~~~~| send batch                   |
           |~~~~~~~~~~~~~~~~~~~~~~~~~~~~~>| receive at response          |
           |                ||            |----------------------------->|
          ...              ....          ...                            ...
```

Shut Down Steps

```
----------------------------------------------------------------------------------------------
   labyrinth server         ||       lab client                      user closes STDIN
           |                ||            |                              |
           |                ||            |<--------------------------- | ^D to client
           |<-----------------------------| tcp disconnect              ---
           |                ||           ---
```

User Requests

The user employs the exact same JSON commands as specified in Assignment C

TCP Messages

Note the difference between the labyrinth creation JSON request and the protocol message passed through TCP.

message	well-formed JSON format	denotation
sign-up name	string	“observations” are identified

message	well-formed JSON format	denotation
session ~~name~~ id	string (to distinguish connections)	using a unique session id

labyrinth creation	`["lab", [string, ...],`
	`  [[string, string], ...]]`
		sets up labyrinth with nodes
		. . . and edges

batch	`[["add",Color,string], ...,`
	` ["move",Color,string]]`
		a batch adds tokens. . .
		. . . and ends in a query

response	[ADD, . . . , Boolean]	the ADDs (see below) are well-formed but invalid ADD requests, the Boolean is the answer to the move query

From the perspective of the client program, all JSON values that match the above format are well-formed and valid, and can be sent to the server. If the user enters JSON that does not represent a well-formed request, the client program says

```
["not a request", JSON]
```

where `JSON` stands for the JSON the user entered.

From the perspective of `server`, validity requires the satisfaction of additional constraints:

message type	well-formed shape	validity
LAB =	`["lab", LON, LOE]`	
LON =	array of Nodes	
Node =	String	
LOE =	array of 2-element arrays	each of which contains two Nodes meaning the strings must be in LON
ADD =	`["add", Color, Node]`	the Node must be in LON
QQ =	`["move", Color, Node]`	the Color must have been set, and the Node must be in LON
Color =	String, one of: `"white"`, `"black"`, `"red"`, `"green"`, or `"blue"`	

The server will shut down the connection if:

- the “batch” command is ill-formed,

- the “create” command is invalid, or
- the “move” command is invalid.

Any ADD request which is well-formed but invalid gets sent back as a part of the response to a batch.

The client program `D` renders responses as quasi-English JSON for the user as follows:

- `["the server will call me", String]`
- `["invalid", ADD]`
- `["the response to", QQ, "is", Boolean]`