Assignment - SW Engineering

Your Task

Your task is to create a console application that in real-time transform the information about objects from the server and relay them at fixed time intervals to the console. The included SAAB.jar is a server application that provides the objects using a TCP/IP connection. A client is expected to connect via TCP/IP to the console application. Once connected, at fixed time intervals, relay the latest current status of the objects according to certain conditions.

The objects from the server have different types (1-3), where type 1 and type 2 are of the same category and type 3 belong to the second category. In the output channel (to the client), the objects should be categorized according to type, category, and current distance from a certain point in a coordinate system. The objects coordinates are updated over time thus the application shall relay the latest status on the output at fixed time intervals.

Requirements:

- Implement using C++.
- The incoming objects categorizations shall be updated to reflect type, category, proximity to the Cartesian coordinates (150, 150).
- The application shall log errors to `std::clog`.
- In the output, which occurs every 1'500 milliseconds, the objects latest type/categorization/proximity level according to the requirements in the *client application* section below.

Example information flow:

(server) ==real-time==> (console-app) ==fixed-interval==> (console)

Server Application

The software package sent to you contains a server application that via TCP/IP delivers data in text format. The server application is configured to communicate on port 5463.

Configuration and Start of the Server

The server application configuration file looks like this:

```
SERVERPORT=5463             # Default server port
MAP=map.gif                 # Path to map
```

To start the server from the command prompt:

```
java -classpath SAAB.jar com.saabtech.server.SAABServer
```

Server Data Specification

Data delivered from the server looks like:

```
ID=<LONG>;X=<INT>;Y=<INT>;TYPE=<INT>
```

The table below describes the data fields in detail.

Field	Type	Example data	Description
ID	LONG	2691882127991893	ID-number on the object
X	INT	119	X-coordinate for the object
Y	INT	227	Y-coordinate for the object
TYPE	INT	2	Type-id for the object

Example 1:

```
ID=2691882127991893;X=250;Y=150;TYPE=3
```

Example 2 - Two objects with the same category:

```
ID=2691882127234543;X=199;Y=230;TYPE=1
ID=2691882127221587;X=229;Y=310;TYPE=2
```

Console application

The application will connect via TCP/IP and receive data in binary format.

Console Data Specification

Once the console application connects to the server, data delivered from the console application every 300 milliseconds shall start like this:

Parameter Name	Units	Data Type
preamble	`0xFEFF`	32-bit int
count		32-bit int

`count` is the number of objects to directly follow. Each object after the preamble has the following format:

Parameter Name	Data Type
Id	64-bit int
X	32-bit int
Y	32-bit int
Type	32-bit int
Color	4 bytes

Where the 4-byte color sequence representation is:

- *Red*: `0x1B 0x5B 0x31 0x6D`
- *Yellow*: `0x1B 0x5B 0x33 0x6D`
- *Green*: `0x1B 0x5B 0x32 0x6D`

Objects are colored according to:

- Category 2 objects: *Yellow* unless closer than *100* from the designated coordinate then *red*.
- Type 1 objects: *Green* unless closer than *75* from the designated coordinate then *yellow* and if closer than *50* then *red*.
- Type 2 objects: *Green* unless closer than *50* from the designated coordinate then *yellow*.