Algonquin College Logo

# SCHOOL OF ADVANCED TECHNOLOGY

### ICT - Applications & Programming

### Computer Engineering Technology – Computing Science



A31

Game C/S Model

Team:

John Vinh - Id: 041-004-759

NumPuz Proposal

***This template is suggested (not mandatory) to answer A31 Specification.***

|  |  |
| --- | --- |
| **Part**  **1** | **C/S Architecture** |

* 1. **Server Model**

*Describe how your server interface should be organized and the main methods to be defined*

* + - ***Example****:*

**Example** (see A31 specification)

INTERFACE:

Class: NumPuzServer

→ Components:

JLabel: labPort

JTextField: txtPort

JButton btnResults

JButton btnEnd

JTextArea results

CONTROLLER:

Class: NumPuzServer – Object: “**server**”

→ Method: Start:

Start method will initialize the server and start listening for connections on one specific port.

→ Method: End:

The end method will close all the threads associated with each client. Afterwards, it will close the server.

→ Method: Results:

The results method will show the scores and statistics of each player in the results JTextArea.

* ***Note****: The professor interface continues being a proposal. Focus on your ideas using the best user experience.*
  1. **Client Model**

*Describe aspects of your client (interface and methods) considering the proposed idea.*

**Example** (see A31 specification)

INTERFACE:

Class: NumPuzClient

→ Components:

JLabel: labUser

JTextField: txtUser

JLabel: labServer

JTextField: txtServer

JLabel: labPort

JTextField: txtPort

JButton: btnConnect

JButton: btnEnd

JButton: btnNewGame

JButton: btnSendGame

JButton: btnReceiveGame

JButton: btnSendData

JButton: btnPlay

JTextArea: logTextArea

CONTROLLER:

Class: NumPuzClient – Object: “**client**”

→ Method: Connect:

Initialize a thread. Read the Server and Port from the JTextFields and initialize a connection to the NumPuzServer with those details.

→ Method: End:

Send a message to the server that the connection has closed. Then, close the thread and the connection for the client.

→ Method: SendGame:

Send the current game configuration to the server, allowing another client to receive it.

→ Method: ReceiveGame:

Receive a game configuration from the server which has been sent by another client.

→ Method: Play:

Start playing the game (open up the window from A22).

→ Method: Send:

After the game is finished, send the details of the game to the server.

* 1. **Protocol Proposal**

*Finally, what is your idea to define the protocol to be used.*

**Example** (using the string definition mentioned in the A21 specification)

CONFIGURATION STRING:

Class: NumPuzModel

→ Property: String: gameConfig:

→ Format: <dim><dataSeparator><dataConfig>, where:

→ <dim> = integer (from 2, 3, etc.)

→ <dataSeparator> = comma (,)

→ <dataConfig> = chars (example: 1-9), obeying the formula (dim2)2.

→ Example:

3;1,2,3,4,5,6,7,8,0

2;F,o,u

PROTOCOL P1:

→ protocolSeparator: hashtag (#)

→ Format: <clientId><protocolSeparator><data>

→ Example: 1#2;F,o,u

|  |  |
| --- | --- |
| **Part**  **2** | **Game Evolution** |

* 1. **Notes about upgrading the game**
  + *Describe the main modifications to be proposed in the C/S version of the game.*
    - *What are the differences between the original proposal (A11 / A21) and the current project to be developed (A31).*
    - *If so, explain why you need to do some adjustments.*

**Example** (About MVC modifications)

MODEL component:

Public methods to change private data (ex: dataConfig), that can receive inputs, but evaluate if they are valid.

// CONTINUE…

There needs to be a new JFrame application for both the client and the server. The client should start the application from A22 after the “Play” button is clicked. The application from A22 should be adjusted to send the data of the user, a name, a score, and their moves, to the server.

* 1. **GitHub / Database Integration (Bonus)**
  + *The use of GitHub is also a bonus to be considered:*
    - *Be sure that you can inform the updated repository and branch.*
    - *TIP: To avoid problems, also include the document (template answer) in the BrightSpace.*
  + *Considering this proposal for 3-tier architecture using Databases, define:*
    - *What to persist.*
    - *What is the DB datatype to be used.*
    - *How frequently to update.*

The database should have a table **scores** consisting of columns:

VARCHAR(30) name,

INT moves,

INT score,

After a game is finished, the database should have a new row inserted into it (updated) with the above columns.

**References**

*[Include eventual references used here]*

Algonquin College

Spring / Summer, 2022