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

JCheckBox chkFinalize

JButton btnEnd

CONTROLLER:

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

→ Method: Start:

try (

NumPuzServer **server** = new NumPuzServer (portNumber);

NumPuzClient client = **server**.accept();

}

→ Method: End:

for (Thread thread : connections) {

thread.interrupt()

}

* ***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: Start:

try {

NumPuzClient **client** = new Socket(hostName, portNumber);

} catch (IllegalArgumentException e) {

e.printStackTrace();

}

→ Method: Connect

→ Method: End:

* 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:

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

text;M,y, ,g,a,m,e,!,•.

PROTOCOL P1:

→ protocolSeparator: hashtag (#)

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

→ Example: 1#3,numerical;1,2,3,4,5,6,7,8,0

|  |  |
| --- | --- |
| **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…

* 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.*

**References**

*[Include eventual references used here]*

Algonquin College

Spring / Summer, 2022