Algonquin College Logo

# SCHOOL OF ADVANCED TECHNOLOGY

### ICT - Applications & Programming

### Computer Engineering Technology – Computing Science



A31

Game C/S Model – Collaboration Diagram

Team:

Akpoguma Daniella - Id: 041075624 / Philogene Villanueva - Id: 041063813

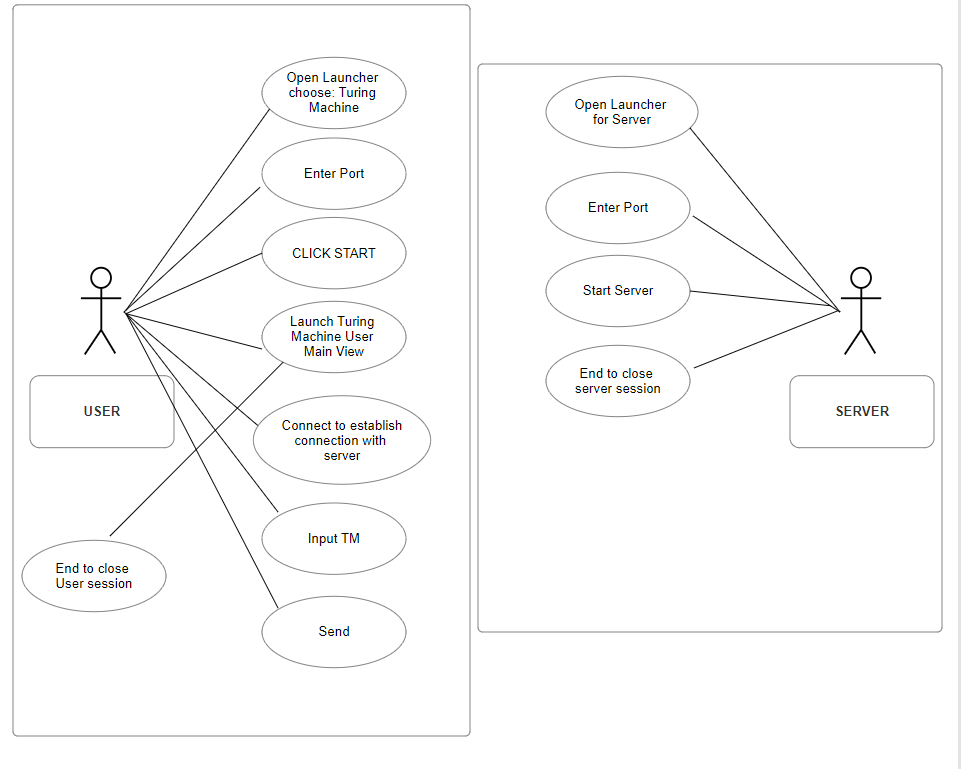
TM Model C/S Proposal

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

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

* 1. **UC Model (1pt)**

🛠 Define the UC diagram and the UC table (actors and functionalities).

****

**Fig. 1 – C/S Model for Chat**

**Actors table**:

|  |  |
| --- | --- |
| **Actors** | **DESCRIPTION** |
| Server | The Server actor embodies the server-side component of the application. It is responsible for managing and processing requests from users. |
| User | The User actor represents an individual interacting with the application. Users initiate requests, provide input, and receive outputs from the system. |

**UC table**:

|  |  |
| --- | --- |
| **Use Cases** | **CLIENT DESCRIPTION** |
| Login | This use case facilitates users in accessing the Turing Machine application by providing a login process, to gain access to the application's features and functionalities |
| View Task | Users can utilize this use case to explore and review the list of tasks available within the Turing Machine application |
| Create Task | This use case allows users to generate new tasks within the Turing Machine application. |
| Update Task Status | Users have the ability to modify and update the status of a task using this use case. |
| Sync Task | This use case enables users to synchronize tasks seamlessly between their local environment and the server. |

|  |  |
| --- | --- |
| **Use Cases** | **SERVER DESCRIPTION** |
| Authenticate User | This use case involves the server's responsibility to authenticate users during the login process, ensuring a secure and valid entry into the Turing Machine application. |
| Fetch Task | This use case is responsible for retrieving task information from the server. It enables the server to send task data to the user for display. |
| Update task Status | This use case empowers the server to update the status of a task based on user interactions, and the consistency of task statuses across the entire application. |
| Sync Task | This use case facilitates the synchronization of tasks between the user and the server. |

* 1. **ClassD (2pt)**

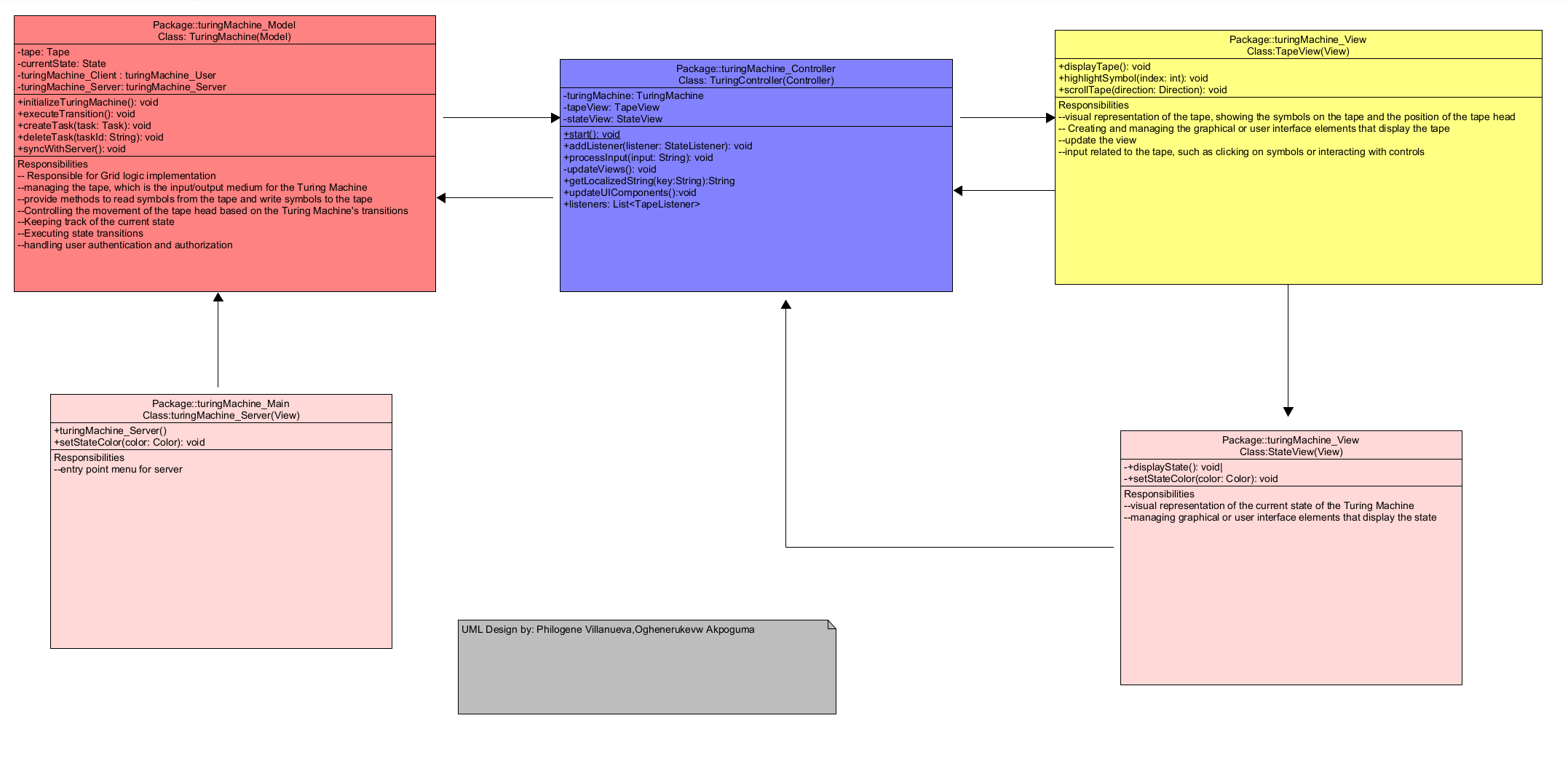
🛠 To draw the diagram, you can use tools (ex: <https://app.diagrams.net/>) or desktop applications (ex: Visio / Powerpoint) or simply take photos from drawings.

**Class Diagram** (change this diagram to accommodate the actors and functionalities to be used):

A diagram of a computer program

Description automatically generated with medium confidence

UML for User



UML for Server

**Fig. 3** – ClassD for a chat[[1]](#footnote-1)

**Class table**:

|  |  |
| --- | --- |
| **Class name** | **TuringMachine (Model)** |
| Inner Fields[[2]](#footnote-2) | tape: instance of tapeView  currentState: instance of the current state  user: User instance  server: Server instance |
| Relationships[[3]](#footnote-3) | TuringController class |
| Methods | initializeTuringMachine(): void  executeTransition(): void  createTask(task: Task): void  deleteTask(taskId: String): void  syncWithServer(): void |

|  |  |
| --- | --- |
| **Class name** | **TuringController (Controller)** |
| Inner Fields[[4]](#footnote-4) | turingMachine: instance of model class  tapeView : instance of tapeView  stateView: instance of stateView |
| Relationships[[5]](#footnote-5) | TuringController class  TapeView class  StateView class |
| Methods | initializeTuringMachine(): void  executeTransition(): void  createTask(task: Task): void  deleteTask(taskId: String): void  syncWithServer(): void |

|  |  |
| --- | --- |
| **Class name** | **TapeView (View)** |
| Inner Fields[[6]](#footnote-6) |  |
| Relationships[[7]](#footnote-7) | TuringController class  StateView class |
| Methods | displayTape(): void  highlightSymbol(index: int): void  scrollTape(direction: Direction): void |

|  |  |
| --- | --- |
| **Class name** | **StateView (View)** |
| Inner Fields[[8]](#footnote-8) |  |
| Relationships[[9]](#footnote-9) | TuringController class  TapeView class |
| Methods | displayState(): void|  setStateColor(color: Color): void |

* 1. **UML Collaboration Diagram**

*Describe the collaboration diagram to be used.*

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

A diagram of a machine

Description automatically generated

**Fig. 4** – Collab model for a chat[[10]](#footnote-10)

**References**

OpenAI. (2020). OpenAI GPT-3 (ChatGPT). Retrieved from <https://www.openai.com>

Algonquin College

Fall, 2023

1. See <https://www.codeproject.com/Articles/443660/Building-a-basic-HTML-client-server-application>. [↑](#footnote-ref-1)
2. The inner fields and relationships together are properties from the class. [↑](#footnote-ref-2)
3. In the diagram, you can see relationships (ex: association / aggregation) between classes. In the implementation, these relations imply in the inclusion of other classes as fields. [↑](#footnote-ref-3)
4. The inner fields and relationships together are properties from the class. [↑](#footnote-ref-4)
5. In the diagram, you can see relationships (ex: association / aggregation) between classes. In the implementation, these relations imply in the inclusion of other classes as fields. [↑](#footnote-ref-5)
6. The inner fields and relationships together are properties from the class. [↑](#footnote-ref-6)
7. In the diagram, you can see relationships (ex: association / aggregation) between classes. In the implementation, these relations imply in the inclusion of other classes as fields. [↑](#footnote-ref-7)
8. The inner fields and relationships together are properties from the class. [↑](#footnote-ref-8)
9. In the diagram, you can see relationships (ex: association / aggregation) between classes. In the implementation, these relations imply in the inclusion of other classes as fields. [↑](#footnote-ref-9)
10. See <https://www.researchgate.net/figure/Collaboration-diagram_fig1_268362338>. [↑](#footnote-ref-10)