Documenting Design & Implementation  
Design

* Objectives and overview
* Design philosophy
* Server-side design classes: UML diagrams
* Client-side design classes: UML diagrams

Implementation

* Objectives and overview
* Application layer implementation: Sequence diagrams and code sections for interesting code
* Presentation layer implementation: Sequence diagrams and code sections for interesting code
* Service layer implementation: Sequence diagrams and code sections for interesting code
* User Manual with screen capture

##### Conclusions

**The specification of a protocol should include:**

* The sequence of data exchange, which can be described using a time event diagram.
* The specification of the format of the data exchanged at each step.

### Service Location

* The client locates the service by connecting to the server’s IP address and port number over a secure SSL/TLS connection.

### Sequence of Inter-process Communication

* The client initiates a connection request to the server using SSL/TLS.
* The server listens for incoming client connections and accepts them
* Upon successful connection establishment, the client and server perform an SSL/TLS handshake to establish a secure communication channel
* Client acknowledges the server's acknowledgment over the secure connection
* Client sends requests to the server
* Server processes requests
* Server sends back responses to the client
* Communication continues until the session is terminated

### Representation and Interpretation of Data Exchanged

* Requests and responses are structured in a predefined format, agreed upon by both the client and the server
* Data exchanged between client and server follow a common schema and are encrypted using SSL/TLS
* JSON data format to be used for encoding commands(requests), responses, and other data.

**Error Handling**

* Define error codes and messages for common error scenarios
* Both client and server must handle exceptions and error appropriately
* Error responses must be meaningful and informative

### Service Session Management

* Upon connection establishment, a service session is initiated over SSL/TLS connection
* The server assigns a unique session identifier to each client session
* During initiation, the server authenticates the client’s credentials.
* Session management includes maintaining the state of the session, including state and uploaded messages
* The server periodically updates the client with relevant game information and state changes (asynch comms)
* When a session ends, either due to client disconnection or session timeout, the server cleans up the session resources and state information associated with that client

**Security**

* Implement SSL/TLS to encrypt communication between client and server