

**Architectural Style**: An *architectural style* defines a family of architectures constrained by component vocabulary e.g., Layers and cells between them, Topology such as Stack of layers and semantic constraints such as a layer may only talk to its adjacent layers.

* **Event Driven Architecture**: The battleship game is developed based on the event driven architecture. In EDA the components do not invoke each other explicitly, but generate signals called events. To receive events, objects can receive events at ports (statically or dynamically bound) or can register for event notification (e.g. via callback). This architecture supports the Java Swing provision of user interface related functionality.

**Advantages of EDA:**

* It supports components reuse because of less coupling between the modules.
* System can be evolved easily by registering new components at construction time or run time.

The listeners used in various source files are listed below and perform the following functionalities:

|  |  |
| --- | --- |
| FILE | EVENTS |
| Battle.java | Mouse Events for positioning ships and hitting on boards |
| Board.java | No mouse events included |
| Ship.java | No mouse events included |
| AI.java | No mouse events included |

**Checked Exceptions**:

Checked exceptions are forced by compiler and used to indicate exceptional conditions that are out of the control of the program.

**Unchecked Exceptions**:

Unchecked exceptions are occurred during runtime, the occurrences of which are not checked by the compiler. They will come into life/occur into your program, once any buggy code is executed.

**2 Player Description**: -

We use UDP to establish connection between the two machines on which the 2 players play the game.

The connection is established once both players place their ships and then they press the P2P (Player to Player) button.

Once the connection is established players are prompted to choose game style as usual.

After that, either players can start hitting the opponent players ships on their respective opponent boards.

Scenario 1) Send: -

If player 1 chooses a coordinate on Opponent board to shoot, those coordinates are sent via the UDP socket to player 2.

Player 2 receives this message and parses it to extract the coordinates. Using these coordinates Player 2 hits on his/her own board.

Scenario 2) Receive: -

Now the player 2 sends the feedback to player 1 stating whether the cell which player 1 chose is a hit or a miss.

As per the feedback received on the socket, player 1 parses it again and