**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 | ACTION |
| Battle.java |  |  |
| Board.java |  |  |
| Ship.java |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

For example, when a customer purchases a phone, the phone's state changes from "for sale" to "sold". A phone dealer's system architecture may treat this state change as an event whose occurrence can be made known to other applications within the architecture. From a formal perspective, what is produced, published, propagated, detected or consumed is a (typically asynchronous) message called the event notification, and not the event itself, which is the state change that triggered the message emission. Events do not travel, they just occur.