**Game-Time –** Moderates the **rounds**. Contains the current **round** and the next **round**.

**Map** – A 2-dimensional array of **spaces**. Owned by the **DM.** Can initiate **events** from **event-conditions**. Each **event-condition** array is checked at different frequencies:

* “Ability-start” is tested after any **ability** is initiated.
* “Ability-end” is tested after any **ability** resolves.
* “Round” is tested at the start of every **round.**
* “Turn” is tested at the start of every **turn**.

**DM –** Dungeon Master. A user that has admin privileges over every **unit** and the **map**. He/she can opt to confirm any change to the game state (**abilities, turns, event-conditions, etc**.).

**Round** – A list of **turns.** Each **turn** is executed in order. The round ends when all **turns** have resolved. **DM** can manually change **turns**.

**Turn –** The time a **unit** has to choose its actions. After finishing its **actions-list**, the turn ends.

**Movement** – Occurs when a **unit** changes **spaces**. The destination **space** must satisfy the **unit’s** **mobility** requirements; if the requirements are not met, the movement does not occur. There are three types of movement:

* “Continuous” movement can only occur if there is a satisfactory orthogonal path of spaces between the source and destination spaces. The path is satisfactory if every space on the path satisfies the **mobility** requirements. This movement can also fail due to the route being too lengthy.
* “Linear” movement can only occur if all of the spaces along a line from the center of the source space to the center of the destination space satisfy the **mobility** requirements.
* “Flash” movement does not require a direct path.

**Space** – Contains a **terrain** and may contain **units**.

**Event** – An abstract class which is capable of modifying **units** and the **map.** Can initiate other events. When an event occurs, it MUST record itself in the **system-log.**

**System-Log** – A record of previous events. Can be parsed by **events** and **event-conditions**.

**Event-Condition –** Allows **events** to be passively initiated.Contains a condition and an array of **events.** Once the condition is tested and satisfied, the event-condition triggers all of its **events** in order. Can be set to require **DM** confirmation, who can choose to “accept” (**events** proceed), “skip” (**events** are not triggered), or “cancel” (return to before the event-condition; note that if nothing was changed, the event-condition will ask for **DM** confirmation again).

**Terrain** – Has a **sprite** and a type. The type is used as a requirement for **unit mobility** and can be used in **events**. Has the following **event-condition** arrays:

* “Unit-enter” is tested every time a unit enters the space.
* “Unit-exit” is tested every time a unit exits the space.
* “Round” is tested at the start of every round.

**Sprite** – A 2-dimensional texture.

**Super-Unit** – A set of **units** that are combined to be equivalent to a large **unit** spanning several **spaces**. The sub-**units** share **turns** and **status.**

**Unit –** An entity that is not bound to a space unless otherwise specified. Has a **sprite**, **mobility, status**, **action-list**, the coordinates of the unit’s residing space, and a set of **event-condition** arrayswith the following frequencies:

* “Ability-initiate” is tested every time this unit initiates an **ability.**
* “Ability-receive” is tested every time this unit is **targeted** by an **ability.**
* “Status-change” is tested after this unit’s **status** is changed.
* “Turn-start” is tested every time this unit starts its **turn**.
* “Turn-end” is tested every time this unit ends its **turn**.

**Mobility –** The information used for **movement**. Handles the ability to share and pass through spaces with other units.

**Action-List –** The actions available to the unit every turn. It is often a combination of **abilities, ability** arrays and **movement**. **Events** and **abilities** can cancel other actions when used.

**Status** – A set of information specific to a unit. Always contains a name and **mobility** information. Other information is used to calculate the outcome of **events**.

**Ability** – An **event** that is initiated by a **unit**. When initiated, the **unit’s** **owner** selects a **target** for the event. Can require **DM** confirmation, who can choose to “accept” or “deny.”