Marvel Tactics Encounter Creation

**Summary**

Hello, in order for our team to create as much content as we can by launch we need to regulate the production of combat maps and make sure we can create a substantial amount of variation in our different combat encounters regardless of how many ingredients we have.

The concept is that with little content (a handful of maps and enemies, a single objective) we can build out the full amount of actual encounters we need. Then, as new maps, characters, objects and objectives are added to the game we can add them to the encounters, creating increased variation and diversity as we go, without having to re-write or re-implement the encounters themselves.

**Terminology**

Here’s a rundown of the common definitions we’re using:

* **Chapter:** Marvel Tactics tells stories, at a macro level, in chapters. A chapter is a self-contained story, within the overall arc of the game’s narrative, with a specific set of enemies, locations, events and ultimately a resolution (generally in the form of a boss battle).
* **Missions**: Chapters are subdivided into missions. Each mission is set in (generally) a single geographic location with one or two main antagonists and “grunt” factions. A Mission is comprised of up to five *encounters*.
* **Encounter**: An encounter is the smallest atomic piece of our game’s story. An encounter is one gameplay session pitting the player’s attackers against AI-controlled defending team of villains and super villains in a combat map.
* **Map**: This is the combat map on which encounters are played. The map is a bunch of static and dynamic objects as well as encounter data like spawn points that make up the *playfield* for an encounter

**Obligatory Diagram**

Here, also, is a visual breakdown of how we’ll be creating the story content in MarTac.



**Building Encounters**

In their purest form, encounters should exist as entities with a unique name and a variety of pointers to other data which will feed the combat engine. The relevant data points are:

* **Name**: a unique name for the encounter
* **GUID**: automatically assigned by Cerebro
* **Description**: used where the UI need a descriptive paragraph for encounter previews
* **Map**: what combat map the encounter will use
* **Spawn Lists**: information about what characters and objects will spawn at which points, and with what weighting
* **Objectives**: any special objectives that can result in victory or failure beyond the default objective of eliminating all enemies
* **Story Dialogues**: what “toast” sequences will play before, during and after the encounter

**Map Standardisation**

In order to keep things modular and make sure any map can be plugged into any encounter (aside from a handful of one-off handcrafted missions like Boss battles) each map will need to be constrained to a set of standardisations with regards to the number of spawn points and general size. The actual rules are forthcoming but the end result is that bespoke maps or non-conventional numbers of spawns will break the system and the designers will be punished to the fullest extent of the law.

**Spawn Lists**

The bulk of being able to control the spawning of enemies, from bulk lists for random dispersal to specifically handcrafted spawn lists for critical story missions, rests mostly with Cerebro. Here’s a rough idea of how the tool should look for creating spawn lists for populating encounters.



The main components here are:

* **Character or Group**: this points to a specific character entity to spawn or to a character group entity, which is a pre-created, arbitrary list of characters. For instance, a group could be ten instances of the same AIM Grunt, seven individual AIM Grunt variants or a list of twenty characters comprising of ten AIM and ten HYDRA grunts.
* **Character Spawn Percentage**: for each character or group selected, a spawn percentage can be set. This allows a character to always appear, sometimes appear or - technically - never appear, which would be a weird thing to set up. If no percentage is set the chance to spawn is equally weighted with any other characters or groups with no percentage set.
* **Spawn Point, Range or Group**: for every character or group set above there must be at least one entry point chosen an entry point can be a single spawn point (e.g. D1), a specific range (e.g. D1 - D5) or a pre-set group entity that lists specific spawn points.
* **Spawn Point Percentage**: for every spawn point, range or group called out there needs to be a percentage weight of the characters) spawning there, or as above a blank value will assume equal weighting with other blank spawns.

The core concept is that an encounter can be as random (by pulling its population from large buckets of characters and having them appear across random spawn points) or as unique (through calling only unique characters to manifest only at single specifically determined spawn points) as it needs to be.

**Objectives**

If an encounter has no specific objectives set, then it it will only operate under the default "eliminate all enemies" objective. Otherwise there should be controls for a Designer to build in objectives for specific encounters:

* **Type**: specifies the kind of objective, pre-coded based on the individual types we will be supporting. Currently for launch this can be Eliminate Character, Destroy Object or Hold Area.
* **Target**: this sets the target entity(s) for the objective type(s) called
* **Trigger**: this holds any additional information about what is needed to fulfill the objective conditions (# of objects, main character)
* **Result**: the result of fulfilling the objective condition or trigger (victory, percentage of victory?)

**Story**

We have a separate spec that covers story narrative “toasts”, but the important not here is that an encounter should set what toasts will be associated it either before, during and after, by calling them as entities, with the hooks and trigger conditions held separately within the toast entities themselves.