Title — *Systemic design in Unreal Engine.*

Objective — The goal of this prototype is to provide a basic systemic behavior with only built-in Unreal Engine features.

Important: this prototype won't provide a fully fleshed-out system, but can give you some ideas on how to expand it or build your own system.

Requirements — In order to be able to understand the prototype you need to be at least familiar with Unreal engine and the GAS (Gameplay ability system).

Detailed Design —

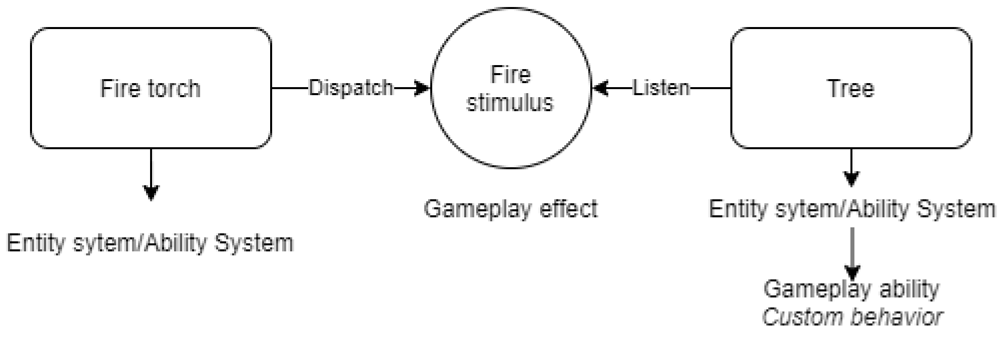
Reminder: Systemic games are games that are created such that all their individual systems can reach out and influence one another.

This will enforce the credibility of the world and gives to the player more freedom.
For instance, if you use fire against an element that can burn, you'll expect it to burn, in the same way, if you put water on the fire, you expect the fire to be extinguished.

For this prototype I decided to only use common element interaction such as "water", "fire" and "electricity". With those three elements you can already do great combination and behavior.

So how do we create systems that can interact which each other without hard scripting them?

Through **stimulus**.



Each world element that can be its own system has an entity system component attached to it.

This component contains a list of stimuli where each stimulus can be aware of a behavior or spread a behavior to another one.

In order to achieve that in the engine I decided to use the Gameplay Ability System.

So here is how I used the system and I translated it for the systemic design purpose.

Gameplay effect: Normally they're used for applying attribute modifiers. In my system I use them as stimuli, they have a duration and apply tags while they're alive.

A gameplay effect can be cancelled by another effect and fit perfectly the system intention.

For example, the fire stimuli can be cancelled by the water stimuli.

Gameplay ability: they're used as listener for specific stimuli and can run a dedicated behavior.

For example, a stimulus with fire dispatched to a "dynamite" entity will trigger the ability who ignites and blow up the dynamite after a while.

Gameplay cue: Used for player feedback (particle, sound, …), the gameplay cue is spawned when a stimulus is applied and will disappear when the stimulus is over.

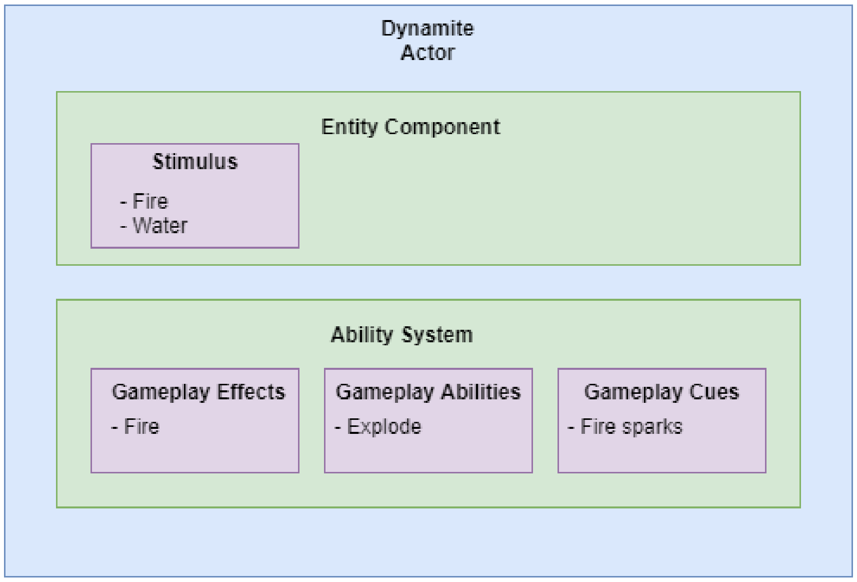


Figure 1 - Dynamite actor example.

Below a matching table for each element and stimulus.

	Wood	Water	Dynamite	Metal
Fire	Spread	Extinguish	Ignite	Nothing
Water	Extinguish	Nothing	Extinguish	Nothing
Electricity	Nothing	Spread	Nothing	Spread

In order to keep thing simple, the only way to apply a stimulus to another is by overlapping element to each other. However, it's up to you to choose the way you want your system to interact with other.

Additional notes —

To conclude, you can easily improve this system and adapt it to suit your game type.

The attribute system from GAS is a powerful feature and can be easily integrated. You can use it for managing DoT (damage over time) and world element behavior.

In the end, all design rules and decision rely to your game genre and it's up to you to find the right solution for your game.