ECS in Game Development

Jorge Pinto Sousa (He/Him), Luiz Jardim (Viceroy/Viceroyer)

July 2021

Critical Techworks

Entity–component–system (ECS)

Entity–component–system (ECS)

Separate data from behaviour.

Entity–component–system (ECS)

- It addresses some of the problems with object orientation while promoting code reusability, extendability, maintanability and paralle... para... parallelizability (is this a word?).
- One of it's greatest features is easily modifying behaviour at runtime.

ECS - Characteristics

ECS has:

- **Entities** are unique "things (identifiers).
- **Components** which are just datatypes without **behaviour**
- **Systems** which are functions that will act in **Entities** that have a certain set of **Components**.

ECS - Characteristics

Also:

- **Entities** can contain zero or more **components**.
- **Entities** can dynamically change **components**.

There are frameworks that implement and enable this design pattern.

In the scope of this presentation we'll use Bevy.

ECS - Characteristics

Despite the *"by the book"* definition of ECS, which has to have entities, component and systems, sometimes in practice that is not the case.

Usually anything that let's you add stuff to entities and then querying them for those things, are usually considered to be an ECS.

ECS is not EC

In an EC framework, components contain both data and behaviour, and that behaviour is executed directly on the component.

(Place example here)

Is ECS too different from OOP?

Composition over inheritance.

Is ECS too different from OOP?

Exposed Plain Data Objects over encapsulation.

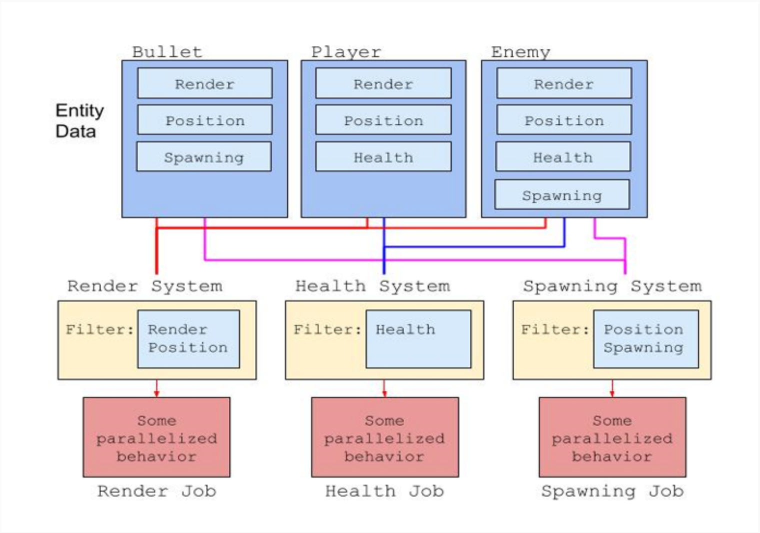
Is ECS too different from OOP?

Separate data and behaviour.

Is ECS too different from OOP?

OOP object instances are of a single non-changing type, while entities can have dynamically changing components.

ECS - An example



References

- Bevy
- ecs-faq
- Get Started with the Unity* Entity Component System (ECS)