

# **Xast:** Comparing relational and event-driven ECS models

Oleksandr Hnutov

December 29, 2025

# Problem

Dealing with single-entity systems, like

$$\mathcal{R} \rightarrow \Delta\mathcal{R}, \text{ where } \Delta\mathcal{R} \subseteq \mathcal{R},$$

e.g. system, which moves player towards X axis

$$\{\text{Player}, \text{Transform}\} \rightarrow \{\text{Transform}\},$$

is straightforward, because both for relational and event-driven systems, formal definition will be the same, so it will have the same properties. We need to define formal definitions for systems, which process 2 and more entities both for the relational ECS model ( $\text{System}_R$ ) and event-driven ECS model ( $\text{System}_E$ ). Systems in the event-driven model are restricted to single-pass iteration over the event sequence and may not construct auxiliary indices whose size depends on  $|\text{Seq}(\text{Event})|$ .

As an example, we will consider a simple system with two entities:

**EnemyFollowPlayer:**

$$\begin{aligned} &\{\text{Player}, \text{Transform}\}, \\ &\{\text{Enemy}, \text{Transform}\} \\ &\rightarrow \{\text{Enemy}, \text{Transform}\}. \end{aligned}$$

## Formal definition

$$\text{System}_R: \mathcal{R}_1 \times \mathcal{R}_2 \times \cdots \times \mathcal{R}_n \rightarrow \Delta\mathcal{R},$$

where:

1.  $\mathcal{R}_i$  is a set of entities with a component set  $C_i$ ,
2.  $\Delta\mathcal{R}$  denotes a finite set of component updates applied to entities in  $\mathcal{R}$ .

$$\text{System}_E: \mathcal{R} \times \text{Seq}(\text{Event}) \rightarrow \Delta\mathcal{R},$$

where:

1. Event is a finite set of components  $C$ ;
2.  $\text{Seq}(\text{Event})$  denotes an ordered sequence, allowing only sequential access.

Let  $E$  be the set of enemies entities and  $P$  the set of players entities. Let  $n = |E|$  and  $m = |P|$ .

Let  $f: \text{Transform} \times \text{Transform} \rightarrow \text{Transform}$  be a pure function, which computes the next enemy position, moving to a player, based on player's and enemy's positions.

Let **choose**( $P, e$ ) be a deterministic selection function from players for any enemy.

Relational model system will be

$$\begin{aligned} \text{System}_R: \forall e \in E: \\ \text{targetP} &= \mathbf{choose}(P, e) \\ e.\text{Transform} &= f(e.\text{Transform}, \text{targetP}.\text{Transform}); \end{aligned}$$

If **choose** is  $O(1)$  (e.g., the **lookupMap** function on a precomputed map), total complexity will be the following:

$$T_R = O(n + m);$$

In worst case, if **choose** need to scan all the players, total complexity will be

$$T_R = O(n \cdot m).$$

Let Event be a tuple: (Player, Transform). Event-driven model system will be

$$\begin{aligned} \text{System}_E: \forall e \in E: \\ \forall \mathbf{ev} \in \text{Seq}(\text{Event}) \\ e.\text{Transform} &= f(e.\text{Transform}, \mathbf{ev}.\text{Transform}); \end{aligned}$$

So we every time we will scan the entire sequence of events, so total complexity is

$$T_E = O(n \cdot m).$$

## Theorem. Lower bound for event-driven ECS

Let  $E$  be a set of  $n$  enemy entities and  $P$  a set of  $m$  player entities. Assume a system processes enemies by iterating over a linear sequence of events  $\text{Seq}(P)$  and has no auxiliary indexing structure. Then any system computing a function

$$f : E \times P \rightarrow \Delta E$$

has worst-case time complexity  $\Omega(n \cdot m)$ .

## Proof

Let  $\text{Seq}(P)$  be a sequence containing exactly one event for each player. By assumption, the system can only access player data by iterating through  $\text{Seq}(P)$ .

For a fixed enemy  $e \in E$ , any algorithm that depends on player data must, in the worst case, inspect all  $m$  events to determine the relevant player (e.g. nearest player, target selection).

Thus, processing a single enemy requires  $\Omega(m)$  time.

Since the system processes all  $n$  enemies independently, the total worst-case time complexity is  $\Omega(n \cdot m)$ .