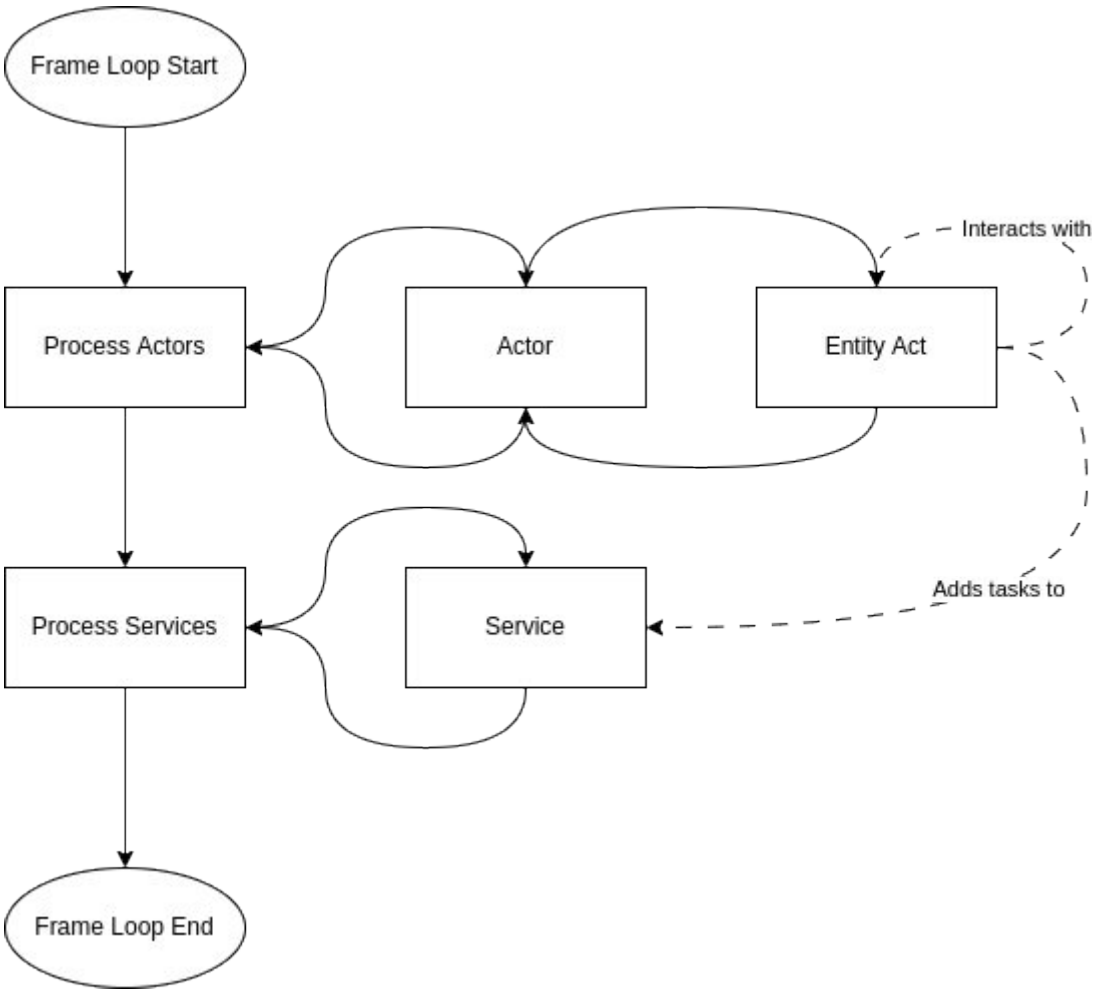

Preprosta zgradba projekta je razvidna iz zgornjega diagrama poteka. Projekt si jemlje zgled po Entity Component Framework-u.

Is samega diagrama je že takoj razvidno da je zanka, ki se bo največkrat ponovila, Entity Act. Velika verjetnost je, da se bo ozko grlo pojavilo prav v tistem predelu kode.

Implementiranih je par entitet, s katerimi se definira igralna logika igralcev na sceni:

- Age Entity
- Animal Behavior Entity
- Metadata Entity
- Movement Entity
- Needs Entity
- Observer Entity
- Position Entity
- Shape Entity

Od teh so Shape, Position, Observer, Metadata entite statične, kar pomeni da samo ponujajo funkcionalnost ali lastnosti, in same ne vplivajo na delovanje igralca.

V nasprotnosti so Age, Animal Behavior, Movement, Needs entitete dinamične, kar pomeni da vplivajo na delovanje igralca.

Najkompleksnejša od teh je Animal Behavior entiteta.

Možne izboljšava je, da se izračuni razvrščenih oddaljenosti shranjuje, in ne na novo vedno izračunava.

Načeloma pa je po celotni aplikaciji ena problematika, da se za pridobivanje entitet uporablja niz, hkrati pa se še pridobljena entiteta tipizira s strani razvijalca.

Dinamična analiza

Profiliranje enega zagona nam da sledeč profil

Z drevesnim pogledom in ignoriranjem delov kode nad katerim nimamo vpliva vidimo da največ časa preživimo v samem izvajanju logike ki jih imajo figure.

Total (samples)		Self	
93%	48,288	199	▼ RefreshDriver tick
86%	44,224	101	▼ FrameRequestCallback
85%	44,116	5	▼ loop http://localhost:5173/node_modules/.vite/deps/chunk-T3XJLO5L.js?v=2d8a39e0:17529:15
85%	44,029	12	▼ render$1 http://localhost:5173/node_modules/.vite/deps/chunk-T3XJLO5L.js?v=2d8a39e0:17507:17
64%	32,977	18	▶ WebGLRenderer/this.render http://localhost:5173/node_modules/.vite/deps/chunk-W7IJVTTO.js?v=2d8a39e0:17158:24
21%	10,955	15	▼ Logic/< http://localhost:5173/src/simulation/world/logic.tsx:46:11
21%	10,696	35	▼ act http://localhost:5173/src/simulation/figures/actors/actor.ts:28:5
20%	10,586	56	▶ act http://localhost:5173/src/simulation/figures/entities/animalBehaviorEntity.ts:11:5
0.1%	47	8	▶ act http://localhost:5173/src/simulation/figures/entities/movementEntity.ts:8:5
0.0%	9	5	▶ act http://localhost:5173/src/simulation/figures/entities/needsEntity.ts:14:5
0.0%	8	8	next self-hosted:458:26
0.0%	6	3	▶ act http://localhost:5173/src/simulation/figures/entities/ageEntity.ts:6:5
0.0%	4	2	▶ values self-hosted:506:21
0.0%	1	1	setToMovement http://localhost:5173/src/simulation/logic/movementCalculator.ts:14:15
0.2%	83	2	▶ setActors http://localhost:5173/src/state.ts:19:13
0.1%	76	27	▶ console.log
0.1%	48	34	▶ filter self-hosted:241:20
0.1%	30	9	▶ act http://localhost:5173/src/simulation/figures/service/globalServices.ts:19:5
0.0%	6	6	next self-hosted:458:26
0.0%	1	1	act http://localhost:5173/src/simulation/figures/service/animalMatingService.ts:12:5
0.1%	77	2	▶ MapControls2</< http://localhost:5173/node_modules/.vite/deps/@react-three_drei.js?v=2d8a39e0:93695:11
0.0%	7	5	▶ getDelta http://localhost:5173/node_modules/.vite/deps/chunk-W7IJVTTO.js?v=2d8a39e0:26409:10
0.0%	1	1	sort http://localhost:5173/node_modules/.vite/deps/chunk-W7IJVTTO.js?v=2d8a39e0:12752:15

Če smo še natančnejši, preživimo večinoma časa v izvajanju logike entitete za obnašanje živali. Ta entiteta je tudi naj kompleksnejša, zato ta ugotovitev ni presenetljiva.

Total (samples)		Self	
100%	10,586	56	▼ act http://localhost:5173/src/simulation/figures/entities/animalBehaviorEntity.ts:11:5
34%	3,632	1	▶ getClosestActor http://localhost:5173/src/simulation/figures/entities/observerEntity.ts:56:17
34%	3,605	631	▶ console.log
13%	1,424	756	▶ filter self-hosted:241:20
8.1%	860	4	▶ flatten http://localhost:5173/node_modules/.vite/deps/lodash.js?v=2d8a39e0:3246:24
4.5%	478	3	▶ getClosestActorVisibleToMe http://localhost:5173/src/simulation/figures/entities/observerEntity.ts:50:28
1.8%	189	15	▶ momentumFromFearOfHunters http://localhost:5173/src/simulation/figures/entities/animalBehaviorEntity.ts:76:27
0.9%	97	9	▶ closestMate http://localhost:5173/src/simulation/figures/entities/animalBehaviorEntity.ts:91:13
0.7%	78	16	▶ getOrderedNeeds http://localhost:5173/src/simulation/figures/entities/needsEntity.ts:23:17

Iz profilerja se da razvidet, da naječ na performanco vpliva klic funkcije getClosestActor od Observer Entitete.

Klic na konzolo, ki ima sicer performančno isto ceno kot getClosestActor klici, se bo ignoriral ker se uporablja samo za spremljanje stanja tekom razvijanja. Največjo ozko grlo je torej ocenjevanje razdalj od trenutnega igralca do drugih igralcev.

Povečevanje število igralcev je vodilo do primerljivih časovih preživljenih v funkcijah, torej je getClosestActor res mesto kjer se mora podati največ pozornosti.

Primerjava statičnega pa dinamičnega

Iz statičnega smo ugotovili in predvidevali v katerem odseku sistema se bo najverjetneje pojavilo ozko grlo. V dinamičnem smo to tudi potrdili. V dinamičnem smo sicer tudi dobili boljši pregled točno v katerem delu odseka se težava nahaja.