Komponentní systém pro herní grafický engine

Tomáš Polášek

Vysoké učení technické v Brně – Fakulta informačních technologií

Cíle

- Cílem byl návrh a implementace knihovny pro správu entit, která umožňuje vývoj pomocí *Entity-Component-System* paradigmatu.
- Mezi požadavky patřily:
  1. Možnosti paralelního provádění datových transformací.
  2. Efektivní použití vyrovnávacích pamětí.
  3. Intuitivní programovací rozhraní.
  4. Jednoduchá integrace do existujících projektů.

Důvody

- Obvyklé návrhové prostředky – jako *objektově orientované programování* – jsou pro návrh her příliš rigidní.
- Zvyšující se požadavky na oddělení jednotlivých částí herního enginu.
- Výkonnostní rozdíl mezi přístupovou dobou pamětí a rychlostí vykonávání instrukcí se stále zvyšuje.
- Vzniká *datově orientovaný návrh*, jehož cílem je vyšší efektivita práce s fyzickým hardware – vyrovnávací paměti, pipelining.

Entity-Component-System

- Návrhové paradigma založené na kompozici. Vychází z principů *datově orientovaného návrhu*.

ID	Comp1	Comp2	Comp3
Entity1	NULL	DATA	DATA
Entity2	DATA	NULL	DATA
Entity3	NULL	NULL	DATA

- Striktní separace logiky a dat, čímž umožňuje vyšší úroveň modularity.
- Skládá se ze tří částí – *entity*, *komponenty* a *systémy*.

Entity

- *Entity* reprezentují virtuální objekty uvnitř herního světa.
- Jejich hlavní funkcí je identifikace daného objektu a vazba na komponenty.
- Identifikátor *entity* je primárním klíčem v tabulce *entit*.

Komponenty

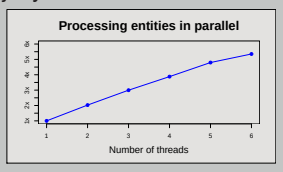
- Základní nosiče dat a zároveň nejmenší datová jednotka, kterou lze *entitám* přiřadit.
- Neobsahují „výkonnou“ logiku, pouze operace pro manipulaci dat.
- Sloupce tabulky *entit* obsahují jednotlivé typy komponent.

Systémy

- *Systémy* implementují datové transformace nad *entitami* a jejich *komponentami*.
- Transformují pouze takové *entity*, které obsahují požadované *komponenty*.

Výsledky

- Výsledkem je multiplatformní knihovna **Entropy**, která umožňuje návrh pomocí *ECS* paradigmatu.
- Implementována v programovacím jazyce *C++*.
- Využívá koncept *skupin*, které umožňují uchovávat seznamy filtrovaných *entit*. *Systémy* iterují pouze nad *entitami*, které již skutečně obsahují požadované *komponenty*.



Lineární škálování s počtem vláken

- Umožňuje paralelizaci na třech úrovních:
  - *Entity* – Rozdělení množiny *entit* na jednotlivá vlákna, která provádí požadovanou transformaci odděleně.
  - *Systémy* – Běh několika *systémů* zároveň na rozdílných vláknech.
  - *Množiny změn* – Pokud vlákna přistupují ke stejným zdrojům, je možné generovat *množiny změn*. Tím je daná operace odložena na dobu, kdy ji bude možné vykonat bez synchronizace.

Použití

- Knihovna **Entropy** je volně dostupná, pod licencí *MIT*, z níže uvedeného repositáře projektu.
- Příklad jednoduchého *systému*, který implementuje pohyb *entit*, definice *komponenty*:

```
struct PositionC
{
  float x, y;
  flot dX, dY;
}
```

- Definice *systému*:

```
struct MovementS : public System
{
  using Require = Require<PositionC>;
  void doMove();
}
```

- Samotný posun entit:

```
void doMove()
{
  for (auto &e : foreach())
  {
    PositionC *p{e.get<PositionC>()};
    p->x += p->dX;
    p->y += p->dY;
  }
}
```

Kontaktní informace

- GIT projektu: https://github.com/T0mt0mp/Entropy
- E-mail: xpolas34@stud.fit.vutbr.cz