ABSTRAK

Perkembangan industri gim dan simulasi digital menuntut kemampuan pemrosesan entitas dalam jumlah besar secara real-time, terutama pada perangkat dengan keterbatasan sumber daya. Pendekatan pengembangan konvensional berbasis *Object-Oriented Programming* (OOP) di Unity sering kali menimbulkan hambatan performa akibat struktur memori yang tidak kontigu dan ketergantungan pada eksekusi sekuensial di main thread. Masalah tersebut semakin terlihat ketika jumlah entitas meningkat, sehingga muncul kebutuhan untuk mengevaluasi paradigma alternatif yang lebih mampu menangani beban komputasi besar secara efisien. Unity menyediakan solusi melalui *Data-Oriented Technology Stack* (DOTS), sebuah pendekatan baru yang dirancang untuk memanfaatkan parallelism dan optimisasi memori.

Penelitian ini melakukan perbandingan performa antara implementasi OOP dan DOTS dengan menggunakan simulasi permainan 2D berbasis banyak entitas yang memiliki logika identik. Pengujian dilakukan dalam kondisi terkontrol menggunakan metrik CPU Frame Time dan *Frame Per Second* (FPS) untuk menilai efisiensi pemrosesan dan kestabilan runtime pada berbagai skenario jumlah entitas. Data diperoleh melalui Unity Profiler untuk memastikan pengukuran yang konsisten dan dapat direplikasi.

Hasil pengujian menunjukkan bahwa OOP mengalami peningkatan CPU Frame Time dari 20–40 ms menjadi 150–250 ms ketika jumlah entitas mencapai 200, sehingga FPS turun drastis hingga di bawah 5 FPS. Sebaliknya, DOTS mempertahankan CPU Frame Time stabil di kisaran 2–3 ms pada seluruh skenario dan menghasilkan FPS konsisten pada 300–400 FPS, memberikan peningkatan efisiensi lebih dari 50 kali lipat. Keunggulan ini berasal dari penggunaan memori kontigu, sistem pemrosesan paralel melalui C# *Job System*, dan optimisasi *Burst Compiler*.

Dengan demikian, penelitian ini menyimpulkan bahwa DOTS merupakan pendekatan yang lebih sesuai untuk aplikasi atau permainan yang membutuhkan skalabilitas tinggi dan pemrosesan entitas dalam jumlah besar. Sementara OOP tetap relevan untuk proyek berskala kecil hingga menengah, DOTS terbukti unggul dalam performa dan kestabilan pada beban komputasi tinggi.

Kata kunci: Unity, DOTS, OOP, ECS, *Job System*, *Burst Compiler*, performa, FPS, CPU *Frame Time*.

ABSTRACT

The growing complexity of digital games and interactive simulations demands real-time processing of large numbers of entities, especially on hardware with limited computational resources. Traditional Object-Oriented Programming (OOP) approaches in Unity often struggle to maintain performance due to fragmented memory layouts and reliance on sequential execution on the main thread. These limitations become increasingly apparent as the number of entities scales upward, highlighting the need for a more efficient paradigm capable of handling high computational loads. In response, Unity introduced the Data-Oriented Technology Stack (DOTS), designed to maximize memory locality, parallel processing, and scalability.

This research compares the performance of OOP and DOTS by implementing a 2D multi-entity simulation with identical logic across both paradigms. Performance evaluation was conducted under controlled conditions using CPU Frame Time and Frame Per Second (FPS) as the primary metrics to assess processing efficiency and runtime stability across varying entity counts. All measurements were collected using the Unity Profiler to ensure consistency and reproducibility.

The results indicate that the OOP implementation exhibits a sharp increase in CPU Frame Time—from 20–40 ms up to 150–250 ms at 200 entities—resulting in FPS dropping below 5 FPS. In contrast, DOTS maintains stable CPU Frame Time values of 2–3 ms across all scenarios and consistently achieves 300–400 FPS, demonstrating more than 50× improved efficiency under heavy workloads. This performance advantage is attributed to contiguous memory layout, parallel execution through the C# Job System, and low-level optimizations provided by the Burst Compiler.

Therefore, this study concludes that DOTS is a more suitable paradigm for high-performance and large-scale real-time simulations, while OOP remains appropriate for small to medium-sized projects prioritizing development simplicity. DOTS provides superior efficiency, stability, and scalability when processing large numbers of active entities.

Keywords: Unity, DOTS, OOP, ECS, *Job System*, *Burst Compiler*, *performance analysis*, FPS, CPU *Frame Time*.