

Killzone Shadow Fall: Threading the Entity Update on PS4

Jorrit Rouwé

Lead Game Tech, Guerrilla Games



GAME DEVELOPERS CONFERENCE™ EUROPE

CONGRESS-CENTRUM OST KOELNMESSE · COLOGNE, GERMANY

AUGUST 11-13, 2014 · EXPO: AUGUST 11-12, 2014



Introduction

- Killzone Shadow Fall is a First Person Shooter
- PlayStation 4 launch title
- In SP up to 60 characters @ 30 FPS
- In MP up to 24 players @ 60 FPS
- Gameplay logic has lots of
 - Branches
 - Virtual functions
 - Cache misses
- Not suitable for PS3 SPU's but PS4 has 6 x86 cores



What do we cover?

- What is an Entity?
- What we did on PS3
- Multi threading on PS4
- Balancing Entities across frames
- Performance issues found
- Performance results achieved
- Debug tools



What is an Entity?





What is an Entity?

- Base class for most game objects
 - E.g. Player, Enemy, Weapon, Door
 - Not used for static world
- Has Components
 - E.g. Model, Mover, Destructibility
- Update at a fixed frequency
 - 15, 30, 60 Hz
 - Almost everything at 15 Hz
 - Player updated at higher frequency to avoid lag



What is a Representation?

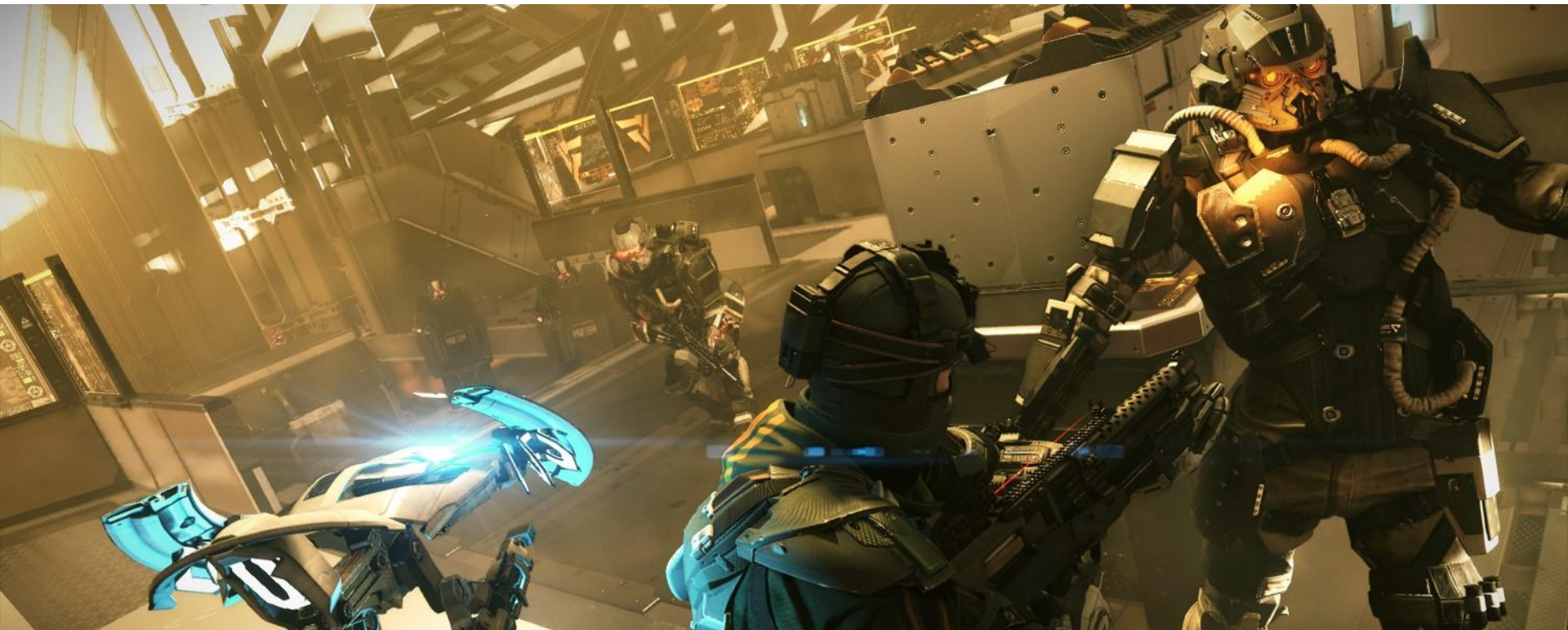
- Entities and Components have Representation
- Controls rendering, audio and VFX
- State is interpolated in frames where entity not updated
 - Cheaper to interpolate than to update
- Introduces latency
 - Always blend towards last update

A soldier in a dark, tactical suit with orange accents is running across a light-colored tiled rooftop. The soldier is holding a handgun in their right hand. White motion capture lines are overlaid on the soldier's body, showing the movement of various joints and limbs. In the background, there is a modern building with a light blue and grey facade and a dark doorway. A metal railing is visible on the left side of the rooftop. The sky is a clear, dark blue.

Movie

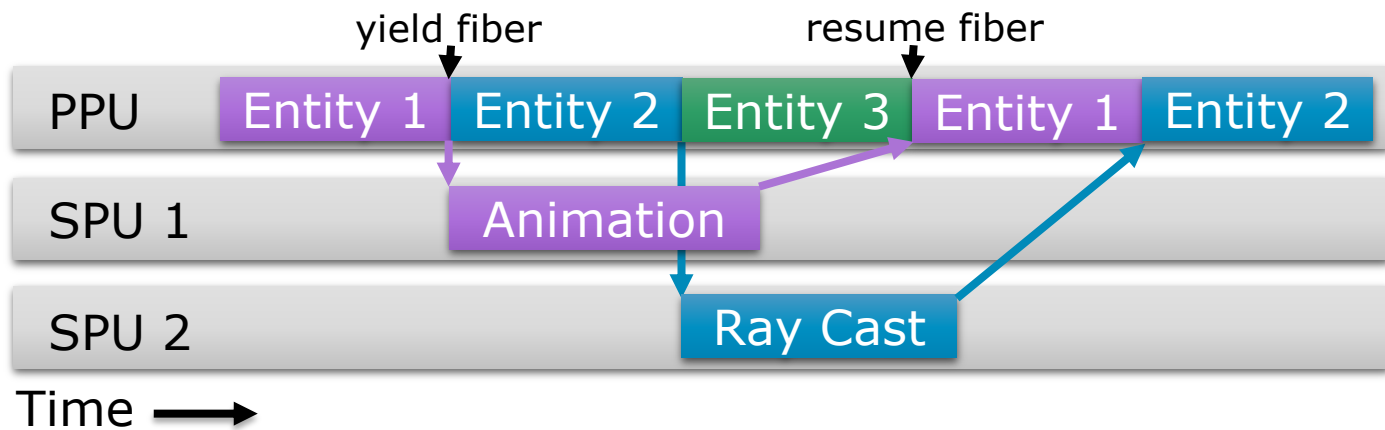


Multi Threading Approach





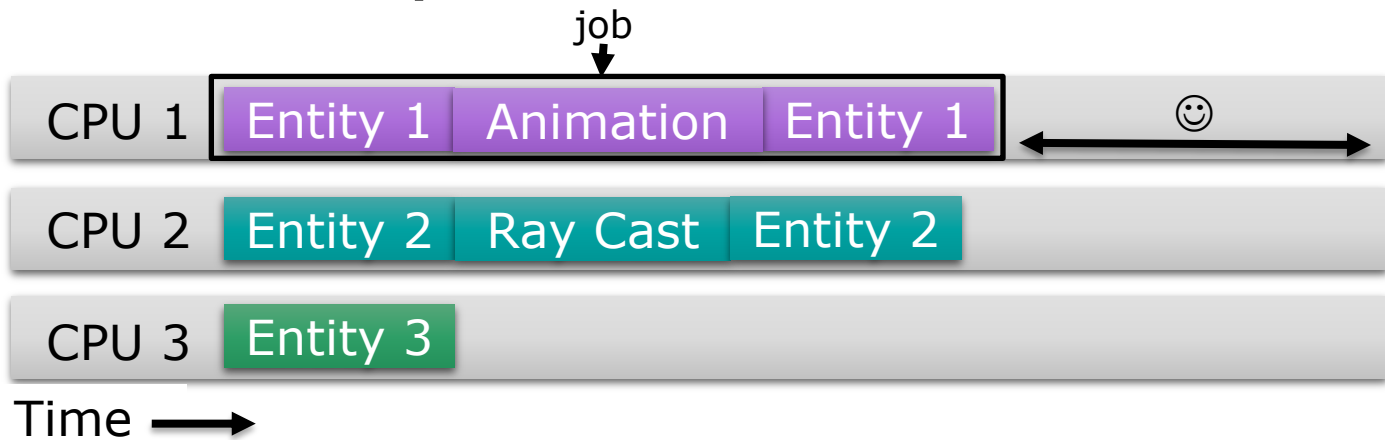
PS3: 1 Entity = 1 Fiber



- Most time spent on PPU
- No clear concurrency model
 - Read partial updated state
 - Entities deadlock waiting for each other



PS4: 1 Entity = 1 Job

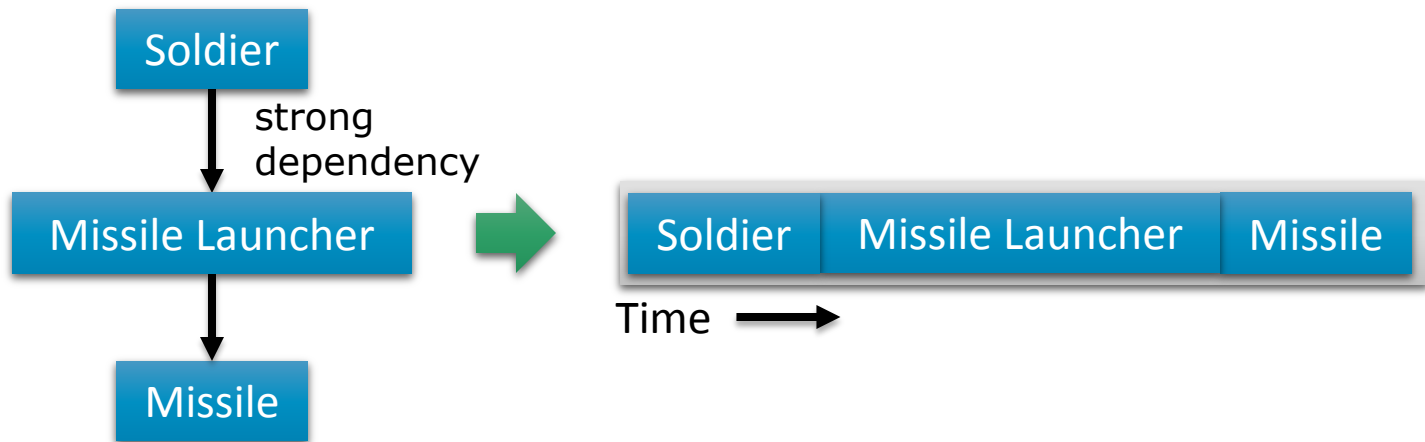


- No fibers
- Entity updates as a whole
- How to solve race conditions?



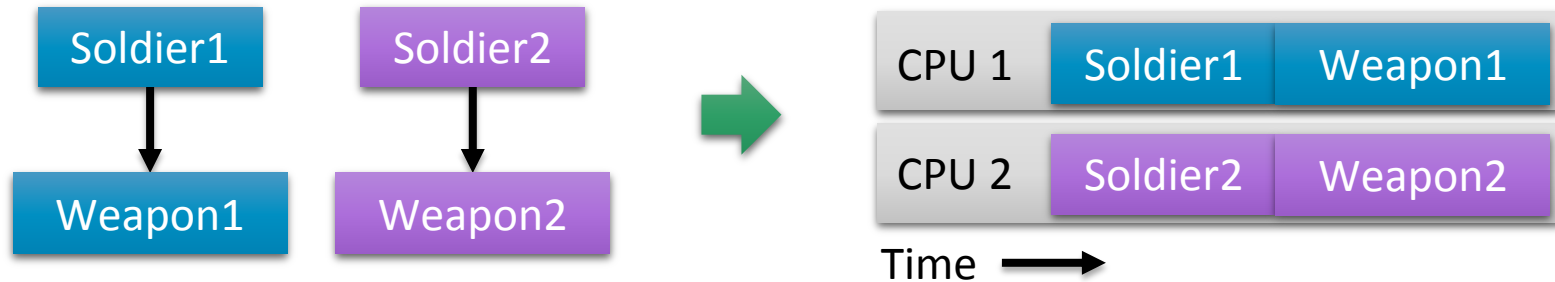
Strong Dependencies

- Make update order explicit:





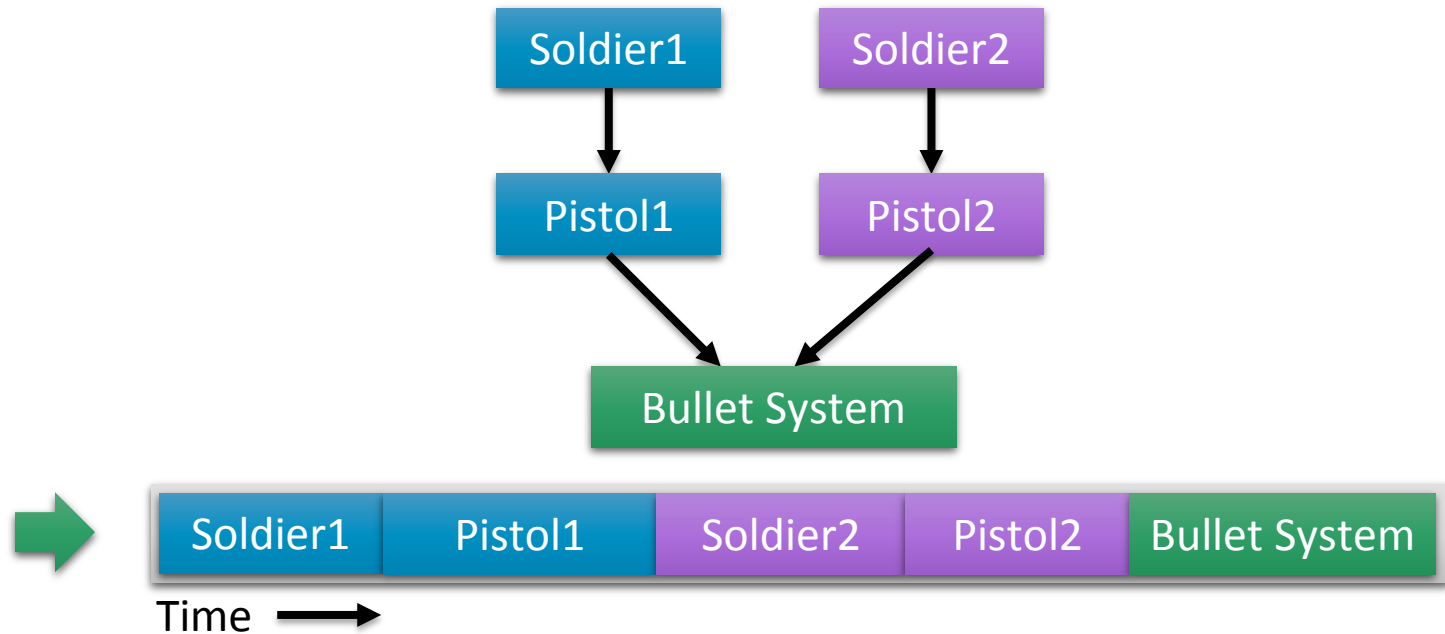
Non-dependent Entities can Execute Concurrently



- No (indirect) dependency = no access
- Works two ways: Weapon can access Soldier too
- Create dependency has 1 frame latency
- Global systems need locks



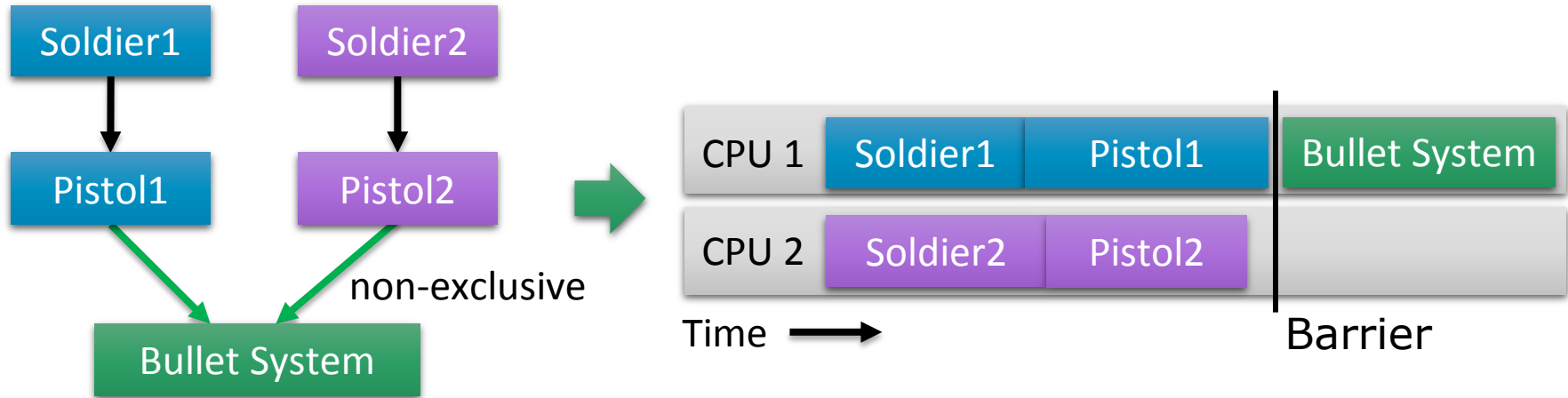
What about this?



- A few entities cause huge bottleneck



Non-exclusive Dependencies

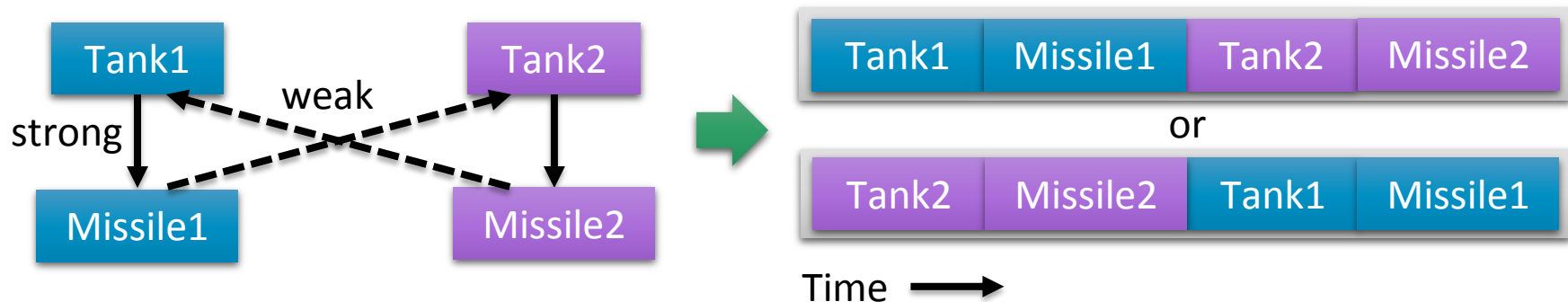


- Access to 'Bullet System' must be lock protected



Weak Dependencies

- 2 tanks fire at each other:

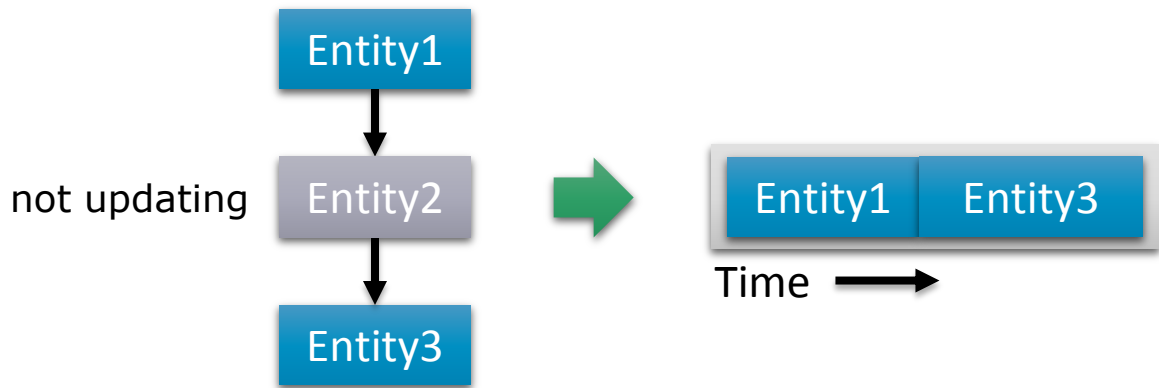


- Update order reversed when circular dependency occurs
- Not used very often (< 10 per frame)



Non-updating Entities





- Entity can skip updates (LOD)
- Entity can update in other frame



- Do normal scheduling!



Summarizing Dependencies

	Strong Exclusive	Weak Exclusive	Strong Non-excl.	Weak Non-excl.
Symbol				
Two way access	✓	✓	✓	✓
Order guaranteed	✓		✓	
Allow concurrency	+	+	++	++
Require lock			✓	✓



Referencing Entities

- Dev build: `CheckedPtr<Entity>`
 - Acts as normal pointer
 - Check dependency on dereference
- Retail build: `Entity *`
 - No overhead!
- Doesn't catch everything
 - Can use pointers to members
 - Bugs were easy to find



Working with Entities without dependency

- ThreadSafeEntityInterface
 - Mostly read only
 - Often used state (name, type, position, ...)
 - Mutex per Entity
- Send message (expensive)
 - Processed single threaded when no dependency
- Schedule single threaded callback (expensive)
 - Everything can be accessed

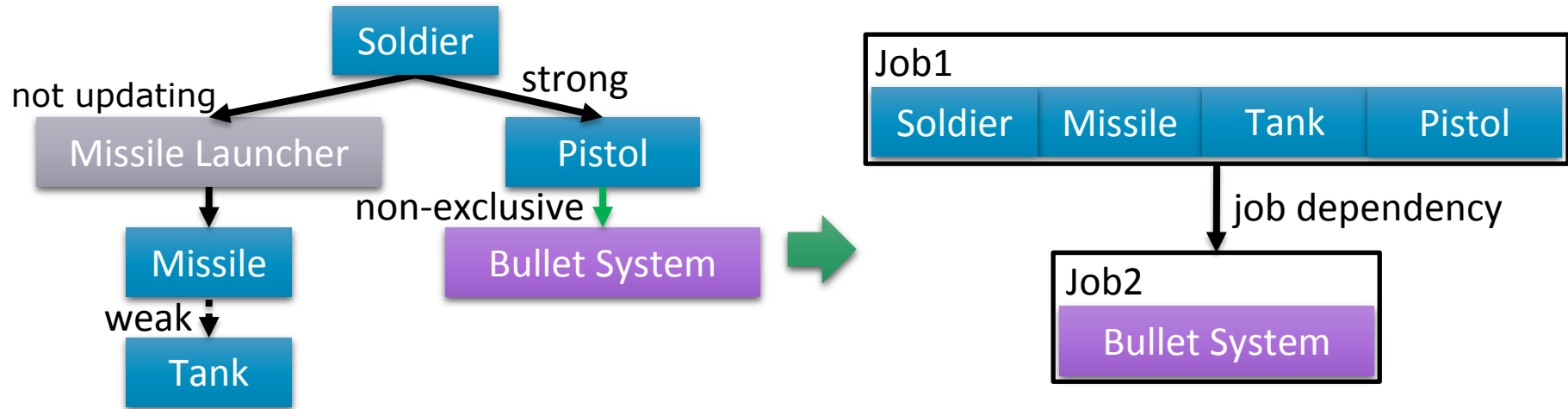


Scheduling Algorithm





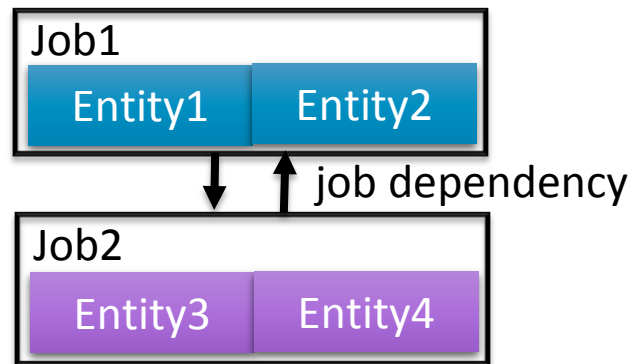
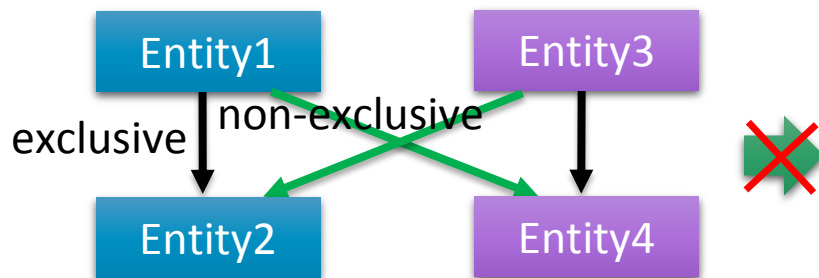
Scheduling Algorithm



- Entities with exclusive dependencies merged to 1 job
 - Dependencies determine sorting
- Non-exclusive dependencies become job dependencies
- **Expensive jobs kicked first!**

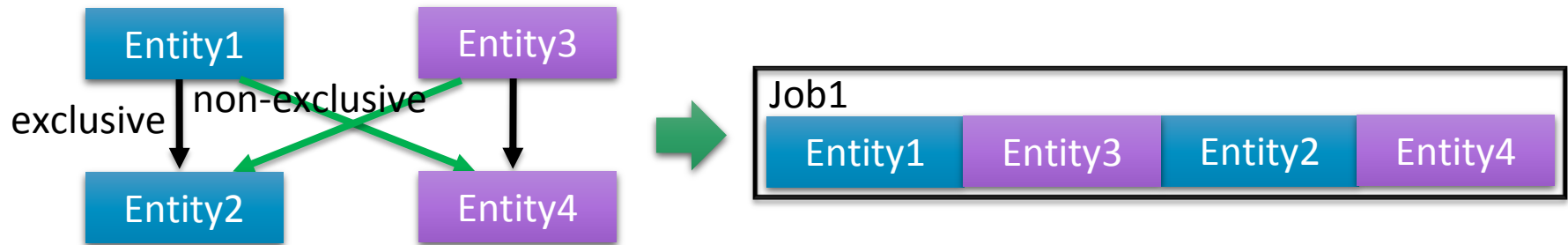


Scheduling Algorithm – Edge Case





Scheduling Algorithm – Edge Case



- Non cyclic dependency becomes cyclic job dependency
- Job1 and Job2 need to be merged



Balancing Entities Across Frames





Balancing Entities Across Frames

- Prevent all 15 Hz entities from updating in same frame
- Entity can move to other frame
 - Smaller delta time for 1 update
- Keep parent-child linked entities together
 - Weapon of soldier
 - Soldier on mounted gun
 - Locked close combat



Balancing Entities – In Action





Balancing Entities – In Action





Balancing Entities – In Action





Balancing Entities – In Action





Balancing Entities – In Action



20.73ms
20.44ms

Movie



A soldier in full tactical gear, including a helmet and body armor, is shown in a ready stance holding an assault rifle. The soldier is positioned within a blue wireframe rectangular box that appears to be a virtual environment or a tracking volume. The background consists of a light-colored tiled floor and a wall with a dark rectangular opening. The word "Movie" is written in a large, white, sans-serif font across the center of the image, partially overlapping the soldier and the wireframe box.

Movie



Performance Issues



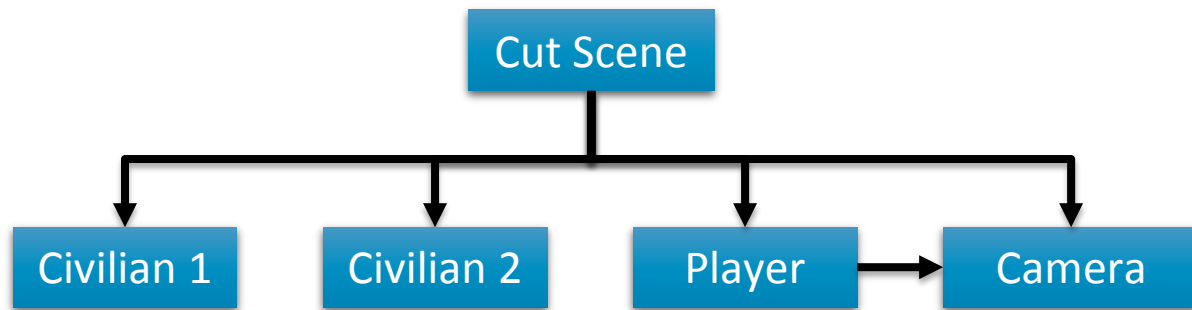


Performance Issues

- Memory allocation mutex
 - Eliminated many dynamic allocations
 - Use stack allocator
- Locking physics world
 - R/W mutex for main simulation world
 - Second 'bullet collision' broadphase + lock
- Large groups of dependent entities
- Player update very expensive



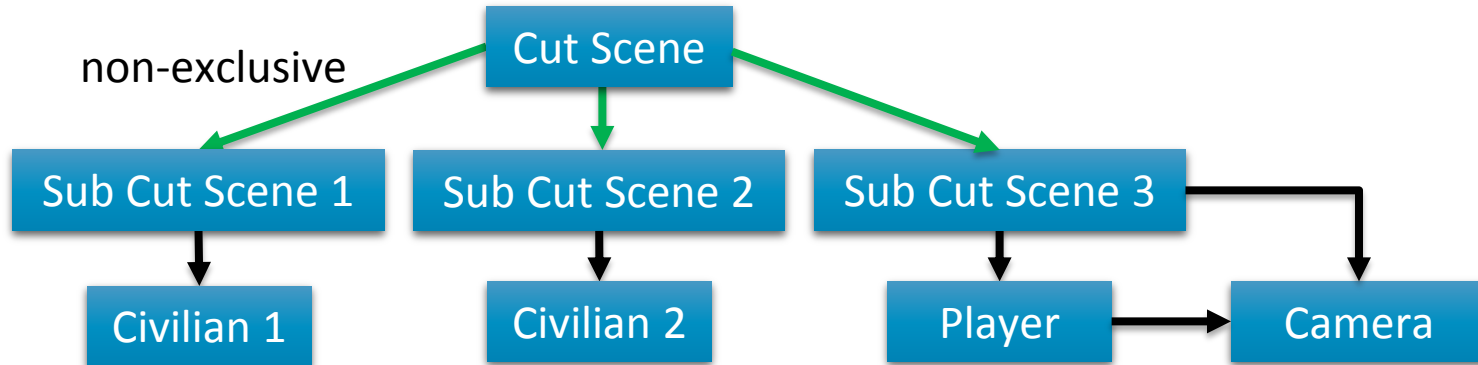
Cut Scene - Problem



- Cut scene **entity** requires dependencies
- 10+ characters in cut scene creates huge job!



Cut Scene - Solution



- Create sub cut scenes for non-interacting entities
- Master cut scene determines time and flow
- Scan 1 frame ahead in timeline to create dependency



Using an Object

- Dependencies on useable objects not possible (too many)
- Get list of usable objects
 - Global system protected by lock
- 'Use' icon appears on screen
- Player selects
 - Create dependency
 - Start 'use' animation
- Start interaction 1 frame later (dependency valid)
- Hides 1 frame delay!

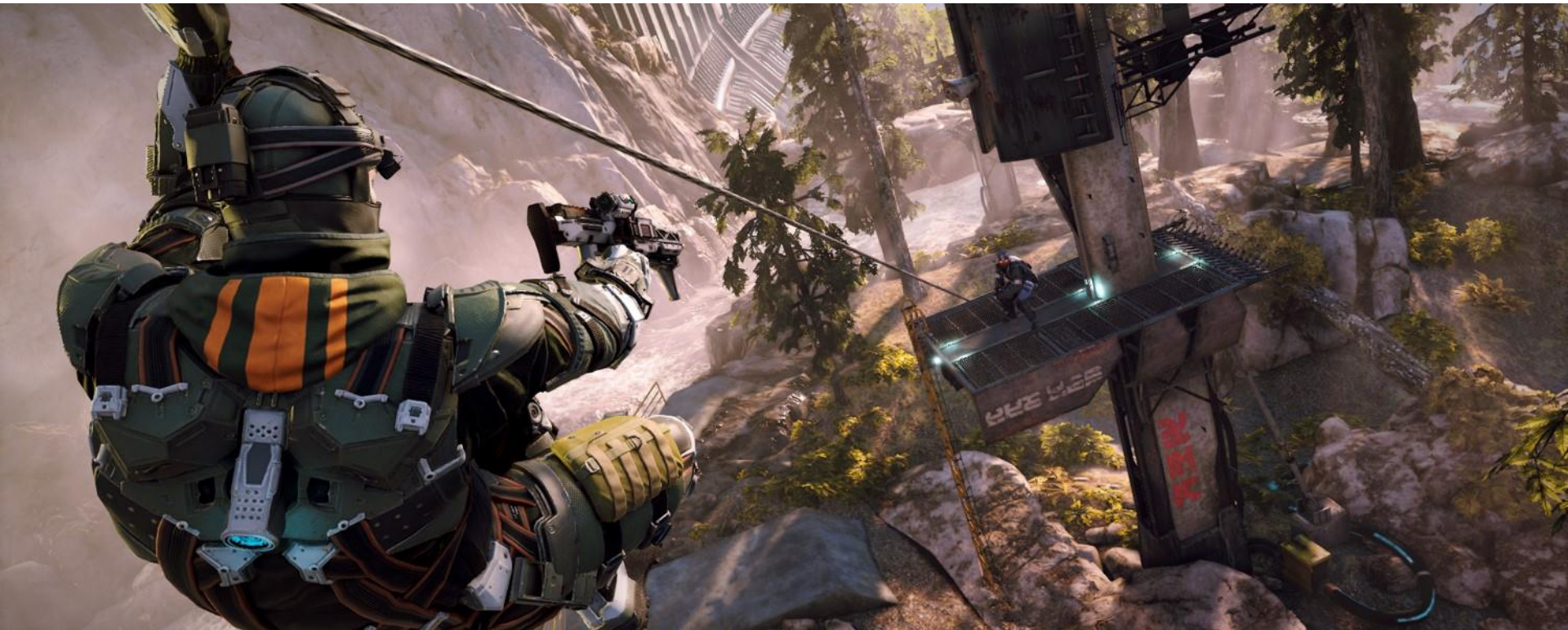


Grenade

- Explosion damages many entities
- Creating dependencies not option (too many)
- `ThreadSafeEntityInterface` not an option
 - Need knowledge of parts
- Run line of sight checks inside update
- Uses scheduled callback to apply damage



Performance Results





Performance Results - Synthetic

Level	Counts			Dependencies		Max Entities in Job	Speedup
	Number Entities	Updating Entities	Number Humans	Strong Excl	Strong Non-Excl		
5000 Crates (20 μ s each)	5019	5008	1	12	4	13	2.8X
100 Soldiers (700 μ s each)	326	214	105	212	204	19	4.2X
500 Flags (160 μ s each)	519	508	1	12	4	13	5.2X

6 cores!



5000 Crates



100 Soldiers



500 Flags



Performance Results - Levels

Level	Counts			Dependencies		Max Entities in Job	Speedup
	Number Entities	Updating Entities	Number Humans	Strong Excl	Strong Non-Excl		
The Helghast (You Owe Me)	1141	206	32	71	23	20	4.1X
The Patriot (On Vectan Soil)	435	257	44	199	107	15	4.3X
The Remains (12p Botzone)	450	128	14	97	44	18	3.7X



The Helghast



The Patriot



The Remains

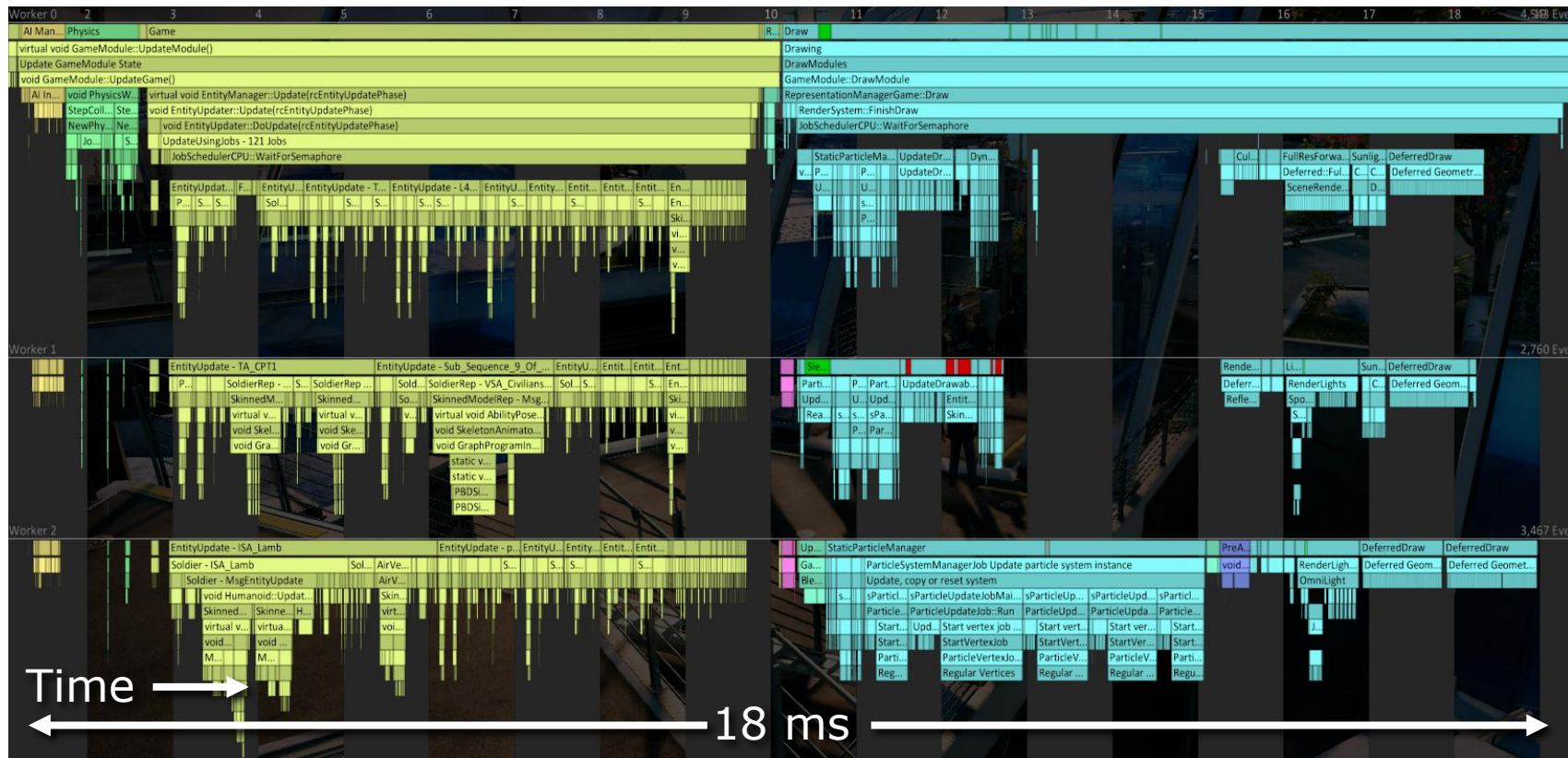


Game Frame - The Patriot

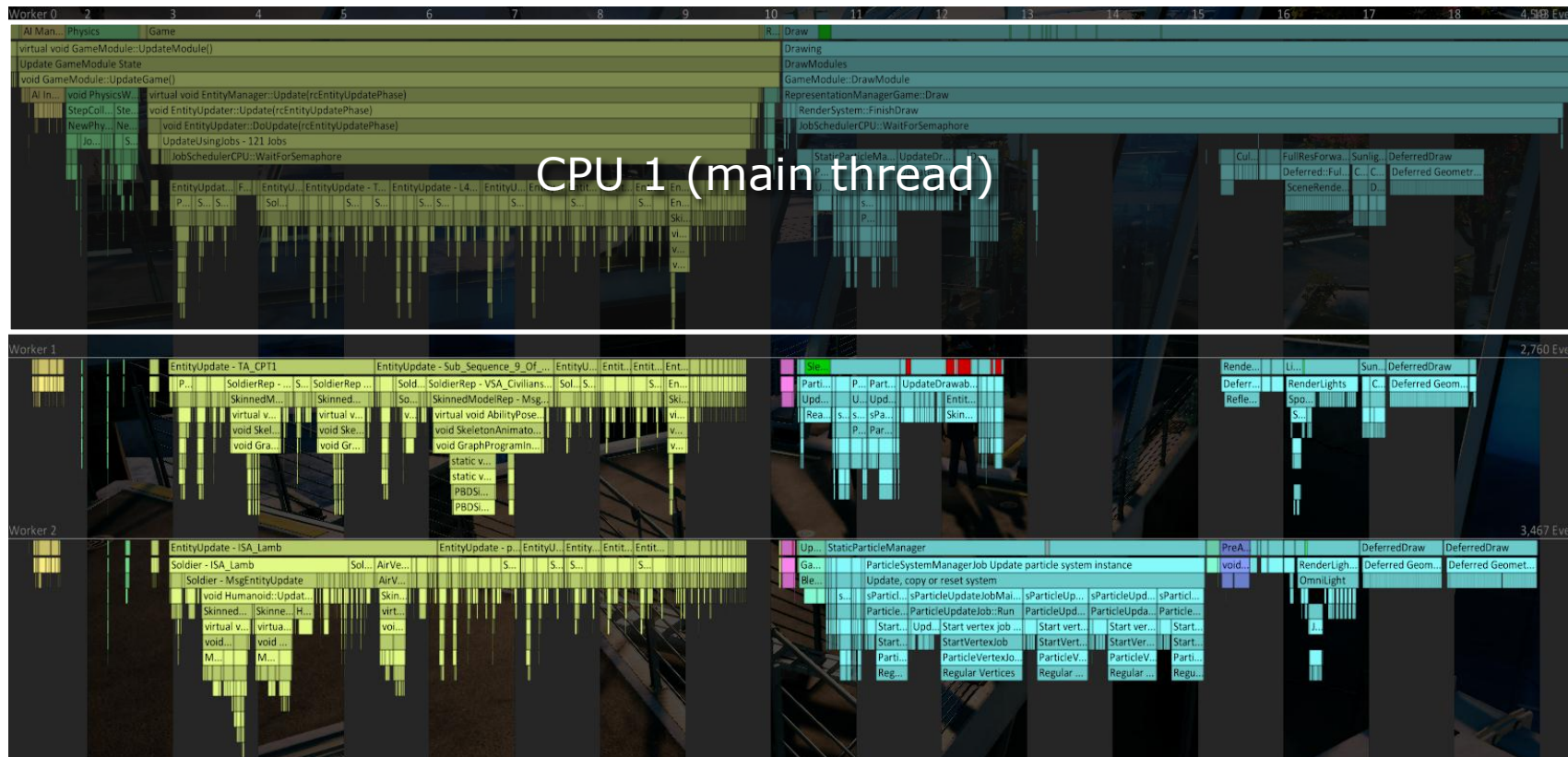




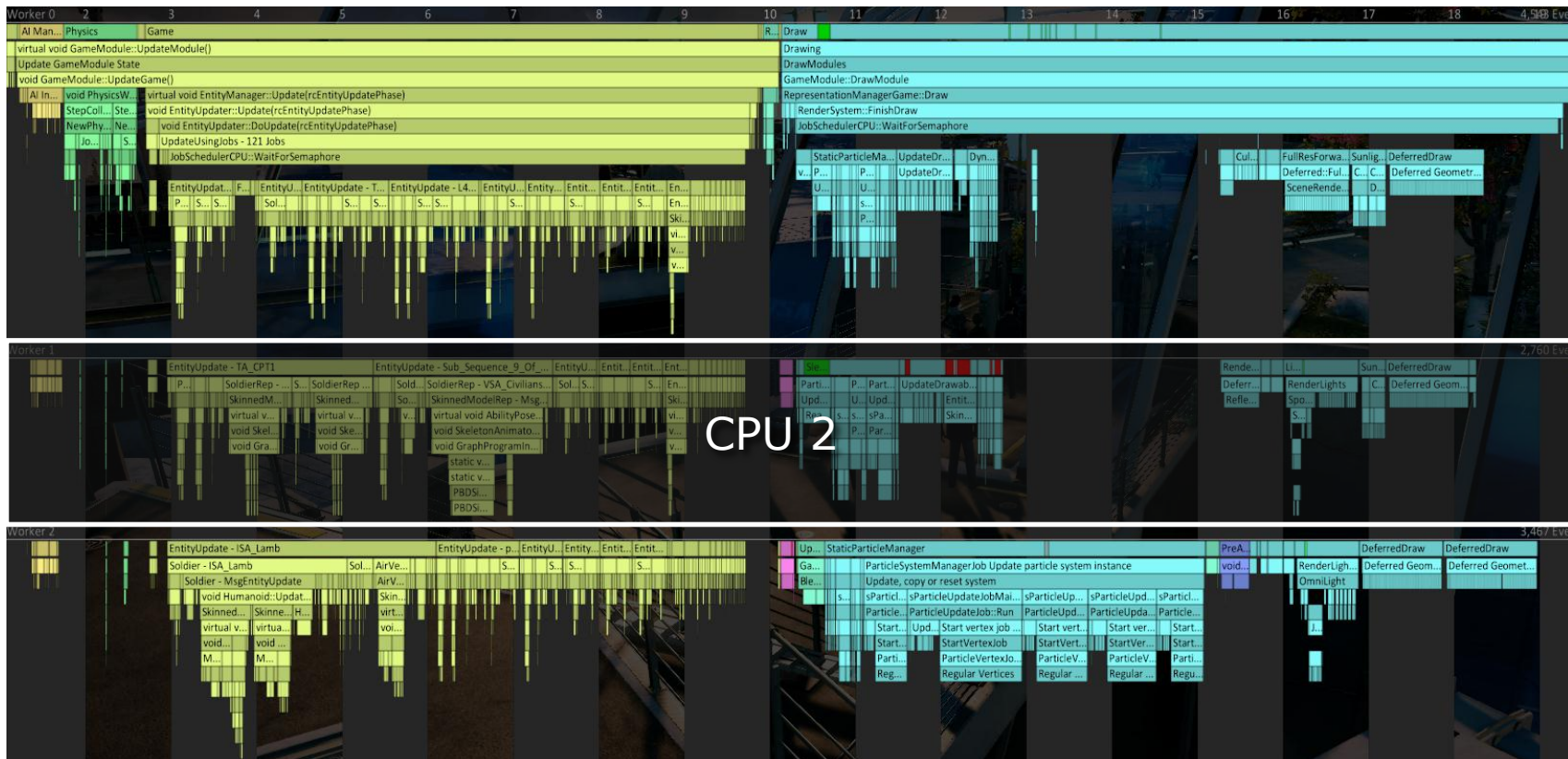
Game Frame - Global



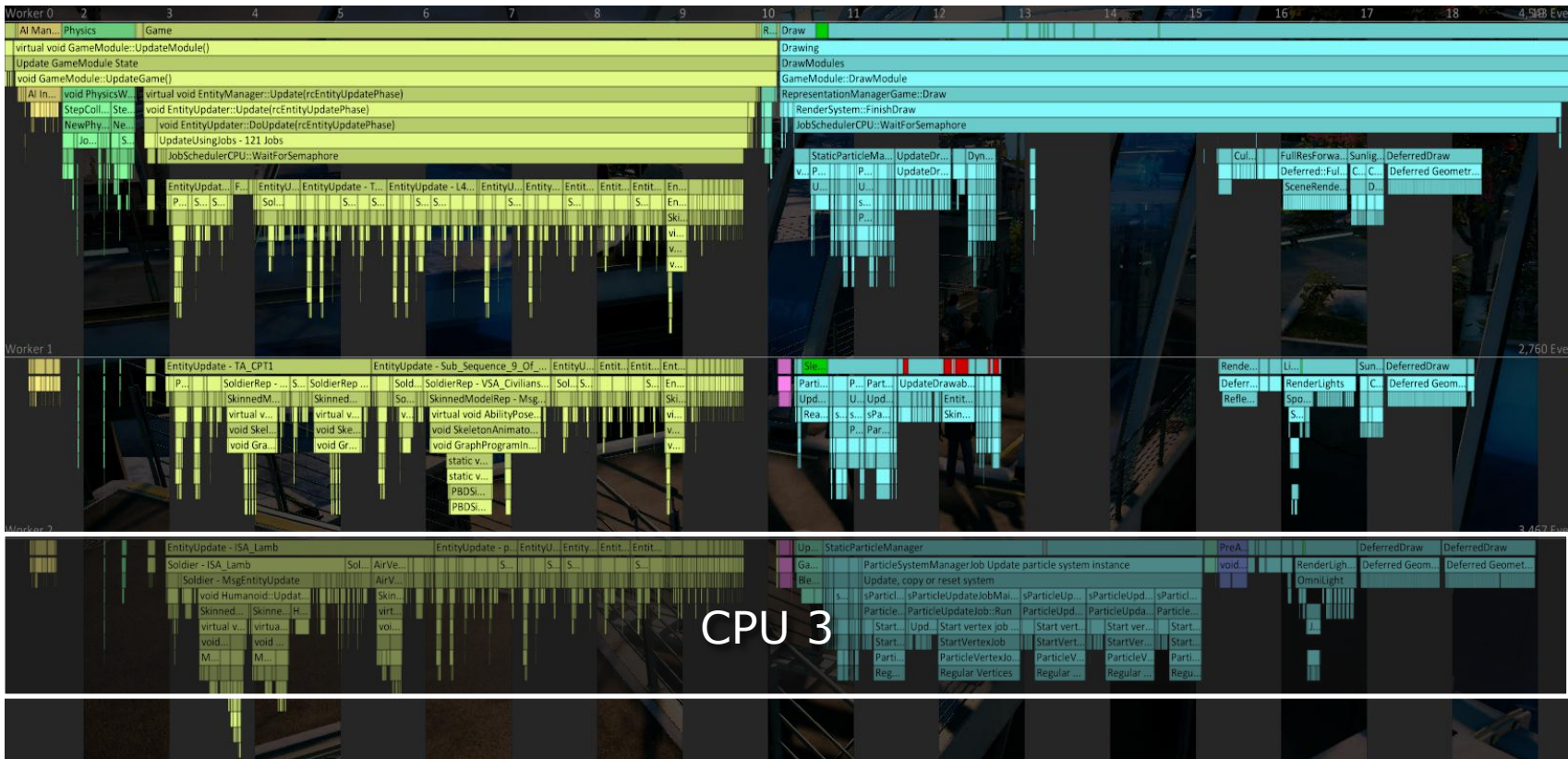
Game Frame - Global



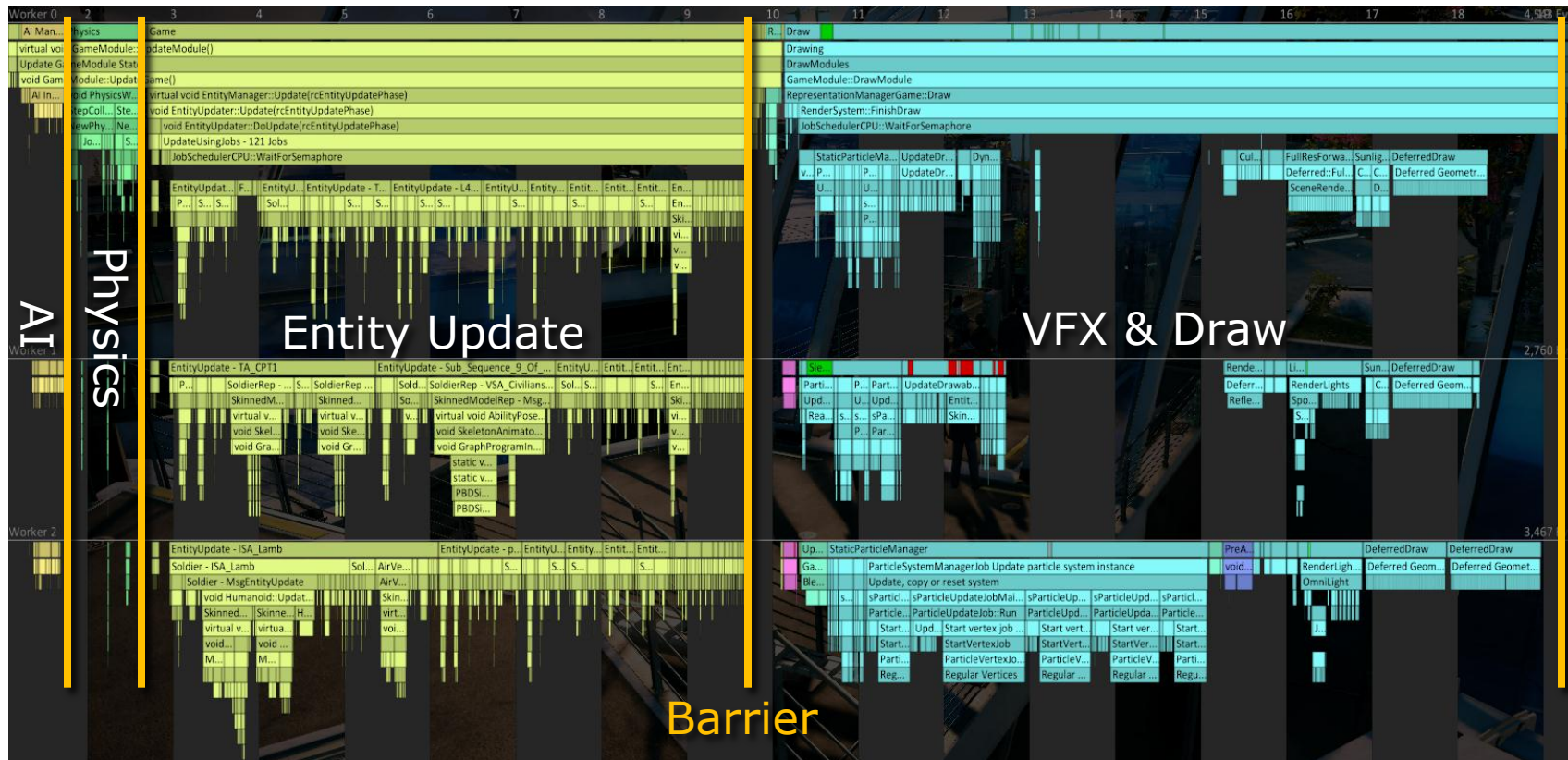
Game Frame - Global



Game Frame - Global



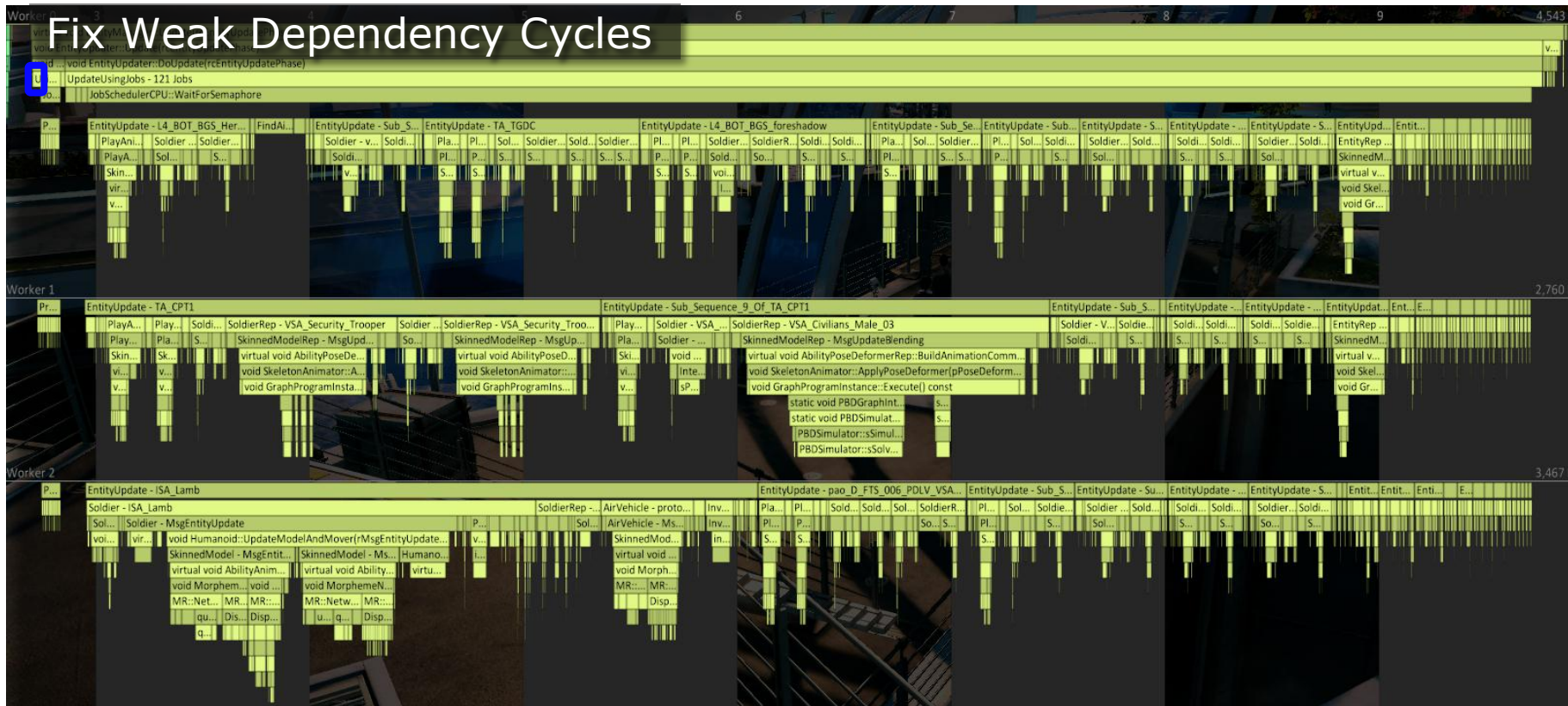
Game Frame - Global





Game Frame – Entity Update

Fix Weak Dependency Cycles





Prepare Jobs



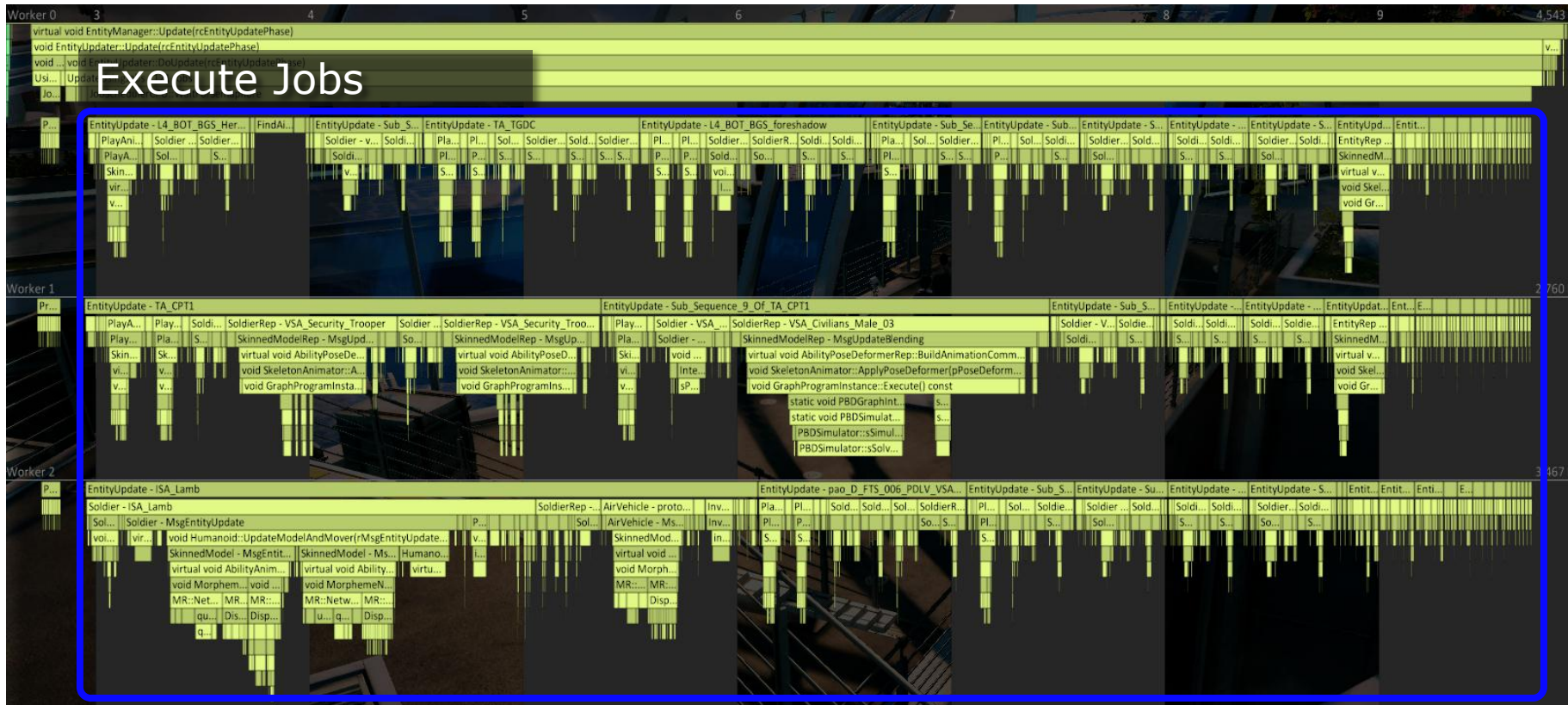


Link and Order Jobs





Execute Jobs



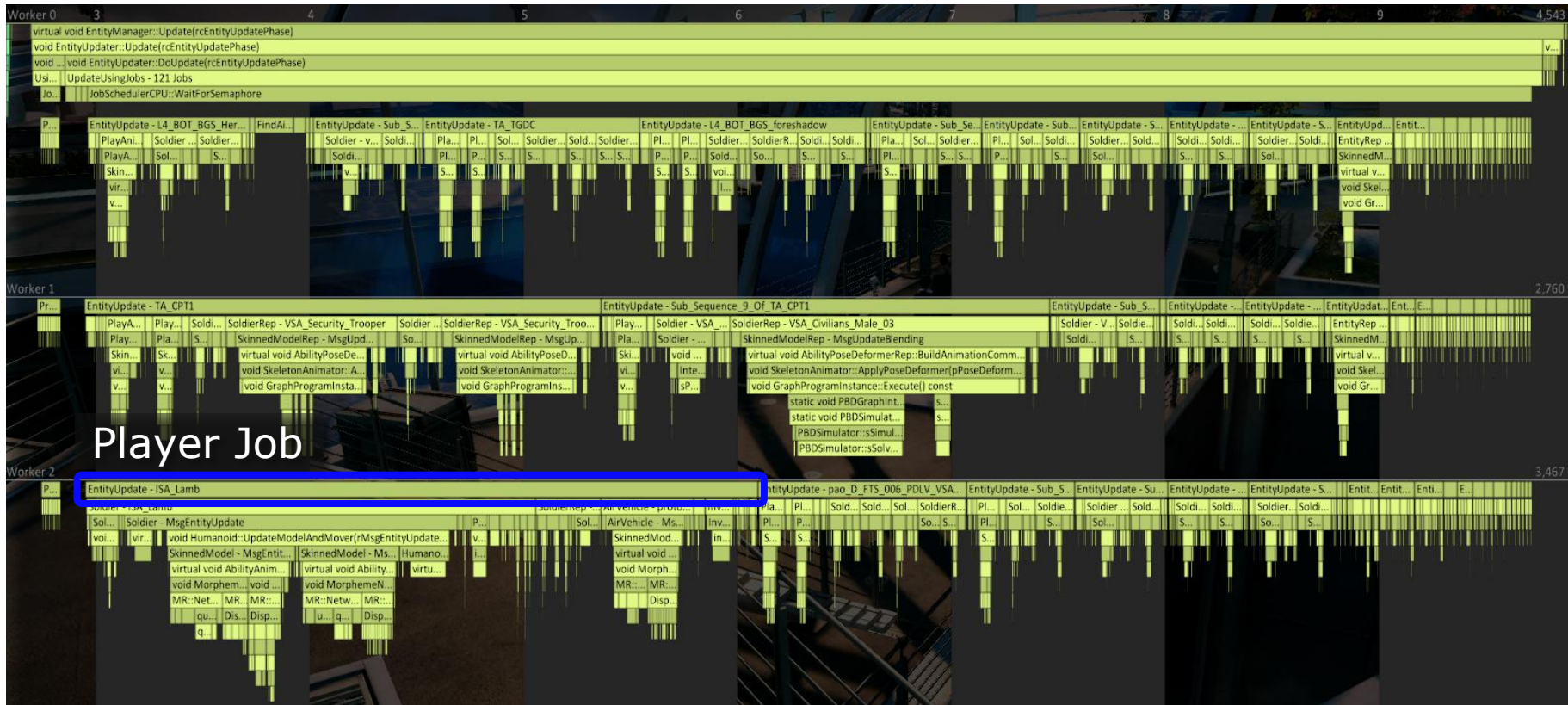


Single Threaded Callbacks



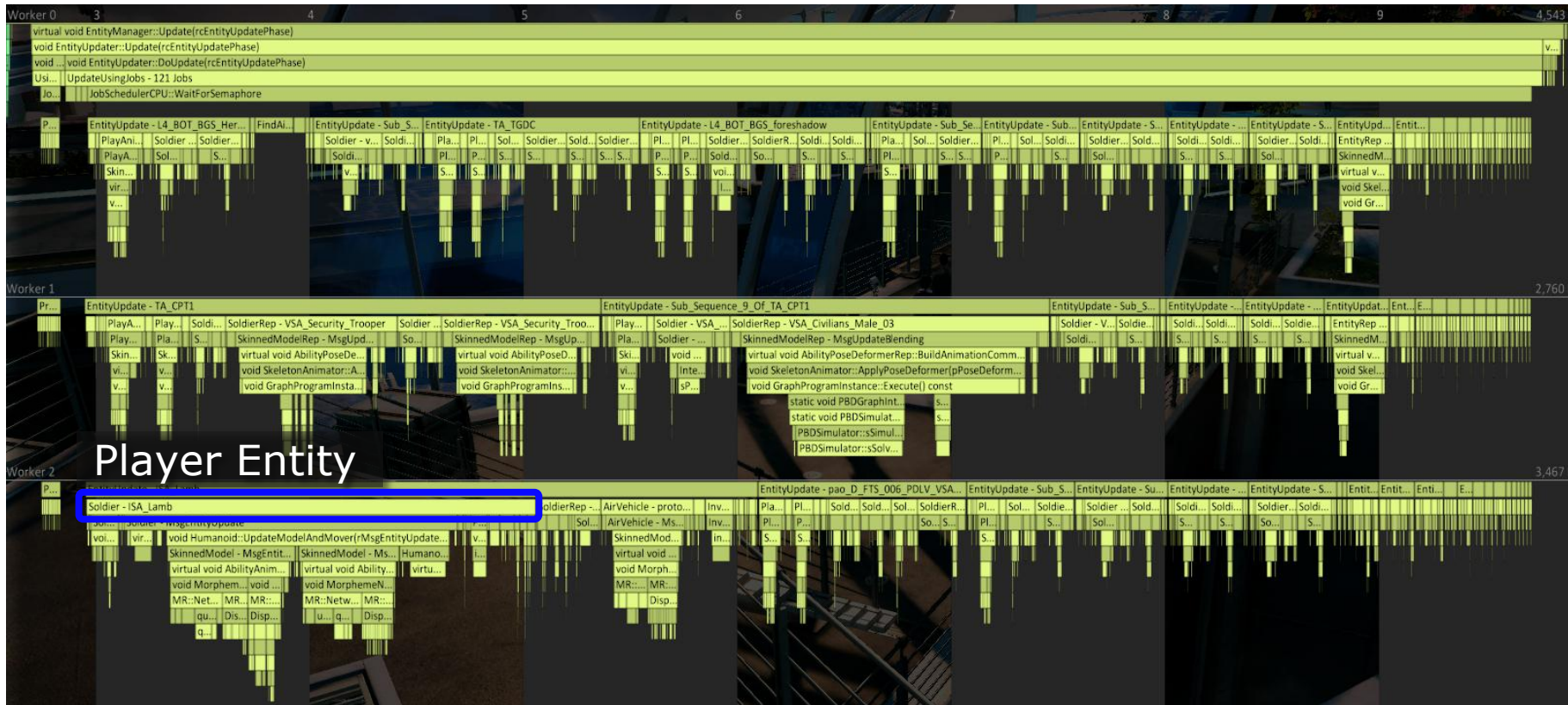


Game Frame – Entity Update





Game Frame – Entity Update





The screenshot displays the Unity Profiler's Animation timeline. The timeline is organized into sections for Worker 0, Worker 1, and Worker 2. The 'Animation' section is highlighted in blue. The timeline shows various animation events, including 'SkinnedModel - MsgEntit...', 'SkinnedModel - Ms...', 'void Morphem...', 'MR-Net...', 'qu...', and 'Disp...'. The timeline is marked with time in seconds, ranging from 0.000 to 4.543.



The screenshot displays the Unity Profiler's 'EntityUpdate' phase, divided into two worker threads: Worker 0 and Worker 2. Worker 0's timeline shows a total duration of 4.543 seconds, with sub-phases like 'EntityUpdate - L4_BOT_BGS_Her...' and 'EntityUpdate - Sub_S...'. Worker 2's timeline shows a total duration of 3.467 seconds, with sub-phases like 'EntityUpdate - ISA_Lamb' and 'EntityUpdate - pao_D_FTS_006_PDLV_VSA...'. A red box highlights the 'Humanoid' component in the 'EntityUpdate - ISA_Lamb' sub-phase. The background shows a 3D scene of a building interior.



Game Frame – Entity Update



[illegible]

[illegible]

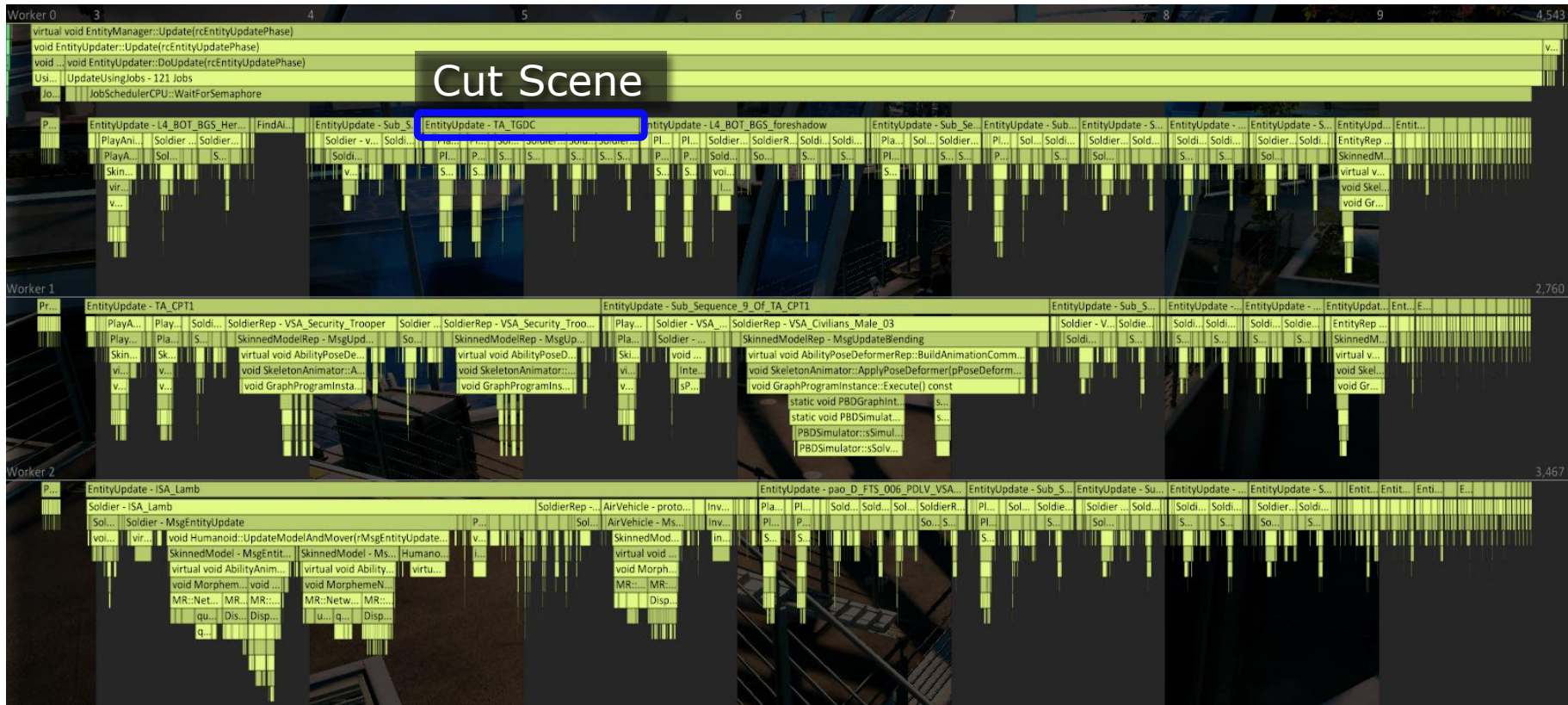
[illegible]



The screenshot displays the Unity Hierarchy view for a scene. The top of the hierarchy shows the 'Worker 0' and 'Worker 1' entities. Below them, various 'EntityUpdate' components are listed, such as 'EntityUpdate - L4_BOT_BGS_Her...', 'EntityUpdate - Sub_Sequence', and 'EntityUpdate - ISA_Lamb'. The 'EntityUpdate - Sub_Sequence' entity is highlighted, and its 'Cloth Simulation' component is selected. The 'Cloth Simulation' component is highlighted with a blue box. The 'Cloth Simulation' component is highlighted with a blue box. The 'Cloth Simulation' component is highlighted with a blue box.

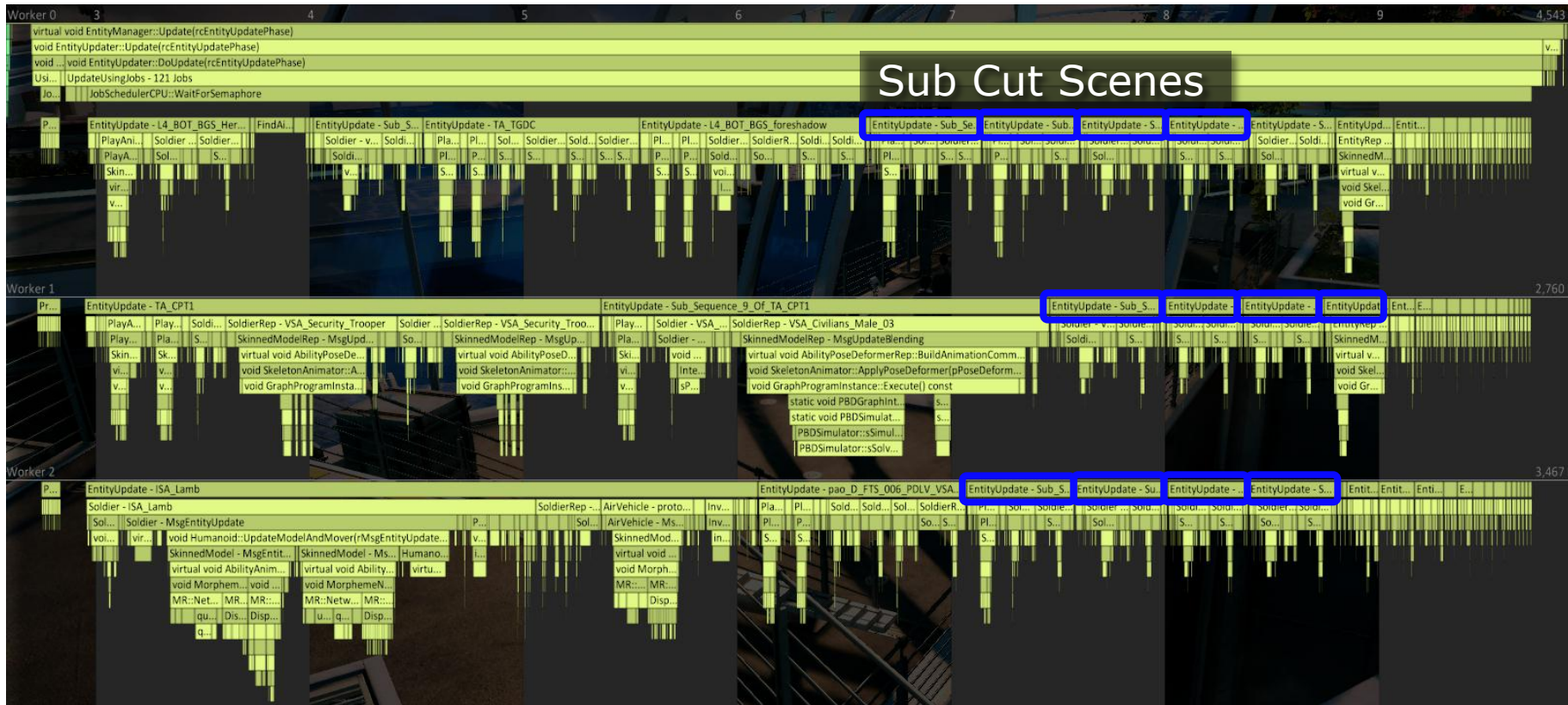


Cut Scene





Game Frame – Entity Update





Cheap Entities (Destructibles)



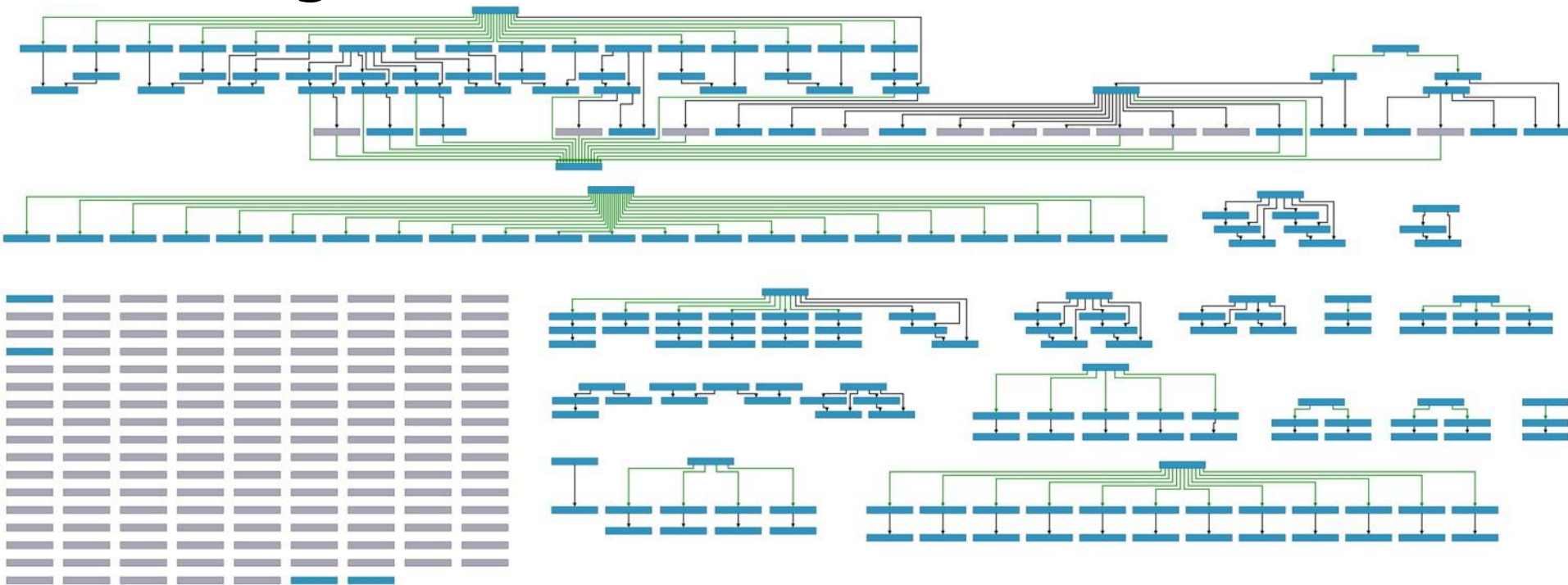


Debug Tools





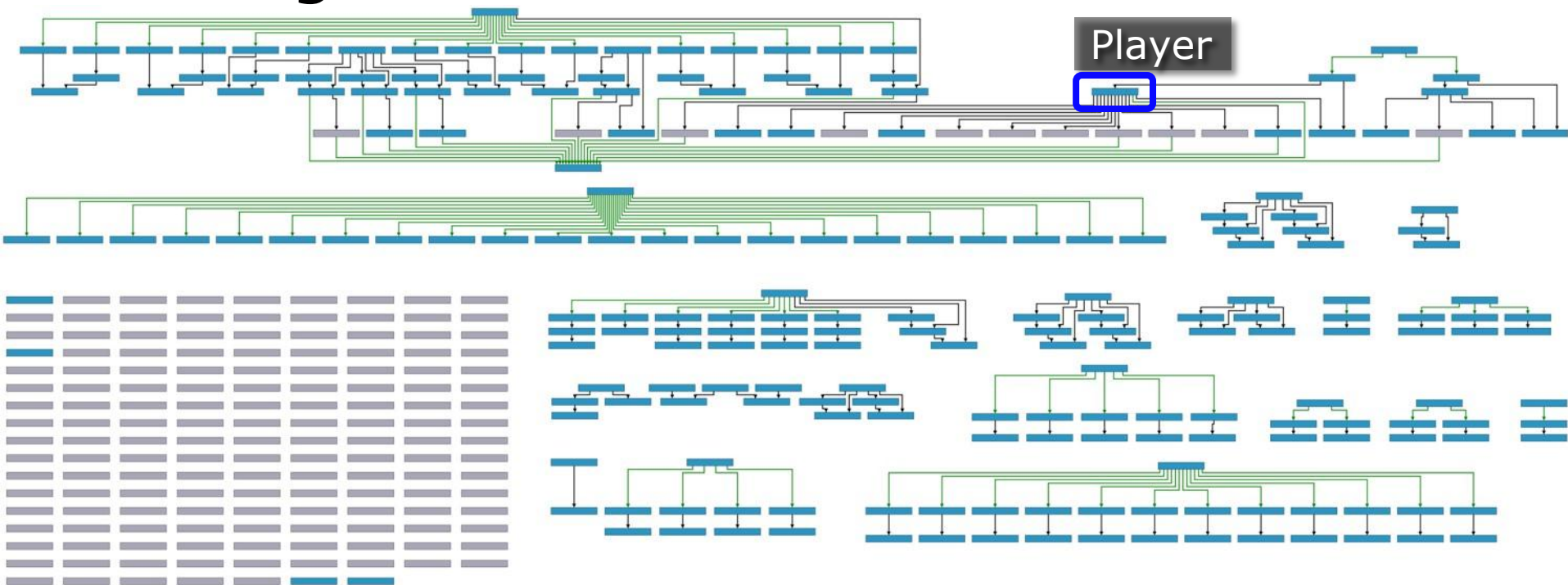
Debug Tools



- Dependencies get complex, we use yEd to visualize!

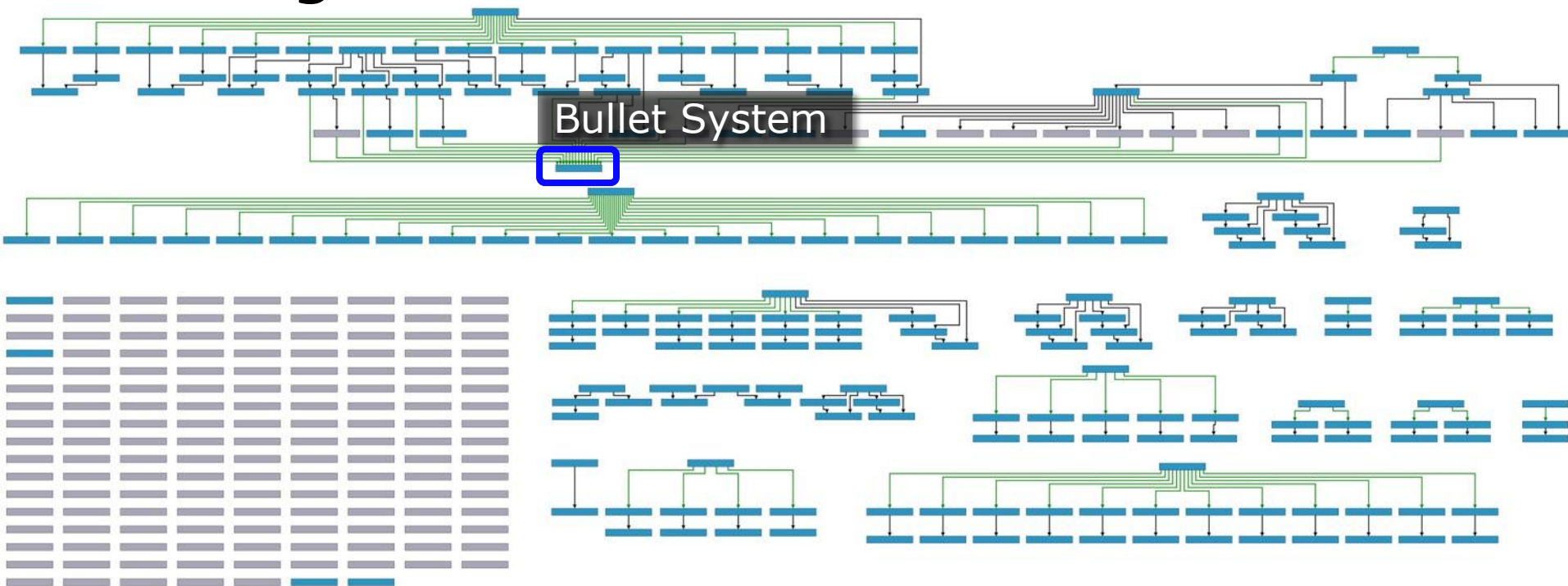


Debug Tools





Debug Tools





Debug Tools

Cut scenes





Conclusions

- Easy to implement in existing engine
- Game programmers can program as if single threaded
- Very few multithreading issues



Questions?

jorrit@guerrilla-games.com