International Scientific Conference 20 UNITECH 26

INTERNATIONAL SCIENTIFIC CONFERENCE
19-21 November 2026
GABROVO

# ПРАКТИЧЕСКИ ПОДХОД ЗА СЪЗДАВАНЕ И ИНТЕГРИРАНЕ НА MAP LOADING OBJECT (MLO) В ИГРАТА GTA V ЧРЕЗ FIVEM

**Денис Коев[1], Христо Килифарев[2], Виктория Иванова[3]**

*[1,2,3]Технически университет - Габрово, ул. Хаджи Димитър 4, Габрово, България*
**кореспондиращ автор: [1]koev59@abv.bg, [2]hkilifarev@tugab.bg, [3]v.ivanova@tugab.bg*

# PRACTICAL APPROACH FOR CREATING AND INTEGRATING A MAP LOADING OBJECT (MLO) IN GTA V WITH FIVEM

**Denis Koev[1], Hristo Kilifarev[2], Viktoria Ivanova[3]**

*[1,2,3]Technical University of Gabrovo, 4 Hadzhi Dimitar St., Gabrovo, Bulgaria*
**Corresponding author: [1]koev59@abv.bg, [2]hkilifarev@tugab.bg, [3]v.ivanova@tugab.bg*

**Abstract**
*Creating a custom interior requires consistent geometry, resource metadata and deployment settings. This paper presents a documentation-based workflow for Grand Theft Auto V (GTA V), an open-world video game, using FiveM, a multiplayer modification platform. An MLO is treated as an interior archetype whose rooms, portals and entities describe the interior structure. The workflow combines Blender, a 3D modeling application; Sollumz, a Blender plugin for GTA V assets; and CodeWalker, a map and asset inspection and editing tool. Eight stages connect spatial planning, modeling, material preparation, collision geometry, interior definition, export, deployment and testing. The methodological results are an ordered sequence of deliverables, a division of responsibilities between tools, and three acceptance levels covering resource loading, interior functionality and runtime behavior. These criteria distinguish a visible static asset from an interior with verified collisions and portal visibility. The contribution is a synthesis of documented operations into a reviewable procedure.*

**Keywords:** GTA V Modding; FiveM Platform; Map Loading Object (MLO); Blender; Sollumz; CodeWalker.

## ВЪВЕДЕНИЕ

FiveM е клиент-сървър платформа за управление на ресурси, разширяваща Grand Theft Auto V (GTA V) чрез интегриране на картови обекти (**Map Loading Object, MLO**). Динамичното зареждане на активи (assets) в отделни области позволява внедряване на сгради и интериори без модификация на базовите файлове на играта [1, 2]. Структурната единица за разпространение е ресурсът, дефиниран чрез fxmanifest.lua и дистрибутиран от FXServer към клиентите [3].

Готовите MLO рядко съответстват на изискванията на ролевите (**roleplay**) сървъри. Собствената разработка осигурява контрол над архитектурата, маршрутите, помещенията и интеракциите, но изисква съчетаване на 3D моделиране, специфични файлови формати и сървърно администриране.

Картовият обект е интериорен архетип, описващ пространствена система чрез помещения, портали, ограничителни обеми и обекти. Помещенията определят вътрешните обеми, а порталите – логическите връзки и видимостта между тях [4].

Целта на доклада е да представи последователна и повторима методика за създаване и интегриране на интериорни обекти като картови ресурси във FiveM среда.


*Print ISSN: 1313-230X, Online ISSN: 3033-1404*

## ИЗЛОЖЕНИЕ

### Използвани средства и ресурсни формати

За валидиране на методиката са използвани следните програмни средства:

- Blender (v5.1.2) – за 3D моделиране и геометрично структуриране на обектите;
- **Sollumz (v2.8.3)** – плъгин за експорт на собствените за GTA V формати, материали, текстури и Drawable йерархии [4, 5];
- **CodeWalker** – за верификация, дефиниране на MLO архетипи (Archetypes), задаване на граници (Bounds) и позициониране в картата [6];;
- **OpenIV (v4.1)** – за генериране и управление на текстурни речници (.ytd);
- **WinSCP** – FTP/SFTP клиент за трансфер на MLO файловете към хостинг сървъра;
- **FXServer** – сървърна архитектура за поддръжка на платформата FiveM [7, 8];
- **txAdmin** – административен интерфейс за мониторинг, управление на сървъра и внедряване на ресурси [7, 8].

Работният пакет включва следните взаимно свързани формати, структуриращи ресурса:

- **YDR (Drawable)** – визуална 3D геометрия;
- **YTD (Texture Dictionary)** – текстурен речник, съхраняващ графичните карти;
- **YBN (Bounds)** – геометрични граници за изчисляване на колизии;
- **YTYP (Archetype Definition)** – метаданни за архетипа и специфичната MLO вътрешна структура;
- **YMAP (Map Data)** – определя екземпляра (инстанцията), координатите и ориентацията в игровия свят.

Конфигурационният файл fxmanifest.lua регистрира и инициализира ресурса във FiveM. Файловете от тип YMAP не заместват MLO структурата, тъй като само позиционират дефиниран архетип, без да генерира вътрешни пространства (стаи) и преходни зони (портали) [2, 4].

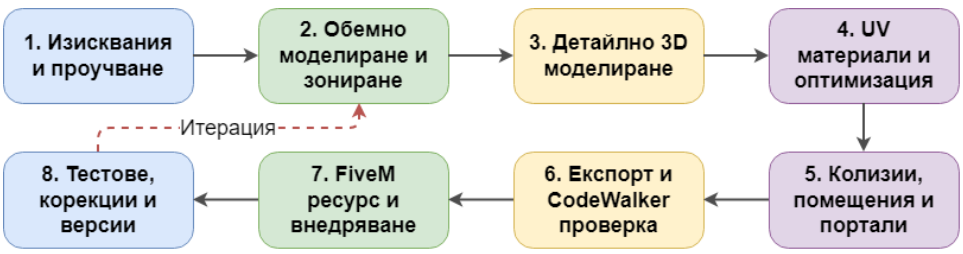
### Итеративна методика

Методиката включва осем взаимно свързани фази (фиг. 1). Първоначално се формулират функционалното предназначение, архитектурните ограничения, пространствената позиция и критериите за приемане. На следващата стъпка се извършва грубо обемно моделиране (blockout) чрез примитиви за проверка на мащаба, етажите, функционалните зони и маршрутите. След приемане на базовата схема се разработват детайлният модел, UV координатите, материалите и оптимизираните игрови варианти.

Моделите за сблъсък (колизии), помещенията и преходните портали се изграждат след стабилизиране на геометрията, тъй като всяка последваща модификация на стените и отворите налага повторно конфигуриране. Експортираните активи се проверяват в CodeWalker, капсулират се като FiveM ресурс и се тестват в сървърна среда. При дефект итеративният характер на процеса позволява връщане към конкретния засегнат етап без рестартиране на целия цикъл, осигурявайки проследимост на версиите и локализиране на грешки.

### Проектиране и моделиране

Разработването започва с дефиниране на функционалното предназначение и зонирането. Определят се входовете, ета-




***Фиг. 1.*** *Итеративен жизнен цикъл за създаване и интегриране на MLO*

жите, маршрутите, функционалните области и очакваният брой потребители. Функционалните зони не трябва автоматично да стават MLO помещения (Rooms). Самостоятелно помещение се дефинира при реален затворен или частично затворен обем. Това намалява изчислителната сложност и риска от неправилно изключване (*culling*) на геометрията от енджина.

Оригиналната геометрия се съхранява в защитена колекция, а редакциите се извършват върху работно копие. Мащабът се верифицира чрез референтен игрови персонаж, врати, стъпала и проходи. Метричната система в Blender не гарантира реалистични пропорции в играта. След утвърждаване на размерите се прилагат трансформациите за ротация и мащаб (**Apply Rotation & Scale**), а координатното начало (**Origin**) се позиционира в основата [9].

Топологичната проверка включва премахване на дублирани върхове (*vertices*), нулева площ, вътрешни повърхнини (*faces*) и незатворени шевове. Обект на инспекция са нормалите (*normals*), не-манифолдните (**Non-Manifold**) участъци и триангулацията. Не е задължително структурата да бъде единна полигонална мрежа. Препоръчително е разделянето ѝ по групи: архитектура, покрив, терен, растителност, интериор и статични обекти (props). Единният Drawable обект улеснява първоначалния тест, но увеличава консумацията на памет и затруднява оптимизацията.

**Материали, текстури и оптимизация**

Текстурните (UV) координати се подготвят с контролирана плътност (**Texel Density**) за съпоставима разделителна способност на повърхностите [10]. Стандартните Blender материали се преобразуват в специфични за GTA V шейдъри чрез Sollumz. Дифузната текстура се свързва към параметъра DiffuseSampler, като името ѝ точно съответства на записа в текстурния речник (YTD). Размерът и форматът се определят според видимата площ, работната дистанция и графичния бюджет. Блоковата компресия (**Block Compression**) и нивата на детайлност (**mipmaps**) ограничават натоварването на видеоконтролера и оптимизират динамичното зареждане (*streaming*) във видеопаметта [11].

Оптимизацията се базира на количествени измервания преди и след модификациите. На постоянен мониторинг подлежат броят на полигоните (триъгълниците), използваните материали, обемът и разделителната способност на текстурите, както и общият размер на ресурсния пакет.

Скритите повърхнини и микродетайли, невидими от стандартна игрова дистанция, се премахват. Основният силует и базовите архитектурни форми се запазват, а финият релеф и текстилните гънки се пренасят върху двуизмерна карта на нормалите (*normal map*). За структурите, видими от големи разстояния, задължително се генерират нива на детайлност (**Level of Detail, LOD**), като плавният преход се верифицира чрез симулация в игровата среда [12].

**Колизии, помещения и портали**

Физическият колизионен модел се генерира независимо от визуалната геометрия. Подовете, стените, стълбищата и парапетите се изграждат чрез примитиви или опростени полигонални мрежи. Например стълбищните клетки се конфигурират с наклонена равнина вместо с индивидуални структури за всяко стъпало, което гарантира плавна навигация. Елементите на микроинтериора и малките орнаменти изключват собствена колизия, ако не влияят върху кинематиката на движение. Чрез Sollumz колизионната структура се подготвя като ограничителен обем (**Bounds**) и се експортира във формат YBN [13].

Разпределението на помещенията (Rooms) следва реалните архитектурни обеми. Външното пространство (Limbo), входовете, стълбищните преходи и изолираните интериорни пространства се дефинират програмно на базата на физическите прегради. Всеки портал (Portal) се

позиционира в архитектурен отвор (врата, прозорец) и свързва точно две топологични области. На задължителна проверка подлежат вертексите (ъгловите точки), нормалните вектори и двустранната видимост. Неправилна топологична връзка предизвиква артефакти при изключването на видимостта (**culling artifacts**) и изчезване на геометрията при преминаване. След всяка промяна в 3D модела се преизчисляват пространствените граници на визуалните активи (Drawable), ограничителните обеми (Bounds), архетипа и помещенията [4].

**Експорт и изграждане на FiveM ресурса**

Преди експортиране се изолира чист визуализационен коренов обект (**Drawable Root**) и се проверяват приложените трансформации, нормалните вектори, активната UV карта, цветните атрибути и GTA V материалите. Чрез Sollumz се генерира изходният YDR файл. В CodeWalker се дефинира YTYP архетипът, чието системно наименование, име на актива (*asset name*) и текстурен речник (*texture dictionary*) трябва точно да съответстват на обекта. След това чрез YMAP се определя местоположението на екземпляра (инстанцията) чрез координати, триосна ориентация и мащабен коефициент, като задължително се изпълняват процедурите **Calculate All Flags** и **Calculate Extents** [6].

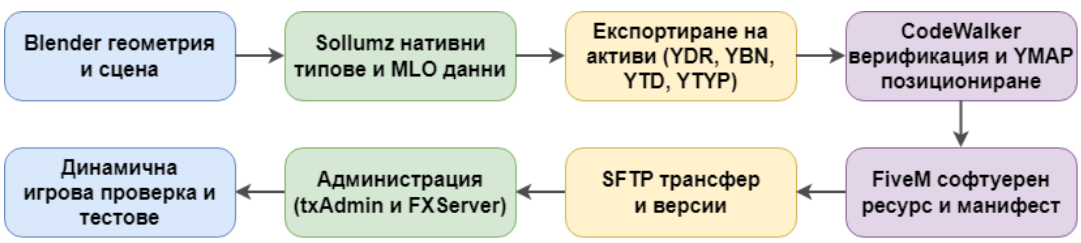
Готовият ресурс съдържа конфигурационен файл fxmanifest.lua и директория stream с файловите структури YDR, YTD, YBN, YTYP и YMAP. Манифестът дефинира версията (fx_version 'cerulean'), съвместимостта (game 'gta5'), индикира картова модификация (this_is_a_map 'yes') и софтуерно регистрира YTYP метаданните чрез директивата DLC_ITYP_REQUEST [3]. Пакетът се разполага на сървъра под пътя resources/[mlo] чрез SFTP. В интерфейса txAdmin се изпълняват командите refresh и ensure с името на ресурса, а за автоматично иницииране съответната директива се вгражда в server.cfg [7, 8].

**Инструментален поток**

На фиг. 2 е представено разграничението между етапите на методиката и програмните средства. Blender съхранява изходната сцена и базовата геометрия. Разширението Sollumz отговаря за присвояването на специфичните за GTA V native типове, шейдъри, ограничителни обеми (Bounds) и MLO метаданни за експортиране. CodeWalker верифицира архетипите, пространственото YMAP позициониране, обхватите (*extents*) и флаговете. Финалният FiveM ресурс капсулира активите в директория stream и ги дефинира чрез fxmanifest.lua. Чрез SFTP клиент се осъществява мрежов трансфер на версията, а FXServer и txAdmin я активират за игрова проверка.

Разделянето на отговорностите има висока диагностична стойност. Дефекти в CodeWalker обикновено произхождат от геометрията, материалите, дефиницията на архетипа или експортирането. Обратно, аномалии след сървърното внедряване насочват към несъответствия в ресурса, синтактични грешки в манифеста, непълен трансфер, софтуерен конфликт или проблеми с клиентския кеш. Благодарение на тази диагностична матрица корекциите се насочват таргетирано към компрометирания етап, без цялостно преизграждане на пакета.

**Валидиране и критерии за приемане**




***Фиг. 2.*** *Обобщен инструментален поток от Blender до проверката във FiveM*

Верификацията на дигиталния актив се извършва на три строго дефинирани нива:

- **Първо ниво (Инфраструктурна съвместимост)** – потвърждава индексирането, инициализирането и рендирането в енджина без конзолни грешки. Това доказва валидността на ресурсната верига, но не гарантира пълна MLO функционалност;
- **Второ ниво (Вътрешна функционалност)** – проверява материалите, физическите модели (колизиите), проходимостта на маршрутите, помещенията и двустранната портална видимост;
- **Трето ниво (Системна интеграция)** – оценява прехода между нивата на детайлност (LOD), натоварването, съвместимостта с други картови ресурси и мрежовото поведение при синхронизация между клиенти.

Функционалното изпитване включва приближаване на персонажа от различни посоки, преминаване през входовете, обход на помещенията, движение по стълби, прагове и повторно стартиране на ресурса. Визуалният тест обхваща оценка на нормалите (ден/нощ), сканиране за липсващи текстури, детекция на артефакти от припокриване на равнини (**z-fighting**) и некоректно изключване (*culling*) на геометрията. Производителността се сравнява по идентичен маршрут и при еднакви графични настройки, като се измерват заделената памет, кадровата честота и наличието на микро-накъсвания (*micro-stuttering*) [12].

Активът е завършен MLO ресурс единствено ако визуалната геометрия, текстурният речник (YTD), колизионният модел (YBN), вътрешната архитектура (YTYP стаи и портали), пространственото позициониране (YMAP) и сървърният пакет са налични и изпитани. Наличието на видим Drawable обект без верифицирани колизии и MLO метаданни се третира само като статичен модел или ранен геометричен прототип. Това методологическо разграничение предотвратява необосновани твърдения и позволява резултатът да бъде оценен обективно на базата на реално доказани свойства.

**Документиране и възпроизводимост**

Управлението на конфигурацията и проследимостта на ресурса налагат за всяка итерационна версия да се архивират следните артефакти: изходният .blend проект, междинните XML метаданни, конфигурациите за експортиране, проектът в CodeWalker, финалният сървърен ресурс, списъкът с референции и тестовият протокол. Идентификаторите на файловите структури се подчиняват на конвенция за именуване – изписване само с малки латински букви и без интервали, което гарантира *cross-platform* съвместимост на файловата система при внедряване в Linux и Windows среди.

Преди всяка замяна или надграждане стабилната версия се архивира, а в регистъра на модификациите (**Change Log**) се документират промяната, идентифицираният дефект и резултатът от повторното валидиране. Въведената архитектурна организация гарантира възпроизводимост на експеримента, позволява незабавно възстановяване на последното стабилно състояние (*roll-back*) и прави инженерния процес напълно проверим и одитируем от външни разработчици и рецензенти.

## ЗАКЛЮЧЕНИЕ

Практическият подход обхваща последователно системното проучване, грубото моделиране (blockout), детайлното 3D моделиране, подготовката на материалите, графичната оптимизация, дефинирането на колизии, изграждането на MLO архитектура, експортирането, позиционирането, сървърното внедряване и изпитването. Моделите на жизнения цикъл и инструменталния поток (фиг. 1 и фиг. 2) разграничават логическите етапи от програмните средства и детерминират фазите за обратно връщане при дефекти.

Предимството на методиката се състои в декомпозирането на разработката до

проверими междинни резултати. Статичният актив се третира като междинна фаза, а статутът на завършена MLO структура се присъжда само след доказване на коректността на текстурите, колизиите, помещенията, порталната видимост и поведението в игрова среда. Подходът притежава висока инвариантност, приложим е към различни модели и може да се разширява с интерактивни функции чрез QBCore, ESX или самостоятелни скриптове, без да се нарушава независимостта на картовото съдържание.

**БЛАГОДАРНОСТИ:** *Настоящият документ е изготвен с финансовата помощ на договор № НИП2026-6 за провеждане на научни изследвания по проект на тема: „Изследване и изграждане на система за откриване на аномалии в компютърни мрежи” към Технически университет – Габрово.*

## ЛИТЕРАТУРА

[1] Gregory, J. *Game Engine Architecture*. 3rd ed. Boca Raton, FL: CRC Press, 2018.

[2] Cfx.re. **Placing Assets and Creating Map Mods**. Cfx.re Documentation, 2026. URL: https://docs.fivem.net/docs/assets-manual/beginner-series/part-4/ (accessed 27 Aug. 2026).

[3] Cfx.re. **Resource Manifest**. Cfx.re Documentation, 2026. URL: https://docs.fivem.net/docs/scripting-reference/resource-manifest/ (accessed 27 Aug. 2026).

[4] Sollumz Project. **Creating YTYP**. Sollumz Documentation, 2026. URL: https://docs.sollumz.org/2.6/tutorials/creating-interiors/create-ytyp (accessed 27 Aug. 2026).

[5] Sollumz Project. **Sollumz Documentation**. Sollumz Documentation, 2026. URL: https://docs.sollumz.org/ (accessed 27 Aug. 2026).

[6] dexyfex. **CodeWalker - GTA V Map and Asset Explorer**. GitHub Repository, 2026. URL: https://github.com/dexyfex/CodeWalker (accessed 27 Aug. 2026).

[7] Cfx.re. **Server Commands**. Cfx.re Documentation, 2026. URL: https://docs.fivem.net/docs/server-manual/server-commands/ (accessed 27 Aug. 2026).

[8] Cfx.re. **txAdmin Documentation**. Cfx.re Documentation, 2026. URL: https://docs.fivem.net/docs/resources/txAdmin/ (accessed 27 Aug. 2026).

[9] Blender Foundation. **Blender Manual**. Blender Documentation, 2026. URL: https://docs.blender.org/manual/en/latest/ (accessed 27 Aug. 2026).

[10] Blender Foundation. **Editing UVs**. Blender Documentation, 2026. URL: https://docs.blender.org/manual/en/latest/modeling/meshes/uv/editing.html (accessed 27 Aug. 2026).

[11] Microsoft. **Texture Block Compression in Direct3D 11**. Microsoft Learn, 2026. URL: https://learn.microsoft.com/en-us/windows/win32/direct3d11/texture-block-compression-in-direct3d-11 (accessed 27 Aug. 2026).

[12] Akenine-Möller, T., E. Haines, and N. Hoffman. *Real-Time Rendering*. 4th ed. Boca Raton, FL: CRC Press, 2018.

[13] Sollumz Project. **Creating Static Collisions**. Sollumz Documentation, 2026. URL: https://docs.sollumz.org/2.6/tutorials/creating-static-collisions (accessed 27 Aug. 2026)