Politecnico di Torino

Corso di Laurea Magistrale
In Architettura Costruzione Città
A.a. 2021/2022
Sessione di Laurea Dicembre 2022

Serious Game per simulazioni di evacuazione antincendio

Applicativo per l'evacuazione di studenti in una scuola primaria e secondaria

Relatore:
Prof.ssa Anna Osello

Candidato:
Elena Cloverio

Correlatore:
Ing. Francesca Maria Ugliotti

ABSTRACT

Per sopravvivere a un incendio, gli occupanti devono poter evacuare l'edificio prima che la situazione diventi insostenibile. Il tempo di evacuazione è quindi un fattore critico e le normative antincendio disponibili raccomandano di ridurlo attraverso un'adeguata progettazione della struttura e degli ausili all'evacuazione. Oltre alle decisioni ingegneristiche prese nella progettazione di un edificio, il comportamento degli occupanti è un fattore importante che influisce in modo significativo sui tempi di evacuazione. La scarsa conoscenza che le persone hanno sullo sviluppo degli incendi e sulle dinamiche di un'emergenza incendio non le prepara ad avere la migliore risposta durante gli incendi. Pertanto, conoscere il corretto comportamento di sicurezza antincendio è una strategia di protezione personale che consentirebbe a ciascun occupante di aumentare significativamente le proprie possibilità di sopravvivenza riducendo i tempi di evacuazione e prevenendo i comuni errori che gli occupanti commettono.

In questo lavoro di tesi viene proposto lo sviluppo di un applicativo per la simulazione di evacuazione antincendio per gli studenti di una scuola primaria e secondaria di I grado. Vengono proposti Serious Games e Virtual Reality come strumento per acquisire competenze personali in materia di sicurezza antincendio. Nel primo capitolo vengono approfonditi i temi inerenti i Serious Games, la Virtual Reality e il BIM, applicati in seguito al caso studio. Nel secondo capitolo vengono fornite le normative della prevenzione incendi, in particolare applicate alle scuole, fondamentali per lo studio dell'esodo. Nel terzo capitolo viene descritto come Serious Games e Virtual reality possano essere applicati alla prevenzione incendi, fornendo la metodologia utilizzata per lo sviluppo dell'applicativo. Nel quarto capitolo viene descritto l'applicativo che è stato sviluppato per l'esodo degli studenti all'interno del plesso scolastico Ettore Morelli. Infine, vengono discussi gli sviluppi futuri.

ABSTRACT

To survive a fire, the occupants must be able to evacuate the building before the situation becomes untenable. Evacuation time is therefore a critical factor and the available fire regulations recommend reducing it through adequate design of the structure and evacuation aids. In addition to the engineering decisions made in a building's design, occupant behavior is an important factor that significantly affects evacuation times. The poor knowledge people have about fire development and the dynamics of a fire emergency does not prepare them to have the best response during fires. Therefore, knowing the correct fire safety behavior is a personal protection strategy that would allow each occupant to significantly increase their chances of survival by reducing evacuation times and preventing common mistakes that occupants make.

In this thesis work, the development of an application for the fire evacuation simulation for the student of a primary and second grade school is proposed. Serious Games and Virtual Reality are proposed as a tool to acquire personal fire safety competitions. In the first chapter, the issues relating to Serious Games, Virtual Reality and BIM are deepened, an then applied in the case study. In the second chapter, the fire prevention regulations are provided, in particular the application of schools, fundamental for the case study. In the third chapter it is described how Serious Games and Virtual Reality are applied to fire prevention, providing the methodology used for the development of the application. The fourth chapter describes the application that was developed for the exodus of the students at the school complex Ettore Morelli. Finally, future developments will be discussed.

INDICE

INDICE DELLE FIGURE

1. SERIOUS GAME, VIRTUAL REALITY E BIM

1.1. Serious Games

Per comprendere meglio i videogiochi ed essere pienamente consapevoli del loro significato, del loro ruolo e della loro funzione, ripercorriamo brevemente le tappe principali della loro nascita e del loro sviluppo, con una breve descrizione delle principali trasformazioni. La storia dei videogiochi è legata all'evoluzione del computer e iniziò negli anni '50 e '60 del Novecento, quando gli scienziati informatici iniziarono a progettare semplici giochi e simulazioni. Il 1952 è riconosciuto come l'anno di nascita del primo videogioco "OXO", sviluppato da un ricercatore inglese di nome Alexander S. Douglas, e non è altro che il famoso gioco del tris. Se si considera lo scopo ludico dei videogiochi bisogna spostarsi qualche anno in avanti, più precisamente nel 1958, quando William Higinbotham creò "Tennis for Two" per intrattenere i visitatori del Brookhaven National Laboratory di New York. Questo gioco rappresenta, in forma stilizzata, il gioco del tennis, mediante il collegamento tra un oscilloscopio e uno schermo. La maggior parte dei primi videogiochi giravano sui sistemi centrali delle università statunitensi ed erano sviluppati nel tempo libero dagli studenti. Questi giochi però tendevano ad essere dimenticati, poiché all'epoca, la disponibilità dei computer era scarsa.

È nel 1962 all'interno del MIT (Massachusetts Institute of Technology), dove i tre ricercatori Steve Russel, Peter Samson e Dan Edwards, programmarono un gioco chiamato "Spacewar!". Il gioco, ambientato nello spazio, mostrava due navicelle spaziali che combattevano lanciando missili e lo scopo era quello di distruggersi evitando gli ostacoli lungo il percorso. Questo gioco ebbe molto successo e diventò il primo videogioco largamente diffuso della storia, nonostante non fosse accessibile a tutti.

Per la piena diffusione dei videogiochi bisognerà attendere gli anni ’70, con lo sviluppo delle prime console collegate alla televisione e alle sale giochi. Bushnell è stato una delle figure più importanti dell’epoca, avendo sviluppato il primo videogioco a gettoni, “Computer Space”, nel 1971, e avendo fondato, l’anno successivo, Atari, una delle più importanti società di videogiochi. Il primo videogioco prodotto da Atari fu “PONG”, che ottenne un successo mondiale. Il modello del gioco era molto simile al predecessore “Tennis for Two” e riproduceva le meccaniche del ping pong. La pubblicazione di “Space Invaders” da parte di Taito nel 1978 segna l’inizio dell’età dell’oro dei videogiochi arcade nelle sale giochi e dell’industria videoludica.

Molti videogiochi classici ancora oggi popolari sono stati creati negli anni ’80. All’inizio di questo decennio sono apparse sul mercato molte nuove console per i videogiochi e sono comparsi anche home computer sempre più a basso costo e potenti. Nel 1980 uscì “Pac Man”, il videogioco per eccellenza delle sale giochi, prodotto da Toru Iwatani. Dal 1981, due grandi produttori giapponesi di videogiochi, Nintendo e Sega, sono entrati nel mercato. In questo periodo si è assistito a una grande popolarità e sviluppo del videogioco, e vennero creati giochi di grande successo ancora oggi. Oltre a Pac Man, tra questi figurano “Tetris” e la prima serie di “Super Mario Bros”, che erano stati creati per una console della Nintendo, il Game Boy. Quest’ultimo, essendo la prima console portatile, porta a una rivoluzione nel mondo dei giochi.

Negli anni ’80 e ’90 il videogioco è soggetto a una seconda fase di sviluppo, e in questo periodo si assiste ad una crescita progressiva di questo media, passando dai giochi in bianco e nero alla grafica tridimensionale. La capacità di muoversi in tre direzioni invece delle due tradizionali ha creato immagini del mondo di gioco più realistiche e possibilità più complesse. I produttori di console e le aziende produttrici di videogiochi cercano di conquistare un mercato in continua espansione. La PlayStation, lanciata da Sony nel 1994, ha rappresentato un grande passo avanti rispetto alle console già esistenti.

All’inizio del nuovo millennio, oltre allo sviluppo delle console, l’avvento e la diffusione di internet tramite la connessione ADSL ha influito su questo processo

di crescita. In questo modo è stato possibile collegare la rete alla console, consentendo una maggiore espansione della stessa, aggiornare i giochi e, soprattutto, offrire nuove modalità di gioco. Il gioco online migliora notevolmente l'intrattenimento e l'interattività di questo media. Nel corso di questo decennio anche la tecnologia ha fatto grandi progressi, che hanno reso i mondi di gioco ancora più realistici. Con l'introduzione dell'intelligenza artificiale gli avversari simulati, o PNG, reagiscono autonomamente a ciò che accade nell'ambiente invece di comportarsi allo stesso modo in ogni situazione. Nascono i giochi open world, dove i giocatori possono scoprire i mondi da soli e determinare lo svolgimento della storia.

Negli ultimi dieci anni i videogiochi sono diventati un business miliardario e sono nati studi indipendenti che sviluppano giochi per tutte le piattaforme: computer, telefoni cellulare e tablet. Così, sempre più persone giocano ai videogiochi ogni giorno.

Da semplici puntini su schermi a pixel multicolori e paesaggi 3D iperrealistici, la storia dei videogiochi è stata segnata da sviluppi tecnologici che hanno permesso ai giocatori di addentrarsi nel mondo virtuale. Oggi la realtà virtuale permette ai giocatori di immergersi completamente nel gioco.

Videogiochi educativi

In seguito al grande sviluppo e al successo dei videogiochi, nel corso degli anni si sono diffusi anche i videogiochi con funzione educativa o progettati per facilitare l'apprendimento: si tratta dei cosiddetti *edutainment*. L'edutainment è un tentativo di migliorare e supportare l'educazione grazie alle potenzialità dell'entertainment, declinandosi nel *game based learning*, in generale, e nel *digital game based learning*, nello specifico del gioco digitale. In questo contesto, i videogiochi diventano *serious game*, cioè applicazioni interattive divertenti e coinvolgenti che pongono sfide complesse e consentono all'utente di acquisire competenze e abilità che possono essere trasferite al mondo reale. Ad esempio, esistono giochi che affrontano il tema del bullismo, altri quello della migrazione e sulle vittime di

guerra, che trasmettono i concetti di integrazione e inclusione e favoriscono la comprensione e la diversità. Alcuni di essi aiutano anche nell'apprendimento della lingua, della storia e della matematica. La *gamification* invece è l'applicazione di meccaniche e dinamiche di gioco a situazioni non di gioco per stimolare l'interesse attivo e il coinvolgimento dell'utente e per incoraggiare lo svolgimento di un'attività o l'acquisizione di un comportamento. Per questo motivo, l'adozione di specifiche meccaniche di gioco, come livelli di gioco, sfide, ricompense e punti, può agire per motivare i giocatori.

Figura 1. Categorie dei giochi

I videogiochi come strumento educativo sono ancora una novità e non hanno ottenuto un'accettazione unanime nel mondo dell'istruzione. Ciò è dovuto principalmente dal fatto che, soprattutto in Italia, il campo dell'educazione è visto come un settore da affrontare con serietà e, possibilmente, tramite lezioni frontali. Questa netta distinzione tra ciò che è educativo e ciò che è divertente si riscontra anche nel campo dei videogiochi destinati all'educazione, poiché sono pochi i videogiochi che riescono a combinare con successo la componente educativa e quella di intrattenimento. Un punto fondamentale sarebbe quello di abbandonare il concetto che educazione e divertimento siano opposti e divisi, ma bisognerebbe trovare un punto di incontro tra queste realtà. Per fare questo è necessario abbandonare la visione classica dell'istruzione e aprirsi all'uso di nuove tecnologie, che affiancherebbero l'istruzione classica senza sostituirla. I videogiochi si sono dimostrati un buono strumento per l'apprendimento, in quanto anche gli errori che

vengono fatti non sono percepiti come un fallimento, poiché permettono di ricominciare da capo facendo così imparare dalle esperienze fatte.

Serious Games

I Serious Games hanno confini molto ristretti e il loro successo dipende dall'equilibrio di tre componenti. In un gioco solitamente sono presenti tre caratteristiche e secondo Harteveld deve essere utilizzato l'approccio del Triadic Game Design. Questo implica una triade costituita dai mondi interdipendenti di realtà, significato e gioco che devono essere bilanciati durante il processo di progettazione. L'approccio di Harteveld si basa sui due paradigmi progettuali "Concurrent Design" e "Iterative Design". Ciò significa che, in primo luogo, per trovare l'ideale i tre mondi dovrebbero essere considerati contemporaneamente all'interno delle parti critiche del processo di progettazione (Concurrent Design), e in secondo luogo, un ciclo ripetuto di prototipazione, test, valutazione e riprogettazione continua fino a quando i requisiti per i tre mondi sono soddisfatti (Iterative Design).

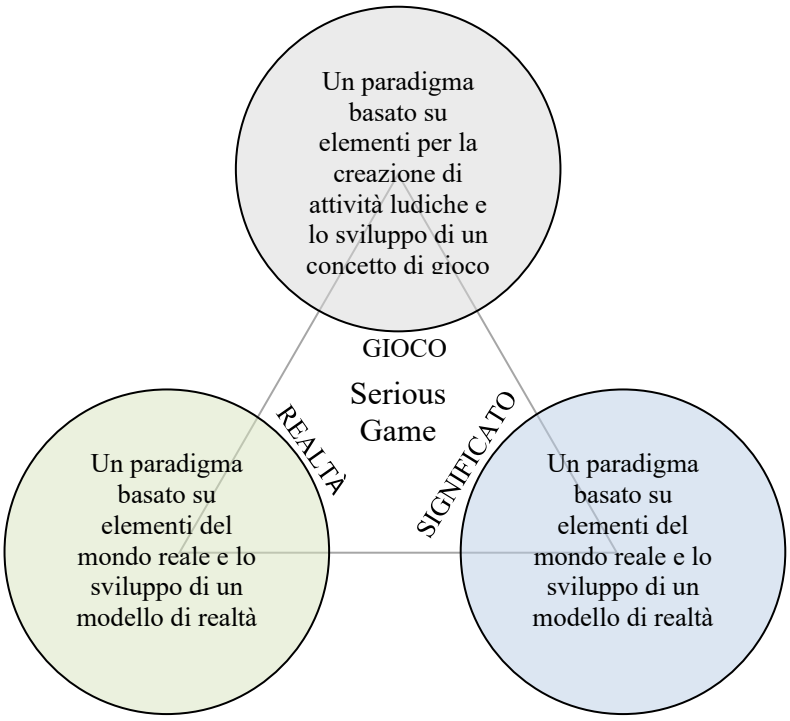


Figura 2. Triadic Game Design

Uno studio dell'Università di Utrecht ha confrontato l'efficacia dei Serious Games con altri metodi di apprendimento e i ricercatori hanno concluso che i Serious Games migliorano l'acquisizione di conoscenze e abilità cognitive. Tuttavia, ad oggi non esiste un modello che spieghi l'efficacia dei Serious Games. Probabilmente questo è dovuto al fatto che ogni gioco ha dinamiche uniche, per cui si creano dei modelli che non spiegano tanto il processo, ma che sarebbero applicabili solo al singolo gioco.

I Serious Games oggi vengono applicati in diversi ambiti: nell'educazione e sensibilizzazione, nel turismo, nell'addestramento, nella simulazione di operazioni militari, in medicina, nella progettazione.

1.2. Virtual Reality

Con i progressi della ricerca e dello sviluppo informatico, è possibile utilizzare la Realtà Virtuale nella progettazione e in diversi altri ambiti. Innanzitutto, vengono forniti i concetti e le definizioni di base di queste nuove tecnologie.

Il primo concetto sviluppatosi è quello di *Virtuality Continuum*, che rappresenta lo spettro delle possibilità tecnologiche tra il mondo fisico o l'ambiente reale e il mondo digitale o l'ambiente virtuale. Include tutte le tecnologie attuali che alterano la realtà con la grafica generata dal computer, nonché quelle ancora da sviluppare. In un continuum, le parti adiacenti sono quasi indistinguibili, ma gli estremi sono molto diversi. Pertanto, i limiti esatti dei vari termini non sono chiari. Il termine *Mixed Reality* copre qualsiasi ambiente in cui gli oggetti reali e virtuali sono combinati all'interno di un unico display. Secondo questo framework, la realtà mista copre la maggior parte del continuum ad eccezione degli estremi. I ricercatori Paul Milgram e Fumio Kishino hanno introdotto per la prima volta il concetto di Virtuality Continuum nel 1994. Inizialmente, il Virtuallity Continuum proposto da Milgram e Kishino considerava solo display visivi. Pertanto, le diverse sezioni all'interno del continuum tengono conto solo dell'aspetto visivo della fusione tra il

mondo fisico e quello digitale, non tenendo conto del suono, dell'olfatto, della sensazione tattile o del gusto. Il Virtuality Continuum è suddiviso in quattro categorie:

- Real Environment – Ambiente reale: consiste esclusivamente di oggetti reali o fisici e rappresenta l'estremità sinistra del Virtuality Continuum;
- Augmented Reality – Realtà aumentata: mondo reale aumentato con elementi digitali;
- Augmented Virtuality – Virtualità aumentata: mondo virtuale aumentato dall'inclusione di oggetti reali o fisici;
- Virtual Reality – Ambiente virtuale: consiste esclusivamente di oggetti digitali e rappresenta l'estremità destra del Virtuality Continuum.

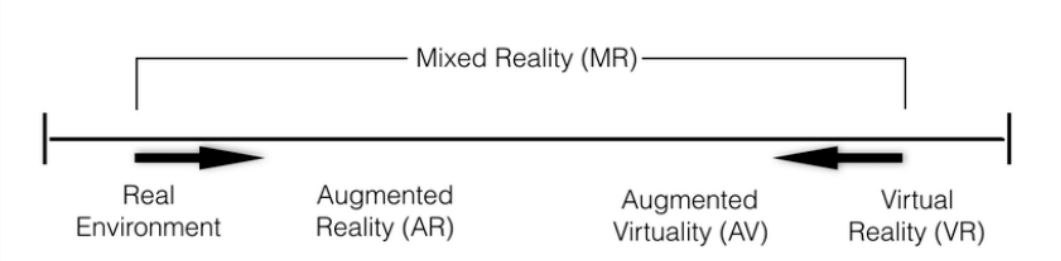


Figura 3. Virtuality Continuum

Dopo aver introdotto questo concetto principale, è possibile addentrarci nelle definizioni di Realtà Virtuale, Realtà Aumentata e Realtà Mista.

La *Realtà Virtuale* (VR, *Virtual Reality*) nasce da una combinazione di dispositivi hardware e software che consentono la creazione di uno spazio virtuale che simula la realtà effettiva, all'interno del quale l'utente può muoversi liberamente. La realtà virtuale si divide in due famiglie:

- Realtà virtuale immersiva: l'utente è isolato dall'ambiente esterno ed è immerso in un ambiente virtuale modellato al computer. Nella VR immersiva vengono utilizzati dispositivi che stimolano i sensi dell'utente, in particolare vengono impiegati i visori;
- Realtà virtuale non immersiva: in questo caso l'ambiente ricreato è meno impattante sui sensi dell'utente, poiché non è possibile interagire con

l'ambiente costruito ma si ha solamente una percezione visiva sullo schermo.

La *Realtà Aumentata* (AR, *Augmented Reality*) indica una tecnologia che consente di sovrapporre il mondo digitale al mondo reale, arricchendo l'esperienza sensoriale con strumenti tecnologici. Questo è possibile mediante l'utilizzo di dispositivi mobili o di computer dotati di webcam. L'obiettivo della Realtà Aumentata è quello di sovrapporre contenuti multimediali alla realtà esistente in tempo reale.

La *Realtà Mista* (MR, *Mixed reality*) combina elementi del mondo reale e di quello digitale, fondendo quindi Realtà Virtuale e Realtà Aumentata. L'obiettivo della Realtà Mista è quello di immergere l'utente in un mondo reale dove può considerare gli oggetti virtuali come esistenti. In questo modo è possibile osservare il mondo reale ricevendo informazioni utili grazie alla Realtà Aumentata, ma anche vedendo oggetti virtuali come reali.

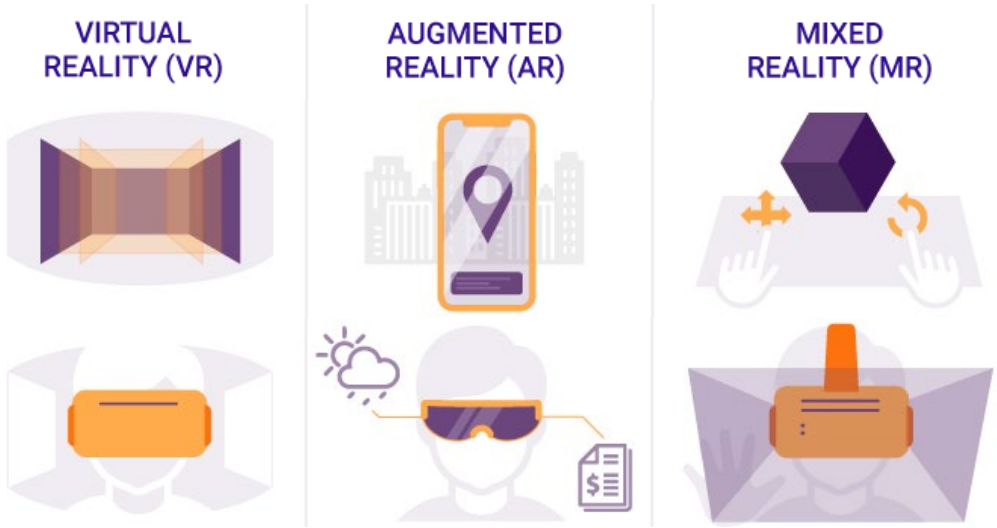


Figura 4. Classificazione Realtà (tratto da RubyGarage)

La Virtual Reality è uno strumento molto versatile, attualmente utilizzata in molti ambiti, sia privati che pubblici. Sempre più aziende e imprese utilizzano la Virtual Reality per facilitare la comunicazione con i clienti e la promozione dei prodotti. Oltre che nell'architettura e nell'interior design, viene anche sfruttata nel settore automobilistico, immobiliare e turistico. La realtà virtuale offre graficamente

l'opportunità di vedere in anticipo il progetto finito in tutte le fasi, correggere errori e apportare modifiche e ridurre i rischi. La realtà virtuale è maggiormente impiegata nel mercato dei videogiochi, in cui gli strumenti sono più implementati e testati, e questo è uno dei motivi principali per cui la Virtual Reality è nota alla maggior parte delle persone. Oggi la realtà virtuale sta diventando molto popolare tra i giocatori e molti produttori di videogiochi stanno pianificando di renderla compatibile con PC e console, creando un doppio formato di gioco: sia giochi tradizionali che giochi virtuali. Altre aree in cui la realtà virtuale è di maggiore interesse sono la ricerca (spazio, medicina) e l'istruzione, dove la realtà virtuale può fornire un aiuto nel processo di digitalizzazione delle risorse e nell'apprendimento, per fornire agli studenti un tipo di istruzione più completo, interattivo e stimolante. La Virtual Reality sta inoltre assumendo un ruolo rilevante nell'ambito artistico e culturale, venendo utilizzata nel sistema dei media.

Dispositivi disponibili

Sono disponibili vari strumenti che permettono di immergersi completamente all'interno della realtà virtuale. Oggi esistono diverse tecnologie per fruire della realtà virtuale, come smartphone e personal computer, collegati a un visore ottico e a dei controllers. Di seguito vengono riportati alcuni dei dispositivi più utilizzati:

- Oculus: è un visore di realtà virtuale sviluppato da Oculus VR indossabile sul viso. Il dispositivo include non solo il visore, ma anche i controller che migliorano l'esperienza dell'utente. Il dispositivo è costituito da due display AMOLED con una risoluzione di 2160 x 1200 pixel e i controller trasportano le mani e i gesti direttamente nel gioco;
- HTC Vive: è un dispositivo di realtà virtuale sviluppato da Valve in collaborazione con HTC. Il dispositivo non solo permette di vedere il mondo virtuale attraverso un visore ottico, ma grazie a una nuova tecnologia chiamata "room scale", trasforma l'ambiente circostante in uno spazio tridimensionale, permettendo di muoversi quasi liberamente. Questa nuova tecnologia rende l'esperienza particolarmente coinvolgente, consentendo agli utenti di interagire quasi interamente con il mondo di gioco:

- Cyberith Virtualizer: è una piattaforma di locomozione di realtà virtuale che simula il movimento completo in un ambiente virtuale seguendo i movimenti fisici dell'utente;
- Cave: il Cave Automatic Virtual Environment è un ambiente di realtà virtuale immersiva composto da una stanza cubica e proiettori video rivolti da tre a sei lati.

1.3. BIM

Il *Building Information Modeling* (*BIM*), è uno dei più importanti progressi recenti nel settore dell'architettura, dell'ingegneria e delle costruzioni. Utilizzando il BIM, è possibile creare digitalmente uno o più precisi modelli virtuali di un edificio, il che supporta il processo di progettazione in più fasi e consente una migliore analisi e controllo rispetto ai processi manuali. Una volta completati, questi modelli contengono la geometria precisa e le informazioni necessarie per supportare le attività di costruzione, produzione e manutenzione di un edificio. Con il termine BIM si intende un ambiente informativo che consente cambiamenti di processo estesi nella gestione, progettazione e costruzione.

Il primo a diffondere il termine BIM è stato Jerry Laiserin nel 2002. La definizione di BIM non è standard e si sta lavorando su una norma europea per la standardizzazione della definizione. L'ultima definizione è "*use of shared digital representation of a built object (including buildings, bridges, roads, process plants, etc.) to facilitate design, construction and operation processes to form a reliable basis for decisions*", ovvero "*utilizzo di condivise rappresentazioni digitali di oggetti costruiti (inclusi edifici, ponti, strade, impianti tecnologici, ecc.) per facilitare i processi di progettazione, costruzione e funzionamento per creare una base affidabile per il processo decisionale*".

Il BIM può essere definito come il processo di sviluppo, crescita e analisi di modelli multidimensionali virtuali generati digitalmente da programmi su computer. Il ruolo del BIM nel settore delle costruzioni è quello di comunicare, collaborare,

modellare e ottimizzare i progetti durante l'intero ciclo di vita di una struttura. Il BIM è una rappresentazione dei vari modelli di dati di costruzione associati alle varie discipline che definiscono un edificio. I dati contenuti nel modello sono parecchie, in quanto definiscono tutte le informazioni su un particolare componente di progettazione. Con il BIM, il ciclo di vita di un oggetto costruito è definito dalla fase di progettazione , attraverso la fase di costruzione, fino alle fasi di uso e manutenzione. Il BIM può contenere qualsiasi informazione su un edificio o sue parti. Le informazioni più comunemente raccolte nel BIM riguardano la posizione geografica, la geometria, le proprietà di materiali/componenti/sistemi ed elementi tecnici, le fasi di costruzione, le attività di manutenzione e lo smaltimento alla fine del ciclo. Il BIM può essere applicato con diversi livelli di maturità. Anche un semplice CAD fa parte del modello informativo. Oggi si hanno sei livelli di applicazioni BIM, a seconda informazioni inserite e del grado di collaborazione applicato.

Livelli di maturità

In generale, l'adozione del BIM è considerata come una transizione graduale di crescita, dove si passa da un processo frammentato diviso in fasi fino ad un processo dove tutti i soggetti che partecipano al progetto del modello lavorano in sincronia sull'edificio. La collaborazione è l'obiettivo del processo, e nel 2017 è stato raggiunto il massimo grado di maturità, ovvero il BIM collaborativo. Come appena descritto esistono diversi livelli di maturità del BIM:

- Livello 0 – CAD standardizzato: organizzazione del lavoro tradizionale attorno a un sistema di standard;
- Livello 1 – BIM solitario o non collaborativo: utilizzo del metodo di progettazione parametrica e gestione dei dati solo nel proprio workflow, senza la collaborazione con altri esperti. Vengono seguiti degli standard interni o internazionali;
- Livello 2 – BIM collaborativo: si ha una collaborazione tra tutti i soggetti coinvolti, dove ognuno opera separatamente sul proprio modello e la

collaborazione è gestita dal BIM Leading consultant, che unisce i modelli in un unico modello integrato;

- Livello 3 – BIM condiviso: tutti i professionisti lavorano contemporaneamente sullo stesso modello e gli aggiornamenti sono in tempo reale.

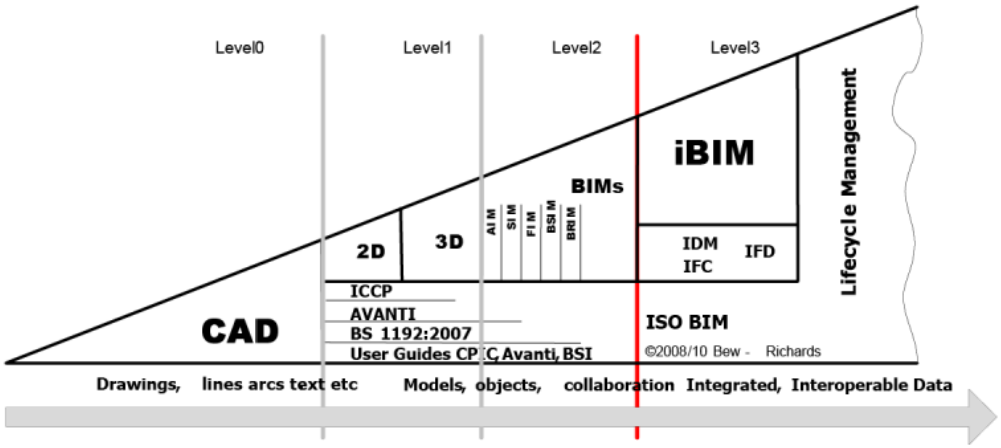


Figura 5. Livelli di maturità del BIM

Livelli di sviluppo

Il livello di sviluppo viene spesso frainteso come livello di dettaglio, ma entrambi sono diversi l'uno dall'altro. Il livello di dettaglio indica la proporzione di dettagli racchiusi all'interno dell'elemento del modello. Il livello di sviluppo (LOD) è il grado in cui la geometria degli elementi e le informazioni ad esso collegato sono state pensate. Può anche essere indicato come il grado in cui i membri del team coinvolti possono dipendere dalle informazioni durante l'utilizzo del modello. Si può dire che il livello di dettaglio funge da input per l'elemento, mentre il livello di sviluppo è l'output. Il grado di sviluppo ottenibile è suddiviso in cinque livelli:

- LOD 100: fase di pre-progettazione dove l'elemento è un modello concettuale in cui sono definiti parametri e informazioni che sono considerate come un'approssimazione;
- LOD 200: fase di progettazione schematica dove l'elemento mostra caratteristiche geometriche. Può anche includere alcune informazioni non

geometriche e come per il LOD 100 le informazioni in questa fase sono considerate un'approssimazione;

- LOD 300: fase di sviluppo della progettazione degli edifici. Le rappresentazioni grafiche possono essere le stesse della rappresentazione grafica di LOD 200, ma la geometria e le caratteristiche in questa fase sono considerate accurate. Le informazioni in questa fase possono essere utilizzate nella fase di costruzione;
- LOD 350: fase della documentazione edilizia. Le informazioni ottenute sono le stesse di LOD 300, ma sono incluse anche le interfacce con altri componenti dell'edificio. Indica il modo in cui il componente può essere installato e il modo in cui interagisce con gli impianti di altri edifici;
- LOD 400: fase di costruzione dove sono inclusi dettagli come la fabbricazione, l'assemblaggio e l'installazione dei componenti. Questi dettagli sono di grande utilità per i fornitori per la produzione dei componenti rappresentati. Può anche includere informazioni non visive;
- LOD 500: fase as-built, ovvero come se fosse costruito. Questi sono i campi che vengono verificati in termini di forma, dimensione, quantità, orientamento e posizione, che sono considerati rappresentazioni accurate degli elementi edilizi post-costruzione. Questi elementi possono essere utilizzati come riferimenti per il funzionamento e la manutenzione da parte dei facility manager.

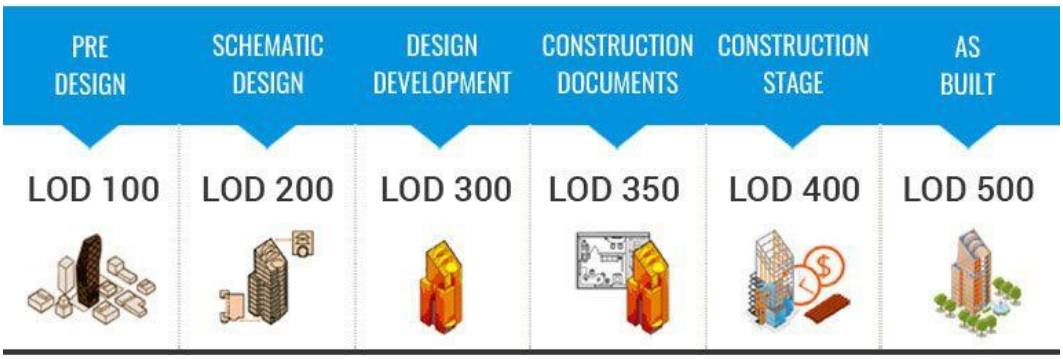


Figura 6. LOD del BIM

Dimensioni del Bim

Il BIM viene spesso erroneamente indicato solo come modello digitale 3D. Il BIM, invece, si estende all'intero processo edilizio, comprendendo la pianificazione, la progettazione, la costruzione, la gestione e l'intero ciclo di vita dell'edificio. Si parla quindi di dimensioni del BIM, e si hanno le seguenti sette dimensioni:

- 1D: concept del progetto;
- 2D: disegni 2D come planimetrie, prospetti e sezioni;
- 3D: rappresentazione tridimensionale;
- 4D: analisi della durata o dei tempi;
- 5D: analisi dei costi;
- 6D: fase di gestione operativa;
- 7D: valutazione della sostenibilità.

Oltre alle sette dimensioni citate ad oggi esiste un dibattito aperto su tre nuove dimensioni del BIM:

- 8D: sicurezza in fase di progettazione e realizzazione dell'opera;
- 9D: costruzione snella;
- 10D: industrializzazione delle costruzioni.

Figura 7. Dimensioni del BIM

Esistono differenze significative nella prevalenza delle procedure BIM tra gli operatori europei. I Paesi Bassi sono lo stato più all'avanguardia, seguito dal Regno Unito, mentre l'Italia è nelle ultime posizioni nelle statistiche.

In Italia, il Decreto del Ministero delle Infrastrutture numero 560 del 1º dicembre 2017 determina come e quando attuare il BIM per gli appalti pubblici di opere civili e infrastrutture. Dal 2019 si applica alle opere con un importo di proposta di 100 milioni di euro. Tale importo diminuirà gradualmente nel corso degli anni fino a raggiungere 1 milione dal 1 gennaio 2025.

In questo lavoro di tesi Revit viene utilizzato per la modellazione 3D del caso studio. Il software è specificamente utilizzato per il BIM dove le sue caratteristiche hanno reso facile creare strutture complesse, progettare e documentare in breve tempo con grande precisione.

2. PREVENZIONE INCENDI IN AMBITO SCOLASTICO

In questo capitolo viene descritta l'evoluzione della normativa italiana riguardante la Prevenzione Incendi, in particolare i decreti applicati alle scuole.

2.1. Evoluzione della Prevenzione Incendi in Italia

In Italia la disciplina della Prevenzione Incendi si è sviluppata attraverso un lungo periodo. Sono stati necessari quarantatré anni per la prima definizione di "Prevenzione Incendi". Questo percorso è stato avviato nel nostro Paese con il **Regio Decreto legge** emanato il **27/02/1939** che istituisce il Corpo Nazionale dei Vigili del Fuoco. Due anni dopo venne emanata la **legge n. 1570 del 27/12/1941**, con la quale si istituisce l'unione di tutti i Corpi Provinciali e Civici dei Pompieri. Inizia così la storia della Prevenzione Incendi, ma le procedure di prevenzione incendi hanno una storia più recente.

Il **D.P.R. n. 547 del 27/04/1955** – Norme per la prevenzione degli infortuni sul lavoro – prevedeva che il Comando dei Vigili del Fuoco doveva prendere in esame i progetti di nuovi edifici o impianti. In seguito, a opera costruttiva ultimata, veniva eseguito il collaudo.
Con il **D.P.R. n. 689 del 26/05/1959** – Determinazione delle aziende e lavorazioni soggette, ai fini della prevenzione degli incendi, al controllo del Comando dei Vigili del Fuoco – e la **Legge n. 469 del 13/05/1961** – Ordinamento dei servizi antincendi e del Corpo nazionale dei Vigili del Fuoco e stato giuridico e trattamento economico del personale dei sottufficiali, vigili scelti e vigili del Corpo Nazionale dei Vigili del Fuoco – vengono introdotti i termini "aziende" e "lavorazioni soggette", terminologie che vengono tuttora utilizzate.
Con la **legge n. 966 del 26/07/1965** – Disciplina delle tariffe, delle modalità di pagamento e dei compensi al personale del Corpo Nazionale dei Vigili del Fuoco

per i servizi a pagamento – venne istituito il Certificato di Prevenzione Incendi (C.P.I.), nel quale vennero identificate 100 attività che dovevano soddisfare precisi requisiti antincendio. Da questo momento in poi, le attività dovranno quindi disporre di C.P.I. e richiedere le visite di accertamento da parte del Corpo dei Vigili. Due mesi dopo, con il **D.M. n. 1973 del 27/09/1965** – Determinazione delle attività soggette alle visite di prevenzione incendi – le 100 attività soggette vennero meglio indicate e furono divise secondo la tempistica delle visite.
Il **D.M. 16/02/1982** – Modificazioni del DM settembre 1965, concernente la determinazione delle attività soggette alle visite di prevenzione incendi – rimasto in vigore per 29 anni, andava a modificare il D.M. precedente, modificando l'elenco delle attività soggette a prevenzione incendi, che passarono da 100 a 97, e le tempistiche delle visite.
Il **D.P.R. n. 577 del 29/07/1982** – Approvazione del regolamento concernente l'espletamento dei servizi antincendi – introdusse i principi e le misure tecniche fondamentali per prevenire situazioni di rischio.

Un evento che diede inizio alla revisione della normativa sulla prevenzione incendi fu l'incendio avvenuto il 13 febbraio 1983 al Cinema Statuto di Torino, dove morirono 64 persone. Le indagini successive alla tragedia dimostrarono che le cause dell'incendio andavano oltre le responsabilità dei singoli, e venne messo così in dubbio l'intero sistema di leggi vigenti in materia di sicurezza.
Così, un anno dopo la tragedia, venne emanata la **legge n. 818 del 07/12/1984** – Nulla Osta Provvisorio per le attività soggette ai controlli di prevenzione incendi, modifica degli articoli 2 e 3 della legge 4 marzo 1982, n. 66, e norme integrative dell'ordinamento del Corpo Nazionale dei Vigili del Fuoco – con la quale venne introdotta la figura del "professionista iscritto in albi ministeriali". Alle attività prive di C.P.I. veniva rilasciato il N.O.P. (nulla osta provvisorio), con la promessa di adeguare tali attività il prima possibile agli obblighi antincendio più necessari. Le attività che non richiedevano né il C.P.I. e nemmeno il N.O.P. venivano sottoposti a sanzioni penali.
Qualche mese dopo, il **D.M. 08/03/1985** – Direttive sulle misure più urgenti ed essenziali di Prevenzione Incendi ai fini del rilascio del nulla osta provvisorio di cui alla legge 7 dicembre 1984, n. 818 – definì il concetto di "operatività antincendio",

che rappresenta la possibilità di lavorare in condizioni di sicurezza per le squadre di soccorso.

Il **DM 26/08/1992** – Norme di prevenzione incendi per l'edilizia scolastica – ha come oggetto la sicurezza antincendio negli edifici e nei locali destinati ad uso scolastico al fine di tutelare l'incolumità delle persone e salvaguardare i beni contro il rischio di incendio. È la prima norma di prevenzione incendi destinata all'edilizia scolastica, in cui viene introdotta la redazione di un piano di emergenza con l'obbligo di verificarlo e testarlo almeno due volte nel corso dell'anno scolastico.

Il **D. Lgs n. 626 del 19/09/1994** – Attuazione delle direttive […] riguardanti il miglioramento della sicurezza e della salute dei lavoratori durante il lavoro – aveva come fine quello di effettuare la valutazione dei rischi e redigere un piano di sicurezza completo di procedure in caso di emergenza.

Con il **D.P.R. n. 37 del 12/01/1998** – Regolamento recante disciplina dei procedimenti relativi alla prevenzione incendi, a norma dell'articolo 20, comma 8, della legge 15 marzo 1997, n. 59 – l'attività amministrativa della prevenzione incendi venne snellita e velocizzata. Questo decreto introdusse tempi di riposta fissi da parte dei Vigili del Fuoco per esaminare un progetto. Altra cosa che venne aggiunta fu la D.I.A. (Denuncia di inizio attività).

Con il **D.M. 29/12/2005** – Direttive per il superamento del regime del nulla osta provvisorio, ai sensi dell'articolo 7 del decreto del Presidente della Repubblica 12 gennaio 1998, n. 37 – dal 31/05/2009 tutti i N.O.P. cessavano di esistere e rimanevano attive solo le attività sprovviste di C.P.I. o D.I.A.

Il **D. Lgs. n. 139 del 08/03/2006** – Riassetto delle disposizioni relative alle funzioni ed ai compiti del Corpo Nazionale dei Vigili del Fuoco – riprendeva i concetti fondamentali della Prevenzione Incendi e responsabilizzava ancor di più i professionisti antincendio iscritti nell'elenco ministeriale. Inoltre, il decreto abrogava la legge n. 818 del 07/12/1984.

Il **D.M. 09/05/2007** – Direttive per l'attuazione dell'approccio ingegneristico alla sicurezza antincendio – è molto importante, poiché introduce il concetto di approccio ingegneristico nella prevenzione incendi.

Il **D. Lgs. n. 81 del 09/04/2008** – Attuazioni dell'articolo 1 della Legge 3 agosto 2007, n. 123 in materia di tutela della salute e della sicurezza nei luoghi di lavoro – è considerato il decreto più importante in merito alla sicurezza nei luoghi di lavoro e per la prevenzione incendi.

La svolta si ha dopo 29 anni dalla emissione del D.M. 16/02/1982, con l'emanazione dei seguenti decreti.

Con il **DPR n. 160 del 07/09/2010** – Regolamento per la semplificazione ed il riordino della disciplina sullo Sportello Unico per le Attività Produttive – viene semplificata la burocrazia della Prevenzione Incendi. Viene individuato il S.U.A.P., che diventa l'unico organo competente a cui inoltrare le istanze di prevenzione incendi, che di seguito le inoltrerà al Comando Provinciale dei Vigili del Fuoco. Inoltre, viene istituito un supporto dato ai S.U.A.P. che consiste in un portale, denominato "impresa in un giorno", al fine di fornire servizi informativi e operativi. I Vigili del Fuoco verificano la regolarità della S.C.I.A. (segnalazione certificata di inizio attività) presentata dal S.U.A.P. e, in caso di esito positivo, viene rilasciata la ricevuta. Alla fine del processo, il richiedente ottiene la concessione di apertura dell'attività.

Il **D.P.R. n. 151 del 01/08/2011** – Regolamento recante semplificazione della disciplina dei procedimenti relativi alla prevenzione degli incendi – contiene una semplificazione delle pratiche amministrative.
Il decreto, in sintesi, prevede:

- nuove descrizioni e nuove attività;
- riduzione del numero di attività soggette ai controlli di prevenzione incendi (da 97 a 80);
- suddivisione delle attività in tre categorie (A, B, C) in proporzione al livello di complessità;
    - Categoria A (attività semplici): attività dotate di "regola tecnica" di riferimento, denominata R.T.V. (Regola Tecnica Verticale), contraddistinte da un livello limitato di complessità;

    - Categoria B (attività mediamente complesse): attività presenti in A, ma caratterizzate da un livello di complessità maggiore. Sono inoltre comprese le attività sprovviste di una specifica regolamentazione tecnica;
    - Categoria C (attività complesse): attività con un alto livello di complessità, indipendentemente dalla presenza o meno della regola tecnica.
- abolizione del C.P.I. e relativa sostituzione con la S.C.I.A;
- attestazione di rinnovo periodico ogni 5 o 10 anni;
- N.O.F. (Nulla Osta di Fattibilità);
- richiesta di sopralluoghi in corso d'opera;
- introduzione della dichiarazione di non aggravio di rischio;
- raccordo con le pratiche del S.U.A.P.;
- inasprimento delle sanzioni amministrative e penali.

In questi punti sono concentrati 70 anni di storia di prevenzione incendi.

Nel 2015 ci fu un grande cambiamento in seguito alla promulgazione del **D.M. 03/08/2015**, il cosiddetto **Codice**. L'obiettivo fu quello di realizzare il "Testo Unico della Prevenzione Incendi" per fornire un unico riferimento normativo. Questo decreto verrà meglio illustrato nel paragrafo successivo.

2.2. D.M. 03/08/2015

Il D.M. 03/08/2015 – Norme tecniche di prevenzione incendi – nasce per dare ai progettisti un unico strumento attraverso il quale applicare i metodi di prevenzione incendi. L'obiettivo del decreto è di semplificare e razionalizzare l'attuale corpo normativo relativo alla prevenzione degli incendi. Il Codice è stato elaborato per fornire un testo modulare e sempre aggiornato nel tempo in funzione al progresso tecnologico e agli standard internazionali. Inoltre, nel Codice cambia completamente l'approccio, poiché si passa da un approccio di tipo prescrittivo ad un approccio di tipo prestazionale; in questo modo al professionista antincendio

viene lasciata più libertà in merito alla valutazione del rischio relativo ad ogni situazione.

Obiettivi della semplificazione

Al fine di semplificare e snellire il Codice, l'impostazione del documento ha le seguenti caratteristiche:

- Generalità: le metodologie di progettazione della sicurezza antincendio possono essere applicate a tutte le attività considerate;
- Semplicità: si prediligono soluzioni più semplici a parità di sicurezza;
- Modularità: il documento è organizzato in moduli di agevole accessibilità;
- Flessibilità: sono indicate diverse soluzioni progettuali prescrittive i prestazionali;
- Standardizzazione e integrazione: il linguaggio usato è conforme agli standard internazionali;
- Inclusione: punto fondamentale è la sicurezza delle persone, in relazione anche alle diverse abilità;
- Contenuti basati sull'evidenza: il documento è basato sulla ricerca e sull'esperienza maturata negli anni;
- Aggiornabilità: il documento è redatto in modo da poter essere aggiornato all'avanzamento tecnologico.

Struttura

Il documento è composto da quattro sezioni:

Sezione G – Generalità: contiene i principi fondamentali per la progettazione della sicurezza antincendio, applicabili indistintamente alle diverse attività;
Sezione S – Strategia antincendio: fornisce le indicazioni delle misure antincendio di prevenzione, protezione e gestionali applicabili a tutte le attività;

Sezione V – Regole Tecniche Verticali: contiene specifiche indicazioni di prevenzione incendi applicabili a specifiche attività, che sono aggiuntive o integrative a quelle previste nella sezione S;

Sezione M – Metodi: contiene le metodologie innovative di progettazione antincendio, volte alla risoluzione di specifiche problematiche tecniche della progettazione antincendio (FSE: Fire Safety Engineering).

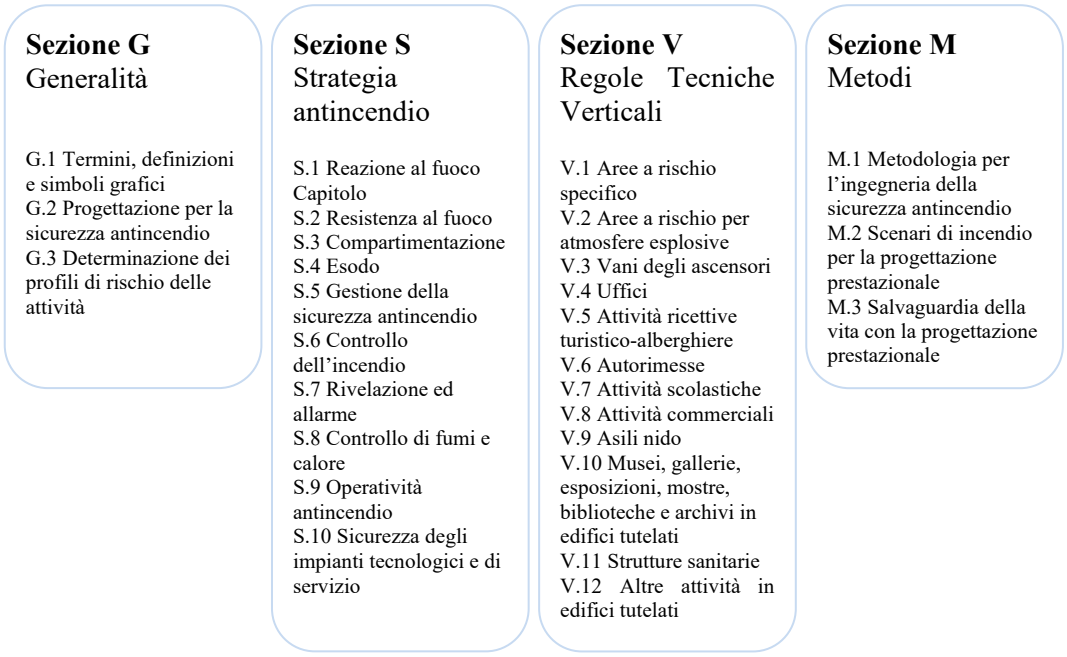


Figura 8. Struttura del Codice

La sezione M, relativa ai metodi, favorisce l'utilizzo dei metodi dell'ingegneria della sicurezza antincendio, in quanto sono previste soluzioni multiple. Questo è un grande cambiamento per la prevenzione incendi, poiché il progettista attraverso gli strumenti della Fire Safety Engineering assume un ruolo determinante nella progettazione della sicurezza antincendio. Questo metodo è anche definito di tipo "prestazionale" e si contrappone a quello di tipo "prescrittivo", ovvero costituito da regole tecniche.

Approccio prescrittivo

Questo approccio metodologico si basa sull'applicazione regole rigide e standardizzate stabilite dal legislatore, che impongono al progettista di seguire procedure, metodi di calcolo e aspetti costruttivi dettagliati ben definiti. Un tempo

questo approccio risultava essere facile da usare ed efficiente per il progettista, ma negli ultimi anni, con l'avanzare della tecnologia, queste regole sono diventate meno adattabili a situazioni specifiche di progetto, richiedendo l'uso di strumenti più flessibili.

Approccio prestazionale

Questo nuovo approccio metodologico si differenzia dal precedente in quanto richiede una serie di valutazioni e considerazioni preliminari diverse per ogni situazione. É il progettista a definire le prestazioni da raggiungere e a verificarne l'ottenimento, mediante l'uso di modelli di calcolo che prevedono gli effetti di un determinato evento e delle rispettive misure adottate. Questo metodo presenta una serie vantaggi, tra cui una maggiore flessibilità, che consente di affrontare situazioni in cui la normativa non risulta applicabile, e un risparmio dal punto di sui materiali e sui costi di gestione.

I metodi di questo approccio sono presenti nella sezione M – Metodi del Codice di Prevenzione Incendi. Questa sezione è suddivisa in tre parti:

Capitolo M.1 – Metodologia per l'ingegneria della sicurezza antincendio: contiene l'iter progettuale da seguire per la progettazione prestazionale. Questa metodologia si suddivide in due fasi: analisi preliminare e analisi quantitativa.
Capitolo M.2 – Scenari di incendio per la progettazione prestazionale: contiene indicazioni per l'applicazione del metodo e l'individuazione degli scenari di progetto da impiegare per l'analisi quantitativa.
Capitolo M.3 – Salvaguardia della vita con la progettazione prestazionale: contiene lo studio dell'esodo dal punto di vista della Fire Safety Engineering introducendo il criterio ASET > RSET, dove ASET (Available Safe Escape Time) è il tempo disponibile per l'esodo e RSET (Required Safe Escape Time) è il tempo richiesto per l'esodo.

Come introdotto precedentemente, la metodologia di progettazione prestazionale del capitolo M.1 consiste in due fasi: una fase di analisi preliminare e una fase di analisi quantitativa.

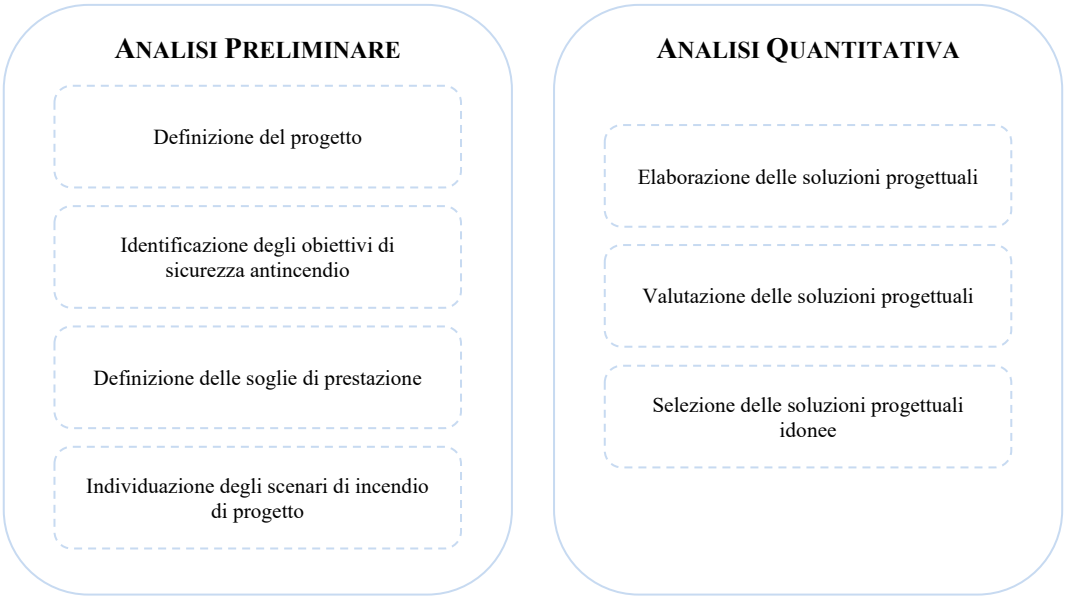


Figura 9. Metodologia progettazione ingegneria della sicurezza antincendio

L'analisi preliminare è composta dalle seguenti sotto-fasi:

- Definizione del progetto: costituisce la parte generale dell'analisi preliminare, nel quale si esplicita lo scopo del progetto, descrivendo l'edificio, la destinazione d'uso, i pericoli di incendio e i fruitori della stessa, inoltre si evidenziano le eventuali difformità rispetto le norme vigenti, andando infine a chiarire i motivi che conducono all'uso della Fire Safety Engineering, da cui derivano gli obiettivi di sicurezza antincendio;
- Identificazione degli obiettivi di sicurezza antincendio: vengono definiti gli obiettivi di sicurezza antincendio. Questi obiettivi specificano qualitativamente, ad esempio, il livello di salvaguardia dell'incolumità degli occupanti, il massimo danno tollerabile all'attività e al suo contenuto, la continuità d'esercizio a seguito di un evento incidentale;
- Definizione delle soglie di prestazione: gli obiettivi antincendio vengono tradotti in soglie di prestazione (performance criteria), di tipo quantitativo e qualitativo, fondamentali per la salvaguardia della vita umana, rispetto ai quali si può svolgere la valutazione oggettiva di sicurezza antincendio;
- Individuazione degli scenari di incendio di progetto: gli scenari antincendio rappresentano una schematizzazione degli eventi che possono verificarsi all'interno dell'attività presa in esame in relazione alle caratteristiche del

focolare, dell'edificio e degli occupanti. La procedura di identificazione, selezione e quantificazione degli scenari di incendio di progetto è descritta nel capitolo M.2.

L'analisi quantitativa è composta dalle seguenti sotto-fasi:

- Elaborazione delle soluzioni progettuali: vengono elaborate le soluzioni progettuali per l'attività in analisi, le quali saranno da sottoporre alla successiva verifica degli obiettivi di sicurezza antincendio;
- Valutazione delle soluzioni progettuali: vengono calcolati gli effetti sull'attività degli scenari di incendio di progetto per ciascuna soluzione progettuale individuata nella fase precedente. Viene, a tale fine, utilizzato dal professionista un modello di calcolo analitico o numerico. Una volta ottenuti i dati di output si verifica il rispetto delle soglie di prestazione prefissate per le soluzioni adottate e per ciascun scenario di progetto ipotizzato.
- Selezione delle soluzioni progettuali idonee: viene selezionata la soluzione progettuale finale dal professionista antincendio fra quelle che sono state valutate positivamente.

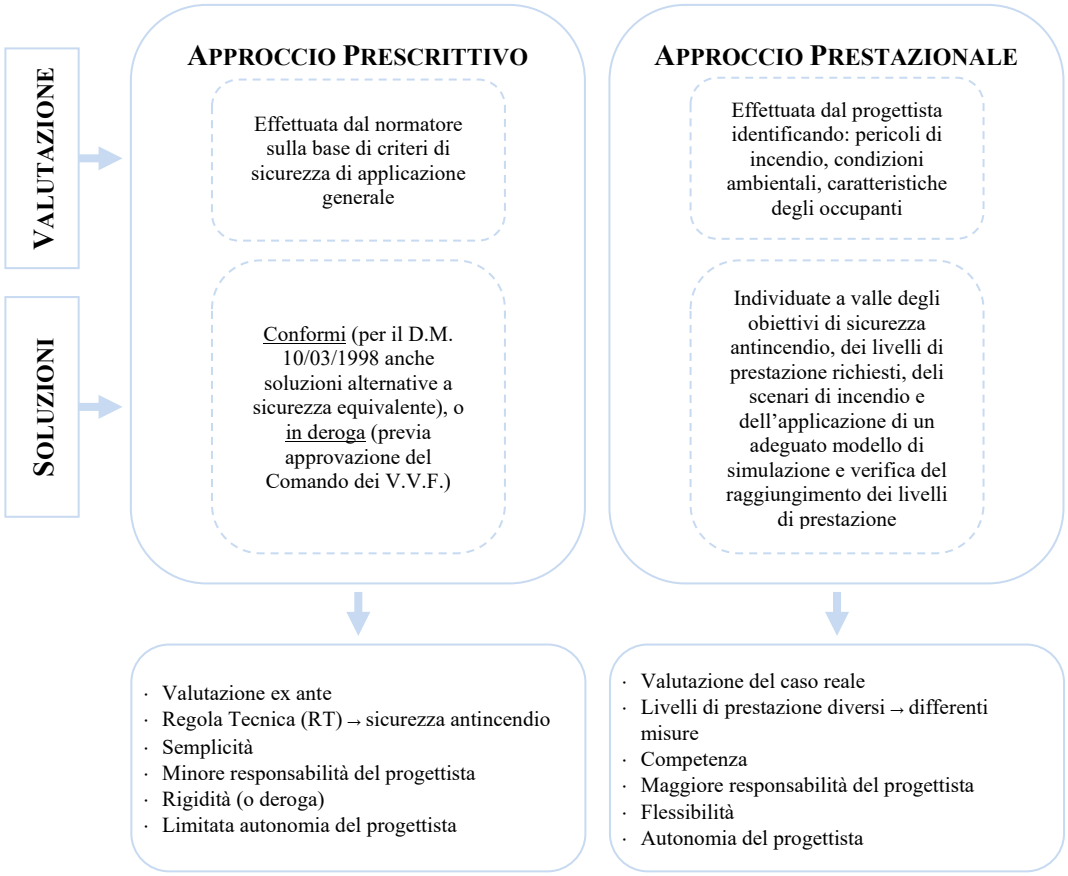


Figura 10. Differenze tra approccio prescrittivo e approccio prestazionale

2.3. Pianificazione dell'emergenza degli edifici scolastici

Come accennato precedentemente, nel D.P.R. n. 151 del 01/08/11 è riportato un elenco di 80 attività soggette alle visite e ai controlli di prevenzione incendi, le quali sono raggruppate in tre categorie in proporzione al livello di complessità:

- **Categoria A** (attività semplici): attività dotate di "regola tecnica" di riferimento, denominata R.T.V. (Regola Tecnica Verticale), contraddistinte da un livello limitato di complessità;
- **Categoria B** (attività mediamente complesse): attività presenti in A, ma caratterizzate da un livello di complessità maggiore. Sono inoltre comprese le attività sprovviste di una specifica regolamentazione tecnica;

- **Categoria C** (attività complesse): attività con un alto livello di complessità, indipendentemente dalla presenza o meno della regola tecnica.

Di queste 80 attività le scuole sono in sessantasettesima posizione.

N.	ATTIVITÀ (DPR 151/2011)	CATEGORIA		
		A	B	C
67	**Scuole di ogni ordine, grado e tipo**, collegi, accademie **con oltre 100 persone presenti**; [1, 2, 3, 4, 5, 6] Asili nido con oltre 30 persone presenti.	Fino a 150 persone	- Oltre 150 e fino a 300 persone; - Asili nido	Oltre 300 persone

Figura 11. Attività D.P.R. 151/2011

Con l'emanazione del D.M. 07/08/2017 è stata introdotta Regola Tecnica Verticale Scuole, denominata V7, la quale applica a edifici o locali adibiti ad attività scolastica con affollamento superiore a 100 persone, di qualsiasi tipo, ordine e grado, comprese quelle universitarie, disposizioni di prevenzione incendi. Per le scuole è dunque possibile l'applicazione della relativa RTV o, in alternativa, del Codice di Prevenzione Incendi.
Di seguito verranno analizzate le informazioni principali tratte dal Codice Prevenzione Incendi e dalla sezione V7.

Classificazioni delle attività scolastiche

La classificazione delle attività scolastiche avviene sia in relazione al numero degli occupanti (n) che alla massima quota dei piani (h) che contiene l'attività scolastica.

Classificazione in relazione al numero di occupanti n:

- **OA**: $100 < n \leq 300$;
- **OB**: $300 < n \leq 500$;
- **OC**: $500 < n \leq 800$;
- **OD**: $800 < n \leq 1200$;
- **OE**: $n > 1200$.

Classificazione in relazione alla massima quota dei piani h:

- **HA**: $h \leq 12$ m;

- **HB**: 12 m < h ≤ 24 m;
- **HC**: 24 m < h ≤ 32 m;
- **HD**: 32 m < h ≤ 54 m;
- **HE**: h > 54 m.

Classificazione delle aree dell'attività scolastica

La RTV Scuole V7 classifica inoltre le aree presenti all'interno degli edifici scolastici:

- **TA**: locali destinati ad attività didattica e spazi comuni;
- **TM**: depositi o archivi di superficie lorda maggiore a 25 m^2 e carico di incendio specifico q_f maggiore a 600 MJ/m^2;
- **TO**: locali con affollamento superiore a 100 persone;
- **TK**: locali pericolosi per incendio o esplosione o con carico di incendio specifico q_f maggiore a 1200 MJ/m^2;
- **TT**: locali con quantità significative di apparecchiature elettriche ed elettroniche, locali tecnici rilevanti ai fini della sicurezza antincendio;
- **TZ**: altre aree.

Profili di rischio

I profili di rischio sono determinati secondo la metodologia al capitolo G.3 del D.M. 03/08/2015, nel quale viene evidenziato il profilo di rischio vita R_{vita} per la tipologia della destinazione d'uso, quindi relativo a palestre scolastiche, aule scolastiche e laboratori scolastici.

Tipologie di destinazione d'uso	R_{vita}
Palestra scolastica	A1
Autorimessa privata	A2
Ufficio non aperto al pubblico, sala mensa, aula scolastica, sala riunioni aziendale, archivio, deposito librario, centro sportivo privato	A2-A3
Attività commerciale non aperta al pubblico (es. all'ingrosso, ...)	A2-A4
Laboratorio scolastico, sala server	A3

Figura 12. Tabella: G.3-5: Profilo di rischio Rvita per alcune tipologie di destinazione d'uso

Strategia antincendio

Devono essere applicate tutte le misure antincendio della RTO attribuendo i livelli di prestazione secondo i criteri definiti.

Per alcune misure antincendio vi sono indicazioni complementari o sostitutive alle soluzioni conformi previste in RTO:

- **Reazione al fuoco**: nelle vie d'esodo verticali, percorsi di esodo e spazi calmi devono essere impiegati materiali di reazione al fuoco appartenenti almeno al gruppo GM2 (cap. S.1). In questi ambienti è ammesso l'impiego di materiali del gruppo GM3 con l'incremento di un livello di prestazione delle misure richieste per il controllo dell'incendio (cap. S.6) e per la rivelazione e allarme (cap. S.7).
- **Resistenza al fuoco**: la classe di resistenza al fuoco dei compartimenti non può essere minore a quanto descritto nella seguente tabella:

Compartimenti	Attività				
	HA	HB	HC	HD	HE
Fuori terra	30	60			90
Interrati	60				90

Figura 13. Tabella V.7-1: Classe di resistenza al fuoco

- **Compartimentazione**: Le aree tipo TA, TO devono essere ubicate a quota ≥ -5 m. Le caratteristiche di compartimentazione sono riportate in tabella:

Area	Attività				
	HA	HB	HC	HD	HE
TA	Nessun requisito aggiuntivo				
TM, TO, TT	Di tipo protetto				
TK	Di tipo protetto [1]		Il resto dell'attività deve essere a prova di fumo proveniente dall'area TK		
TZ	Secondo risultanze della valutazione del rischio				
[1] Di tipo protetto se ubicate a quota ≥ -5 m; in caso l'area TK sia ubicata a quota < -5 m il resto dell'attività deve essere a prova di fumo proveniente dall'area TK.					

Figura 14. Tabella V.7-2: Compartimentazione

- **Gestione della sicurezza antincendio**: nelle aree TA e TO deve essere affissa cartellonistica indicante il massimo affollamento consentito; Nella attività in cui è richiesto il livello di prestazione I di rivelazione ed allarme

(cap. S.7), deve essere prevista una procedura gestionale di sorveglianza periodica delle aree TM e TK, se presenti.

- **Controllo dell'incendio:** sono forniti i livelli di prestazione:

Area	Attività				
	HA	HB	HC	HD	HE
TA, TM, TO, TT	II	III			
TK	III [1]		IV		
TZ	Secondo le risultanze della valutazione del rischio				
[1] Livello di prestazione IV qualora ubicati a quota < -5 m.					

Figura 15. Tabella V.7-3: Livelli di prestazione per controllo dell'incendio

Parametri progettuali per la rete idranti secondo UNI 10779:

Attività	Livello di pericolosità	Protezione esterna	Alimentazione idrica
OA, OB, OC	1	Non richiesta	Singola [3]
OD, OE	2 [2]	Sì [1]	Singola superiore
[1] Non richiesta per HA. [2] Per le eventuali aree TK presenti nella attività classificate HA, è richiesto almeno il livello di pericolosità 1. [3] È ammessa *alimentazione idrica di tipo promiscuo*.			

Figura 16. Tabella V.7-4: Parametri progettuali per rete idranti secondo UNI 10779 e caratteristiche minime

Parametri progettuali impianto sprinkler secondo UNI EN 12845:

Area	Alimentazione idrica
TK	Singola superiore [1]
[1] Per le eventuali aree TK inserite in attività OA, OB, OC alimentazione idrica di tipo singolo.	

Figura 17. Tabella V.7-5: Parametri progettuali impianto sprinkler e caratteristiche minime alimentazione idrica

- **Rivelazione ed allarme**: sono forniti i livelli di prestazione:

Attività	Attività				
	HA	HB	HC	HD	HE
OA	I [2]	II [1]	III		IV
OB	II [1]		III	IV	
OC	III		IV		
OD	III		IV		
OE	IV				

[1] Se presenti, le aree TM, TK e TT devono essere sorvegliate da rivelazione automatica d'incendio (funzione A, capitolo S.7)

[2] Il livello di prestazione I può essere garantito anche dallo stesso impianto a campanelli usato normalmente per l'attività scolastica, purché sia convenuto e codificato un particolare suono nella pianificazione di emergenza (capitolo S.5).

Figura 18. Tabella V.7-6: Livello di prestazione per rivelazione ed allarme

- **Vani degli ascensori**: dove sono previsti vani scala di tipo protetto o a prova di fumo, i vani degli ascensori devono essere almeno di tipo SB (protetti) se attraversano elementi orizzontali di compartimentazione.

3. SERIOUS GAME PER L'EMERGENZA

Le recenti innovazioni tecnologiche hanno consentito a vari settori di migliorare e progettare in modo più avanzato e ottimizzato. Grazie al passaggio dal CAD al BIM sono stati fatti grandi passi avanti, infatti il BIM ha reso la progettazione ancora più multidisciplinare e condivisa in tempo reale. La progettazione, con l'emergere della Virtual Reality, sta cercando di far utilizzare queste nuove tecniche nei vari settori.

I campi in cui la Virtual Reality può essere applicata sono molteplici e può essere impiegata in ogni settore dove fare qualcosa risulterebbe pericoloso, poco pratico o troppo costoso. Questi sono alcuni dei campi in cui la Virtual Reality viene applicata:

- Apprendimento (training): viene utilizzata per presentare il funzionamento di macchine e impianti, reso possibile da applicazioni immersive che simulano diverse situazioni dove l'utente deve prendere delle decisioni;
- Medicina: viene utilizzata dai medici per valutare le procedure da eseguire sul paziente e rilevare eventuali criticità;
- Architettura: viene utilizzata per la creazione di tour virtuali interattivi per valutare gli spazi e consente ai clienti di esplorare virtualmente la casa;
- Militare: viene utilizzata sia per le operazioni logistiche che per gli aspetti bellici. Vengono infatti create applicazioni per il training per dare la possibilità ai militari di allenarsi e addestrarsi;
- Turismo: viene utilizzata per la creazione di virtual tour di luoghi e strutture ricettive;
- Educazione: viene utilizzata pe creare esperienze di apprendimento immersive che si basano sul metodo *learning by doing*, più coinvolgente rispetto al tradizionale *learning by book*.

Risulta interessante l'applicazione della Virtual Reality nel campo della prevenzione incendi, dove i vigili del fuoco vengono formati e addestrati su come utilizzare attrezzature antincendio e possono essere immersi in scenari di emergenza specifici.

3.1. Stato dell’arte

Nell'ultimo decennio, l'importanza della gestione delle emergenze nelle infrastrutture pubbliche è aumentata a causa delle mutate condizioni di sicurezza in tutto il mondo, il che ha portato alla necessità di un processo di valutazione delle emergenze computerizzata. Quasi ogni giorno avvengono disastri naturali o incendi e quindi diventa importante rendere l'ambiente costruito il più sicuro possibile. In particolare, nel campo della sicurezza antincendio si devono affrontare molte sfide. La sfida per gli ingegneri civili non è più costruire nuovi edifici, poiché sempre più edifici esistenti vengono ristrutturati e riqualificati.

Gli ingegneri per la sicurezza antincendio fanno uso di modelli e simulazioni al computer per la descrizione della diffusione prevista di fuoco e fumo, l'evacuazione di sicurezza e l'analisi delle prestazioni di un edificio. In particolare, la stima del comportamento delle persone in pericolo è molto importante per analizzare l'evacuazione in sicurezza di un edificio, poiché la protezione della vita umana è l'obiettivo primario. Oltre al comportamento delle persone, vi sono altri fattori che influenzano l’evacuazione come i sistemi di allarme, gli elementi costruttivi e la diffusione di fuoco e fumo. In particolare, mappare i fattori umani su modelli computerizzati è una sfida, perché il comportamento singolare di ogni persona è basato su decisioni e parametri individuali e non è deterministico come la diffusione del fuoco e del fumo, che può essere modellato e simulato sulla base di principi naturali. Secondo i professori Santos e Aguirre, per una simulazione di evacuazione, devono essere considerate tre dimensioni: in primo luogo, l'ambiente costruito (posizione fisica), in secondo luogo, la gestione dell’ambiente (segnaletica, vie di fuga) e in terzo luogo le caratteristiche sociali, psicologiche e organizzative sociali degli occupanti. L’ingegnere Tavares, invece, ha affermato che un modello di simulazione di evacuazione deve considerare quattro interazioni: occupanti-struttura, occupanti-occupanti, occupanti-incendio e struttura antincendio.

I progressi nelle simulazioni includono l'uso di tecnologie come la Realtà Virtuale per una visualizzazione più realistica dell'ambiente pianificato. In uno studio,

Woksepp e Olofsson hanno analizzato la credibilità e l'applicabilità dei modelli di realtà virtuale per i team di progettazione e pianificazione e per il cantiere. Nell'ambito del loro studio un modello è stato convertito in un modello VR. È stato dimostrato che gli intervistati che utilizzano i modelli VR in cantiere li consideravano utili e potevano immaginare di utilizzarli nel loro lavoro quotidiano, in particolare per la gestione di compiti non familiari. Il secondo gruppo di questo studio era composto dal team di progettazione e pianificazione, che hanno utilizzato i modelli VR per esaminare soluzioni progettuali e i requisiti in materia di funzione, ambiente di lavoro e manutenzione. Un grande vantaggio che hanno scoperto è stato la maggiore comprensione del progetto generale e le conseguenze multidisciplinari di una decisione.

Trenholme e Smith, in un articolo, hanno presentato l'idea di utilizzare la tecnologia dei giochi per computer per costruire ambienti virtuali realistici. Hanno esaminato e confrontato diversi motori di gioco per computer per quanto riguarda la loro capacità di sviluppare ambienti virtuali in prima persona. Uno studio condotto in questo contesto di ricerca ha documentato che un singolo sviluppatore ha avuto bisogno di circa tre settimane per costruire un modello realistico di un edificio del mondo reale. In un edificio virtuale di questo tipo è possibile simulare scenari di evacuazione di esercitazioni antincendio. Un risultato di uno studio sugli utenti condotto in questo contesto è stato che i partecipanti hanno ritenuto realistico l'ambiente simulato, e si può quindi presumere che gli ambienti virtuali possano supportare la formazione e l'osservazione dei comportamenti degli utenti negli edifici virtuali 3D in caso di incendio.

Lo studio di approcci innovativi ed efficaci è importante, poiché in generale, le persone vengono formate e acquisiscono conoscenze sull'evacuazione attraverso approcci tradizionali come video, poster, seminari, corsi o esercitazioni di evacuazione. Tuttavia, questi approcci tradizionali potrebbero non trasmettere efficacemente la conoscenza. Uno dei motivi è che dopo un'esercitazione di evacuazione, agli occupanti dell'edificio non viene generalmente fornito un feedback individuale che valuti il loro comportamento di evacuazione. Un altro motivo è che gli occupanti dell'edificio possono non essere coinvolti emotivamente

nel processo di apprendimento. Per queste ragioni, le innovazioni in merito all'evacuazione antincendio dovrebbero trasmettere una maggiore conoscenza dell'evacuazione degli occupanti dell'edificio. A questo proposito, i Serious Games possono essere un modo efficace ed innovativo di apprendimento. Si sostiene che giocando a Serious Games, i partecipanti possano acquisire e conservare le conoscenze in modo più efficace rispetto all'utilizzo dei metodi di apprendimento tradizionali. I Serious Games possono migliorare la capacità degli approcci tradizionali di fornire conoscenze sull'evacuazione. È possibile poi combinare Serious Games e Virtual Reality, e questa unione incoraggia i partecipanti a conservare le conoscenze più a lungo rispetto agli approcci tradizionali poiché beneficiano del pieno coinvolgimento e dell'elevata eccitazione emotiva e fisiologica.

Nel campo della prevenzione incendi la combinazione di Serious Games e Virtual Reality viene utilizzata in diversi modi. Alcuni di questi sono:

- Formare le persone in situazioni di evacuazione e soccorso;
- Migliorare le capacità e il comportamento dei bambini durante l'evacuazione;
- Formare e allenare i vigili del fuoco;
- Scegliere il percorso migliore di salvataggio;
- Valutare il comportamento umano durante un incendio.

La Virtual Reality applicata alla prevenzione incendi rappresenta un valore aggiunto. Attualmente il principale vantaggio deriva dagli output ottenuti dalle simulazioni, poiché la realtà virtuale fornisce una migliore rappresentazione dei risultati sia per i vigili del fuoco che per il progettista stesso. Le software house del settore della prevenzione incendi stanno cercando di migliorare la visualizzazione degli output ottenuti tramite i dispositivi di VR in modo da garantire una migliore esperienza immersiva nell'ambiente simulato, in modo tale che questo possa essere verificato senza rischi. I software disponibili attualmente non consentono all'utente di immergersi completamente all'interno del modello, poiché non è possibile esplorare un edificio a 360°, ma si può solo creare un percorso definito e controllato o immedesimarsi in un occupante della simulazione.

Ad oggi sono alcune le proposte di sistemi di addestramento basati sul gioco che trattano argomenti di sicurezza antincendio, ma sono principalmente finalizzate alla formazione professionale dei vigili del fuoco. Un esempio di un ambiente di addestramento per vigili del fuoco che combina rappresentazioni di vigili del fuoco animati con una simulazione e animazione di fumo e fuoco è quello sviluppato presso la Georgia Tech in collaborazione con i vigili del fuoco di Atlanta. In questo Serious Game l'utente è un apprendista comandante che istruisce le squadre di vigili del fuoco virtuali a eseguire diverse azioni per aiutare a spegnere gli incendi virtuali. La corretta sequenza di comandi estinguerà con successo la fiamma con il minimo pericolo per i vigili del fuoco e il minor danno per la casa.

Un altro esempio è “Sidh”, un simulatore di addestramento per vigili del fuoco basato su un gioco sviluppato in collaborazione tra l'Università di Skövde e l'Agenzia svedese per i servizi di soccorso. Sidh è stato utilizzato in uno studio di fattibilità in cui 31 studenti vigili del fuoco hanno giocato al gioco e sono state analizzate le prestazioni di questi studenti e le loro riflessioni sull'uso del gioco. I risultati di questo studio mostrano che Sidh è un utile complemento ai metodi di allenamento tradizionali.

Non sono stati trovati Serious Games applicati all’esodo scolastico, dove l’utente è uno studente, e per questa ragione si è deciso di sviluppare un applicativo che comprendesse questa categoria finora non analizzata. Il Serious Game sviluppato serve inoltre per insegnare la sicurezza antincendio, per far immergere l’utente in scenari di emergenza antincendio, in cui l'obiettivo del gioco è sopravvivere e la sopravvivenza del giocatore dipende strettamente dalla scelta delle azioni giuste per evacuare l'edificio e impiegare meno tempo possibile per completare l'evacuazione, rimanendo il più lontano possibile dal pericolo. Per avere successo e progredire nel gioco, gli utenti dovrebbero migliorare il loro processo decisionale in situazioni di incendio, imparando a evitare gli errori comuni degli occupanti.

3.2. Metodologia

Gli attuali approcci per fornire alle persone le informazioni sulla sicurezza antincendio e sulle procedure di evacuazione tendono a basarsi su due soluzioni principali:

- Istruzioni scritte affisse sui muri (la planimetria e le linee guida per la sicurezza antincendi) e cartelli;
- Esercitazioni di evacuazione.

Le istruzioni scritte, affisse su porte e pareti non sono necessariamente lette dagli occupanti e, anche se lo sono, non vi è alcuna garanzia che una singola lettura consentirà all'occupante di ricordarle durante un'emergenza. Le esercitazioni di evacuazione, invece, tendono ad essere eseguite raramente, perché hanno un costo, comportano ore perse, e qualcuno potrebbe farsi male durante l'esercitazione.

Alcune proposte di sistemi di addestramento basati sul gioco trattano argomenti di sicurezza antincendio, ma sono finalizzate alla formazione dei vigili del fuoco, quindi le procedure insegnate sono inappropriate per l'utente preso in considerazione nel caso studio. Per insegnare le abilità personali di sicurezza antincendio, un Serious Game può immergere l'utente in scenari di emergenza antincendio, in cui l'obiettivo del gioco è sopravvivere all'incendio e la sopravvivenza del giocatore dipende dalla scelta delle azioni giuste per evacuare l'edificio e impiegare meno tempo possibile per completare l'evacuazione. Per avere successo e progredire nel gioco, gli utenti dovrebbero migliorare il loro processo decisionale in situazioni di incendio, imparando a evitare gli errori comuni. Un Serious Game potrebbe essere una soluzione efficace per motivare le persone a formarsi sulla sicurezza antincendio.

Il gioco sviluppato mira a riprodurre situazioni in cui potrebbero trovarsi gli occupanti dell'edificio scolastico, nello specifico gli studenti, in caso di emergenze incendi. Di seguito, sono illustrate le principali scelte di progettazione che sono state fatte. Durante la progettazione del gioco di evacuazione tre sono stati gli obiettivi principali:

- immergere i giocatori in scenari che riproducessero l'esperienza di trovarsi in un incendio;
- creare coinvolgimento e motivazione per giocare;
- insegnare le abilità di sicurezza antincendio, sia generali che specifiche dell'edificio.

Rispetto al primo obiettivo, una scelta è stata quella di utilizzare la grafica 3D con visuale in prima persona. Nello specifico, il gioco riproduce:

- la configurazione spaziale e l'aspetto dell'edificio scolastico;
- fenomeni e oggetti correlati all'emergenza incendio e all'evacuazione, come fumo, uscite di emergenza, segnaletica, allarmi acustici;
- oggetti comuni come libri, che sono interattivi, ovvero possono essere raccolti.

Rispetto all'obiettivo di coinvolgimento e motivazione, è stato deciso di:

- presentare al giocatore una situazione di emergenza antincendio e il suo obiettivo è di evacuare l'edificio seguendo i segnali di emergenza e di eseguire tutte le azioni necessarie, evitando allo stesso tempo azioni errate (come prendere un libro, allontanarsi);
- introdurre un timer che tiene conto del tempo impiegato per l'evacuazione e delle azioni giuste/sbagliate compiute nel gioco per fornire un'autovalutazione del livello di abilità raggiunto.

4. CASO APPLICATIVO

In questo capitolo viene descritto il caso studio e lo sviluppo dell'applicativo. Il caso studio riguarda l'Istituto Comprensivo "Torino II".

Figura 19. Scuola primaria "Aurora" e la scuola secondaria di I grado "Ettore Morelli"

4.1. Plesso scolastico Ettore Morelli

L'Istituto comprende tre ordini di scuola, tra i quali troviamo una scuola dell'infanzia, una primaria e una secondaria di I grado. Per il caso studio sono state prese in esame la scuola primaria Aurora e la scuola secondaria di I grado Ettore Morelli, entrambe situate in via Antonio Cecchi 16-18 nel quartiere Aurora di Torino. I due plessi sono collocati in un complesso die tre edifici comunicanti tramite due corpi scala. L'edificio, realizzato negli anni '70, ha una struttura a telaio in calcestruzzo armato, un involucro verticale in muratura che alterna mattoni faccia a vista a muratura in mattoni rivestiti da pannelli in alluminio e coperture a più falde con pendenze diverse. Il complesso si sviluppa su tre piani fuori terra ed uno interrato ed è composto da tre corpi principali collegati da due corpi scala che

fungono anche da ingresso. Nel fabbricato sono inoltre presenti una palestra e una piscina, che non sono però di competenza delle scuole. A seguire sono riportare le piante fuori scala rappresentanti i tre corpi principali, denominati C.1, C.2 e C.3, e i corpi scala, denominati A, C.B, C.C e D.

Le classi evidenziate nelle immagini a seguire, rispettivamente la classe della scuola primaria 2B situata al primo piano e la classe della scuola secondaria di I grado 1F situata al secondo piano, saranno le classi prese in esame per lo sviluppo dell'applicativo, in modo che ci sia una differenziazione del livello scolastico e per comprendere al meglio il comportamento di alunni con età differenti.

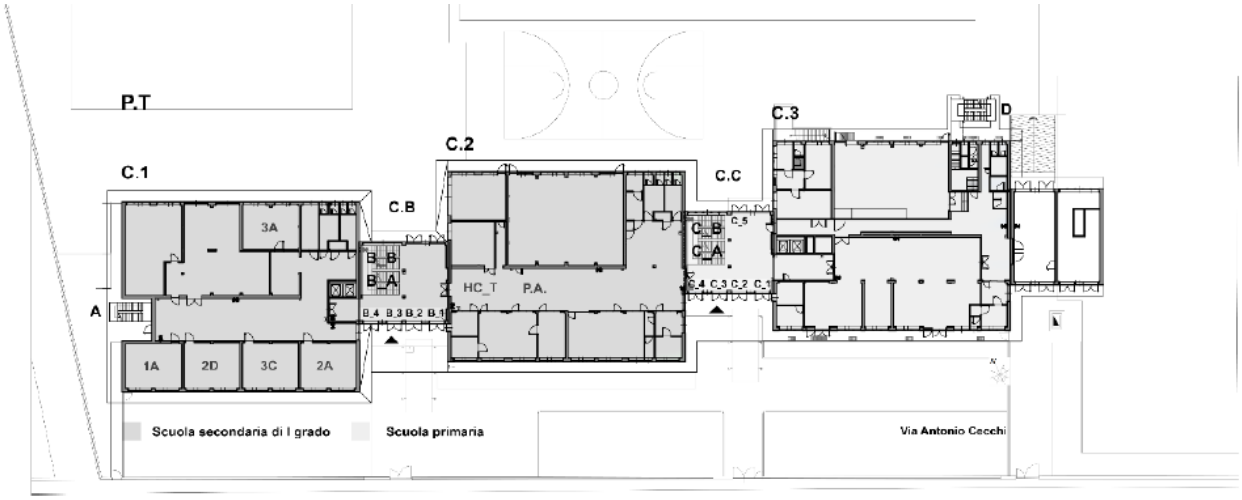


Figura 20. Pianta piano terra

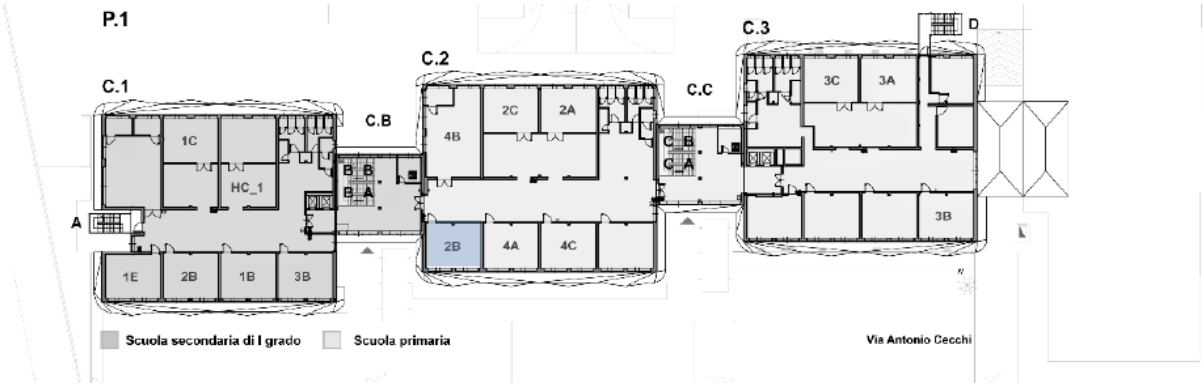


Figura 21. Pianta primo piano, con classe 2B presa in esame evidenziata

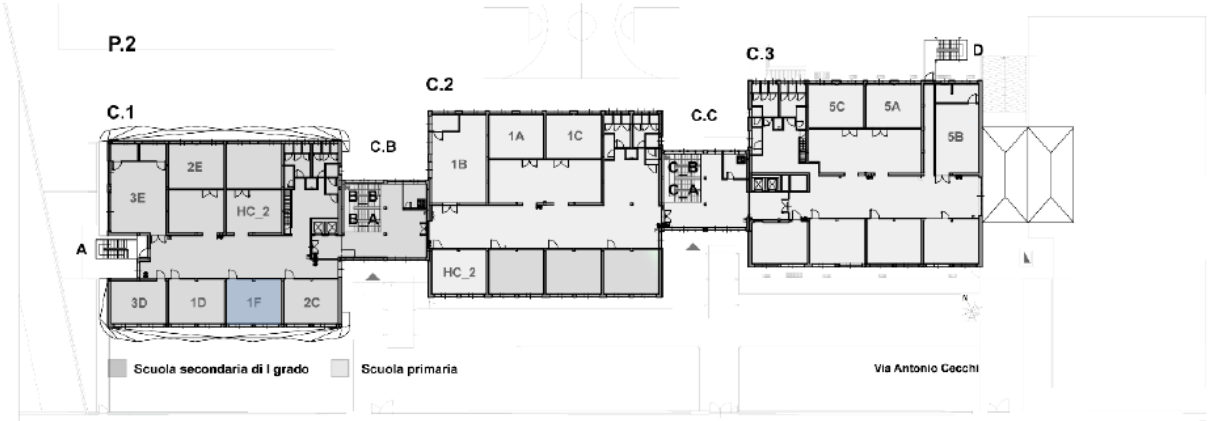


Figura 22. Pianta secondo piano, con classe 1F presa in esame evidenziata

La destinazione d'uso degli spazi interni della scuola primaria è così costituita:

- 15 aule;
- 2 laboratori informatici;
- 1 laboratorio linguistico;
- 1 laboratorio musicale;
- 1 laboratorio scientifico;
- 1 laboratorio audiovisivo;
- 1 laboratorio teatrale – psicomotricità;
- 1 laboratorio artistico – grafico;
- 1 laboratorio ludico – manuale;
- 1 Aula Magna;
- 1 spazio, con relativi servizi, adibito a mensa.

La scuola secondaria, invece, è così costituita:

- 16 aule;
- 1 laboratorio musicale;
- 1 laboratorio audiovisivo;
- 1 laboratorio informatico;
- Vari spazi multifunzionali;
- 1 Aula Magna;
- Spazi adibiti a mensa, con relativi servizi.

Il modello BIM (Building Information Modeling) della scuola, già esistente e realizzato con il software Revit, è stato implementato con l'aggiunta di alcuni arredi delle aule.

A seguire sono stati riportati i layout di esodo per ogni piano, che sono stati necessari per lo sviluppo del videogioco per l'impostazione del percorso che le classi devono seguire durante l'evacuazione. Vengono inoltre inserite le tabelle che mostrano le scale e le uscite che le classi devono seguire per l'evacuazione.

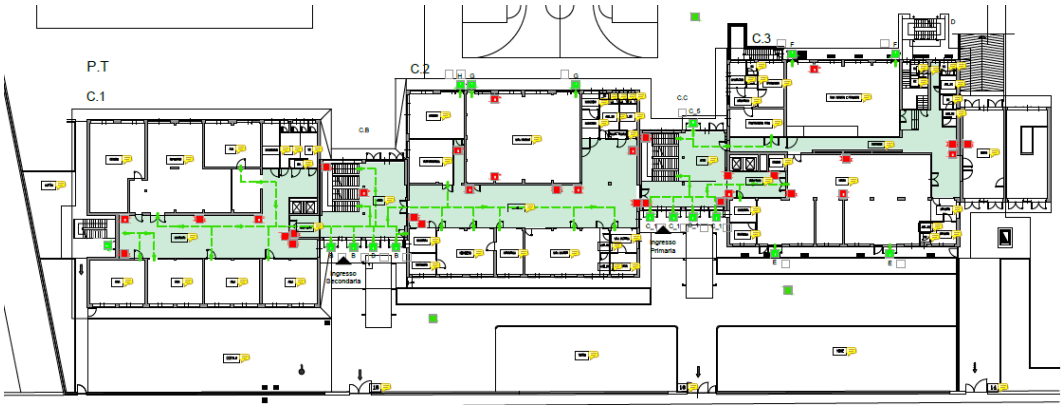


Figura 24. Layout di Esodo - Piano Terra

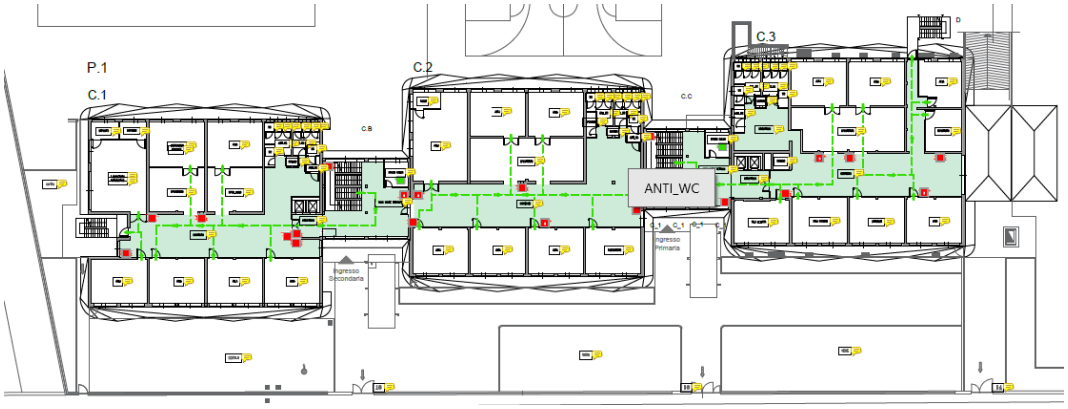


Figura 25. Layout di Esodo - Primo Piano

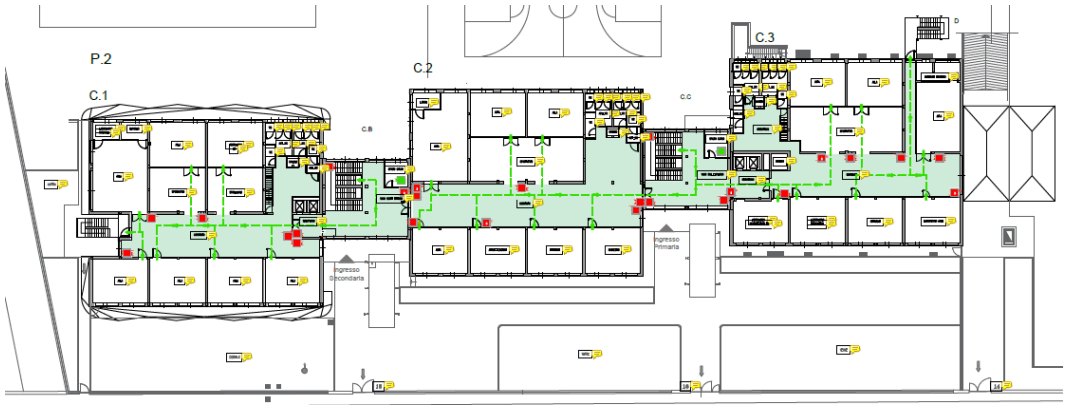


Figura 23. Layout di Esodo - Secondo Piano

Scale e uscite assegnate - Scuola primaria						
Classe	Piano	Corpo	n. studenti	n. docenti	n. ATA	Scale
1A	T	C.3	23	3	-	-
1B	2	C.2	18	1	-	C_A
1C	2	C.2	18	1	-	C_A
2A	1	C.2	15	1	-	C_A
2B	1	C.2	17	2	-	C_A
2C	1	C.2	17	2	-	C_A
3A	1	C.3	18	1	-	D
3B	1	C.3	21	2	-	D
3C	1	C.3	19	2	-	C_B
4A	1	C.2	21	1	-	C_B
4B	1	C.2	16	1	-	C_B
HC_4B	1	C.2	1	1	-	-
4C	1	C.2	18	1	-	C_B
5A	2	C.3	18	2	-	D
5B	2	C.3	21	1	-	D
5C	2	C.3	20	2	-	C_B
HC_2	2	C.2	1	1	-	C_B
HC_2A	2	C.2	-	1	-	C_B
HC_2B	2	C.2	1	-	-	C_B
HC_TA	T	C.C	1	-	-	-
HC_TB	T	C.C	-	1	-	
ATA1	T	C.C	-	-	1	-
ATA2	T	C.C	-	-	1	-

Figura 26. Scale e uscite assegnate - Scuola primaria

Scale e uscite assegnate - Scuola secondaria di I grado						
Classe	Piano	Corpo	n. studenti	n. docenti	n. ATA	Scale
1A	T	C.1	17	2	-	-
1B	1	C.1	17	2	-	B_A
1C	1	C.1	16	2	-	A
1D	2	C.1	15	1	-	A
1E	1	C.1	13	1	-	A
1F	2	C.1	13	1	-	B_B
2A	T	C.1	15	1	-	-
2B	1	C.1	13	2	-	A
2C	2	C.1	11	1	-	B_A
2D	T	C.1	20	1	-	-
2E	2	C.1	16	1	-	A
3A	1	C.1	15	2	-	B_A B_A
3B	1	C.1	13	1	-	B_A
3C	T	C.1	19	1	-	-
3D	2	C.1	18	1	-	A
3E	2	C.1	22	1	-	A
ATA2	T	C.1	-	-	1	-
P.A.	T	C.1	-	5	-	-
HC	T	C.B	1	1	-	-
HC	1	C.1	5	1	-	B_A
HC	2	C.1	1	1	-	-
ATA1	2	C.1	-	-	1	-

Figura 27. Scale e uscite assegnate - Scuola secondaria di I grado

4.2. Progettazione dell'esodo

Per il caso studio è importante prestare attenzione al capitolo S.4 del Codice Prevenzione Incendi, poiché riguarda il massimo affollamento consentito e la progettazione dell'esodo. La finalità del sistema d'esodo è di assicurare che gli occupanti dell'attività possano raggiungere un luogo sicuro o permanere al sicuro a prescindere dall'intervento dei Vigili del fuoco.

- S.4.3 – Criteri di attribuzione dei livelli di prestazione

Le seguenti tabelle riportano i livelli di prestazione e i criteri di attribuzione riferiti agli ambiti dell'attività per la presente misura antincendio.

Livello di prestazione	Descrizione
I	Gli occupanti raggiungono un *luogo sicuro* prima che l'incendio determini condizioni incapacitanti negli ambiti dell'attività attraversati durante l'esodo.
II	Gli occupanti sono protetti dagli effetti dell'incendio nel luogo in cui si trovano.

Figura 28. Tabella S.4-1: Livelli di prestazione

Livello di prestazione	Criteri di attribuzione
I	Tutte le attività
II	Ambiti per i quali non sia possibile assicurare il livello di prestazione I (es. a causa di dimensione, ubicazione, abilità degli occupanti, tipologia dell'attività, caratteristiche geometriche particolari, vincoli architettonici, ...)

Figura 29. Tabella S.4-2: Criteri di attribuzione dei livelli di prestazione

- S.4.6.1 – Profilo di rischio: R_{VITA}

Il profilo di rischio R_{vita} è associato ad ogni compartimento dell'attività e si è fatto riferimento a quanto riportato nel paragrafo G.3.2 del Codice. Il profilo di rischio R_{vita} è attribuito in relazione ai seguenti fattori:

  - δ_{occ}: caratteristiche prevalenti degli occupanti;
  - δ_{α}: velocità caratteristica prevalente di crescita dell'incendio, riferita al tempo t_{α} in secondi, impiegato dalla potenza termica per raggiungere il valore di 1000 kW.

A seguire sono riportate le tabelle riportanti la classificazione dei fattori sopra descritti.

Caratteristiche prevalenti degli occupanti δ_{occ}		Esempi
A	Gli occupanti sono in stato di veglia ed hanno familiarità con l'edificio	Ufficio non aperto al pubblico, scuola, autorimessa privata, centro sportivo privato, attività produttive in genere, depositi, capannoni industriali
B	Gli occupanti sono in stato di veglia e non hanno familiarità con l'edificio	Attività commerciale, autorimessa pubblica, attività espositiva e di pubblico spettacolo, centro congressi, ufficio aperto al pubblico, ristorante, studio medico, ambulatorio medico, centro sportivo pubblico
C	Gli occupanti possono essere addormentati: [1]	
Ci	• in attività individuale di lunga durata	Civile abitazione
Cii	• in attività gestita di lunga durata	Dormitorio, residence, studentato, residenza per persone autosufficienti
Ciii	• in attività gestita di breve durata	Albergo, rifugio alpino
D	Gli occupanti ricevono cure mediche	Degenza ospedaliera, terapia intensiva, sala operatoria, residenza per persone non autosufficienti e con assistenza sanitaria
E	Occupanti in transito	Stazione ferroviaria, aeroporto, stazione metropolitana
[1] Quando nel presente documento si usa C la relativa indicazione è valida per Ci, Cii, Ciii		

Figura 30. Tabella G.3-1: Caratteristiche prevalenti degli occupanti

δ_α	t_α [1]	Criteri
1	600 s lenta	Ambiti di attività con carico di incendio specifico $q_f \leq 200$ MJ/m^2, oppure ove siano presenti prevalentemente materiali o altri combustibili che contribuiscono in modo trascurabile all'incendio.
2	300 s media	Ambiti di attività ove siano presenti prevalentemente materiali o altri combustibili che contribuiscono in modo moderato all'incendio.
3	150 s rapida	Ambiti con presenza di significative quantità di materiali plastici impilati, prodotti tessili sintetici, apparecchiature elettriche e elettroniche, materiali combustibili non classificati per reazione al fuoco (capitolo S.1). Ambiti ove avvenga impilamento verticale di significative quantità di materiali combustibili con $3{,}0\ m < h \leq 5{,}0\ m$ [2]. Stoccaggi classificati HHS3 oppure attività classificate HHP1, secondo la norma UNI EN 12845. Ambiti con impianti tecnologici o di processo che impiegano significative quantità di materiali combustibili. Ambiti con contemporanea presenza di materiali combustibili e lavorazioni pericolose ai fini dell'incendio.
4	75 s ultra-rapida	Ambiti ove avvenga impilamento verticale di significative quantità di materiali combustibili con $h > 5{,}0\ m$ [2]. Stoccaggi classificati HHS4 oppure attività classificate HHP2, HHP3 o HHP4, secondo la norma UNI EN 12845. Ambiti ove siano presenti o in lavorazione significative quantità di sostanze o miscele pericolose ai fini dell'incendio, oppure materiali plastici cellulari/espansi o schiume combustibili non classificati per la reazione al fuoco.
A meno di valutazioni più approfondite da parte del progettista (es. dati di letteratura, misure dirette, ...), si ritengono *non significative* ai fini della presente classificazione almeno le quantità di materiali nei compartimenti con carico di incendio specifico $q_f \leq 200$ MJ/m^2. [1] Velocità caratteristica prevalente di crescita dell'incendio. [2] Con h altezza d'impilamento.		

Figura 31. Tabella G.3-2: Velocità caratteristica prevalente di crescita dell'incendio

Poiché il profilo di rischio R_{vita} si determina per compartimenti, è stata fatta una tabella dei profili assegnati ai compartimenti presenti nell'edificio.

Profilo di rischio R_{vita}			
Compartimento	δ_{occ}	δ_{α}	R_{vita}
Piano interrato	A	2	A2
C.1 - Corpo 1	A	2	A2
C.2 - Corpo 2	A	2	A2
C.3 - Corpo 3	A	2	A2
C.B - Scale interne 1	A	1	A1
C.C - Scale interne 2	A	1	A1
Archivio	A	2	A2
Sala riunioni	A	2	A2
Centrale termica	A	3	A3

Figura 32. Profilo di rischio R_{vita}

- S.4.6.2 – Affollamento

L'affollamento massimo di ciascun locale è determinato:

a. moltiplicando la densità di affollamento della tabella S.4-12 per la superficie lorda del locale stesso;
b. impiegando i criteri della tabella S.4-13;
c. secondo le indicazioni della regola tecnica verticale.

Nella tabella sono evidenziati gli ambiti adibiti ad attività scolastica e laboratori per le quali si ha una densità di affollamento pari a 0,4 persone/m^2 e gli ambiti con posti a sedere o posti letto per le quali si adotta invece il criterio del numero di posti. La somma del numero di posti a sedere per aula e dei metri quadri delle altre aree, moltiplicati per la densità, fornisce il valore di affollamento massimo da prendere in considerazione nella progettazione e nella pianificazione dell'esodo.

Tipologia di attività	Densità di affollamento
Ambiti all'aperto destinati ad attività di spettacolo o intrattenimento, delimitati e privi di posti a sedere	2,0 persone/m²
Locali al chiuso di spettacolo o intrattenimento (es. sale concerti, trattenimenti danzanti, …) privi di posti a sedere e di arredi, con carico di incendio specifico $q_f \leq 50$ MJ/m²	
Ambiti per mostre, esposizioni	1,2 persone/m²
Ambiti destinati ad attività di spettacolo o intrattenimento (es. sale concerti, trattenimenti danzanti, …) con presenza di arredi o con carico di incendio specifico $q_f > 50$ MJ/m²	
Ambiti adibiti a ristorazione	0,7 persone/m²
Ambiti adibiti ad attività scolastica e laboratori (senza posti a sedere)	0,4 persone/m²
Sale d'attesa	
Uffici	
Ambiti di vendita di *piccole* attività commerciali al dettaglio con settore alimentare o misto	
Ambiti di vendita di *medie* e *grandi* attività commerciali al dettaglio con settore alimentare o misto	0,2 persone/m²
Ambiti di vendita di attività commerciali al dettaglio senza settore alimentare	
Sale di lettura di biblioteche, archivi	
Ambulatori	0,1 persone/m²
Ambiti di vendita di attività commerciali all'ingrosso	
Ambiti di vendita di *piccole* attività commerciali al dettaglio con specifica gamma merceologica non alimentare	
Civile abitazione	0,05 persone/m²

Figura 33. Tabella S.4-12: Densità di affollamento per tipologia di attività

Tipologia di attività	Criteri
Autorimesse pubbliche	2 persone per veicolo parcato
Autorimesse private	1 persona per veicolo parcato
Degenza	1 degente e 2 accompagnatori per posto letto + addetti
Ambiti con posti a sedere o posti letto (es. sale riunioni, aule scolastiche, dormitori, …)	Numero posti + addetti
Altri ambiti	Numero massimo presenti (addetti + pubblico)

Figura 34. Tabella S.4-13: Criteri per tipologia di attività

A seguire sono state inserite le tabelle dell'affollamento per piano e per corpo dell'edificio.

Affollamento - Scuola primaria						
Piano	Corpo	Locale	Superficie [m^2]	Densità di affollamento [pers/m2]	Affollamento	Affollamento totale per piano e corpo
P.T	C.3	Guardiola	11,4	0,1	2	275
		Guardiola	10	0,1	1	
		Mensa	140	0,7	98	
		Sala riunioni	142	1,2	171	
		Prep. Pasti	28,2	0,1	3	
P.1	C.2	Aula	-	n. posti	26	149
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Laboratorio	46	0,4	19	
	C.3	Aula	-	n. posti	26	138
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Religione	46	0,4	18	
		Sala docenti	46	0,1	5	
		Sala lettura	46	0,2	10	
P.2	C.2	Aula	-	n. posti	26	116
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Alfabetizzazione	46	0,4	19	
		Religione	46	0,4	19	
	C.3	Aula	-	n. posti	26	153
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Lab. Musica	43	0,4	18	
		lab. Informatica	46	0,4	19	
		Religione	46	0,4	19	
		Lab. Arte	46	0,4	19	

Figura 35. Affollamento scuola primaria

Affollamento - Scuola secondaria di I grado						
Piano	Corpo	Locale	Superficie [m^2]	Densità di affollamento [pers/m2]	Affollamento	Affollamento totale per piano e corpo
P.T	C.1	Aula	-	n. posti	26	218.606
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Refettorio	126,6	0,7	88.606	
	C.2	Guardiola	11,5	0,1	2	100
		Segreteria	48,4	0,1	5	
		Segreteria	9,9	0,1	1	
		Presidenza	22,2	0,1	3	
		Sala professori	63,7	0,1	7	
		Infermeria	10,4	0,1	2	
		Vicepresidenza	35,5	0,1	4	
		Sala riunioni	188,6	0,4	76	
P.1	C.1	Aula	-	n. posti	26	179
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Lab. Informatica	73	0,4	30	
		Lab. Arte	46,4	0,4	19	
P.2	C.1	Aula	-	n. posti	26	175
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Aula	-	n. posti	26	
		Lab. HC	46,4	0,4	19	

Figura 36. Affollamento scuola secondaria di I grado

Caratterizzazione degli occupanti

Al fine di assicurare l'esodo degli occupanti da un edificio è importante capirne il loro comportamento e il movimento rispetto agli altri, poiché i singoli individui, in situazioni di pericolo, si comportano in modo differente. A seguire verranno analizzate le caratteristiche del singolo individuo e i comportamenti collettivi che emergono in situazioni di emergenza. Verranno inoltre individuati i profili e la velocità degli occupanti.

Caratteristiche del singolo individuo

Per queste ragioni è necessario considerare i vari attributi che caratterizzano l'occupante, che possono essere riassunti come segue:

- Genere: in caso di pericolo il comportamento tra uomo e donna presenta delle differenza, in quanto l'uomo è più propenso a cercare di spegnere l'incendio, mentre la donna cerca di fuggire;
- Età: in base all'età di una persona è possibile valutarne le prestazioni, usando tre categorie: abilità sensoriali, decisionali e di azione. È evidente che le persone anziane, ad esempio, sono meno resistenti al fumo e al calore e sono quindi più a rischio;
- Familiarità: in caso di emergenza le persone cercano luoghi e persone familiari. Molto spesso gli occupanti durante l'esodo seguono la strada che conoscono per uscire dall'edificio invece del percorso segnato dal piano antincendio;
- Attaccamento sociale: questo fattore rappresenta un pericolo in caso di emergenza poiché rallenta l'evacuazione. Alcuni individui non pensano immediatamente ad evacuare l'edificio, bensì raggiungono i propri cari per aiutarli;
- Attaccamento agli oggetti: come per l'attaccamento sociale anche quello agli oggetti risulta pericoloso, poiché alcune persone tendono a recuperare i propri oggetti, anche se ciò rallenta la loro fuga.

Pertanto, di fronte a un pericolo di incendio, ogni individuo reagisce in modo diverso a seconda delle caratteristiche personali e delle condizioni sociali e ambientali che incoraggiano determinati comportamenti. Conoscere i fattori umani in queste situazioni può migliorare le operazioni di soccorso e garantire la sicurezza di tutte le persone coinvolte.

Caratteristiche collettive

Lo sviluppo di una situazione di emergenza è il risultato di un confronto sociale, in cui le reazioni degli altri individui vengono osservate e interpretate per prendere una decisione. Il comportamento individuale si trasforma in comportamento collettivo quando l'attività del singolo individuo è influenzata dai suoi vicini e il comportamento di tutti si modifica verso un modello comune.
Negli anni '60, gli esperimenti di due psicologi, Latané e Darley, hanno sottolineato questo concetto attraverso un esperimento del fumo in una stanza. Questo esperimento consisteva nel chiudere dei soggetti in una stanza con il pretesto di rispondere a un questionario, pochi minuti dopo da una feritoia veniva fatto uscire del fumo e in questo modo veniva studiato il comportamento dei soggetti. L'esperimento ha avuto i seguenti risultati:

- Se i soggetti erano soli nella stanza, il 63% si accorge del fumo cinque secondi dopo la sua emissione;
- Se i soggetti erano in presenza di altre persone, solo il 26% si accorge del fumo cinque secondi dopo la sua emissione.

Pertanto, questo esperimento evidenzia che la presenza di altre persone non solo inibisce l'esame dell'ambiente, ma anche ritarda la consapevolezza che stia accadendo qualcosa di insolito.
Per questo motivo è necessario conoscere come si comporta la folla in caso di pericolo. Di seguito sono riportati alcuni comportamenti:

- **Comportamenti gregari**: in una situazione di emergenza, durante la fuga, le persone possono avere comportamenti individualistici o gregari. In caso di emergenza, le persone si muovono più velocemente, spingendosi a

vicenda, creando in questo modo code vicino alle uscite, questo perché hanno paura per la loro sopravvivenza. L'aumento della velocità è un fattore negativo, poiché le uscite si congestionano, e questo effetto paradossale è definito come faster is slower. Inoltre, le persone possono cadere, ferirsi o essere calpestate, diventando degli ostacoli.

- **Comportamenti pro-sociali**: in caso di emergenza spesso si crea cooperazione tra gli occupanti. Infatti, alcuni individui diventano altruisti e collaborative, mostrando persino capacità di leadership che li rendono efficaci durante l'evacuazione.
- **Panico**: il concetto di panico ha subito diverse trasformazioni nel corso della storia. Inizialmente si pensava che le persone in emergenza perdessero la loro umanità e diventassero animali per paura. Negli anni '50, Quarantelli classificò il panico come un comportamento asociale: le persone non si comportano come animali, ma cercano di soddisfare i propri bisogni e non si preoccupano dei bisogni degli altri. Ricerche successive hanno definito le reazioni di panico come: *comportamento collettivo in cui le capacità di giudizio e di ragionamento sono deteriorate, in cui vi sono emozioni forti di paura e in cui vi è comportamento (solitamente fuga) che può risultare in azioni autodistruttive ed eterodistruttive*. In particolare, gli studi psicosociali sul comportamento di evacuazione hanno dimostrato che devono essere presenti contemporaneamente diverse condizioni perché si verifichi il panico:

  · L'ansia è diffusa nel periodo precedente a un disastro, ad esempio come premonizione di un possibile pericolo reale o di informazioni provenienti da fonti autorevoli;
  · Mancanza di una guida in grado di dare istruzioni chiare;
  · Percezione che si è in trappola e che l'unica via di fuga è bloccata;
  · Comparsa di fattori precipitanti d'ansia.

  Pertanto, anche se il termine "panico di massa" viene spesso utilizzato, è necessario sottolineare che questo comportamento distruttivo si verifica

solo in casi eccezionali e in combinazione con i quattro fattori sopra menzionati.

- **Comportamenti affiliativo**: questo modello, sviluppato da Sime, evidenzia come in alcuni casi i legami di gruppo contribuiscano ad aumentare i tassi di morte e di lesioni nelle situazioni di emergenza. Sime ha intervistato 500 sopravvissuti a un incendio in un sito turistico in Gran Bretagna, in cui sono morte 50 persone, e ha osservato che il 73% è fuggito con uno o più persone del proprio gruppo. Un altro studio è stato condotto da Proulx nel 1995, che ha analizzato i registri di evacuazione dei complessi residenziali e ha notato che le famiglie con bambini di solito evacuano in gruppo, con un adulto che tiene il bambino più piccolo in braccio. Proulx ha anche osservato che i gruppi di occupanti adottano la velocità dei membri più lenti, come gli anziani e i bambini, e questo porta a un netto ritardo nell'evacuazione.
- **Cluster di convergenza**: è stato osservato che i gruppi di persone spesso convergono verso un punto comune percepito come un luogo sicuro. Di solito si sceglie un luogo con condizioni migliori, come un locale per non fumatori con una buona visibilità e un balcone per facilitare la ventilazione, e in questo modo gli occupanti percepiscono una riduzione dell'ansia e della tensione.
- **Tendenza al rientro**: diversi studi hanno evidenziato come alcune persone che fuggono da un incendio spesso scelgono di rientrare nell'edificio. Uno di questi studi è quello relativo l'incendio di Arundel Park, in cui si evidenziò che un terzo delle persone che si erano messe in salvo è poi rientrato per cercare di salvare i propri cari e di recuperare i propri beni.

Individuazione dei profili

Come si può notare, nei paragrafi precedenti sono state analizzate caratteristiche che si riferiscono a uno spaccato ristretto della società, in quanto non sono state prese in considerazione persone con ridotto livello di autonomia motoria o di capacità cognitive. In Italia, il mondo del design e della progettazione, per lungo tempo non ha preso in considerazione il mondo delle disabilità. Infatti, si può notare

come molti luoghi sono inadeguati e possono creare notevoli disagi a persone che non sono completamente autonome, e questo porta all'esclusione di questa parte di società dalle attività sociali e non. È solo nel 1989 che si inizia a progettare in modo inclusivo e accessibile, tenendo in considerazione le persone disabili. Nell'ultimo decennio si sta assistendo ad una educazione della popolazione rispetto a questi temi e si ha una maggiore sensibilità. Nel D.M. 03/08/2015, il Codice di Prevenzione Incendi, è stato inserito per la priva volta il tema dell'inclusione nel campo delle normative antincendio. Secondo il Codice di Prevenzione Incendi con inclusione si intende che *le diverse disabilità (es. fisiche, mentali o sensoriali) e le specifiche necessità temporanee o permanenti degli occupanti sono considerate parte integrante della progettazione della sicurezza antincendio*. L'affermarsi di questa nuova filosofia nella progettazione ha fatto emergere la necessità di conoscere tutti gli aspetti delle persone - comportamentali, fisici e geometrici – in modo tale da poter validare un progetto sulla base di una società eterogenea. In primo luogo, si è cercato di individuare tutte le casistiche e caratteristiche degli occupanti, analizzando le questioni relative alla disabilità e all'inclusione. Per questo si è fatto riferimento alla guida DARAC (Disability Access Review and Advisory Committee) "Emergency Evacuation Planning Guide for People with Disabilities" (NFPA DARAC), che definisce categorie di disabilità più precise. L'obiettivo di questa guida è quello di identificare le esigenze e le problematiche della comunità di disabili, al fine di implementare queste necessità all'interno delle guide NFPA. La guida, pubblicata nel 2016, contiene le disposizioni per gli utenti con disabilità ed è stata sviluppata insieme alla comunità di disabili. All'interno della guida sono presenti cinque categorie di disabilità:

- **Mobilità**: le persone con disabilità motorie possono utilizzare uno o più dispositivi per muoversi, come bastoni, stampelle, deambulatori o sedie a rotelle. Le persone che utilizzano tali dispositivi hanno alcuni dei problemi di accesso/uscita più evidenti;
- **Cecità o ipovedenza**: le persone non vedenti o ipovedenti di solito si affidano al tatto e all'udito per percepire l'ambiente circostante, di solito con l'aiuto di un bastone o accompagnati da un cane guida. In situazioni di emergenza, è necessaria una segnaletica alternativa a quella visiva;

- **Udito**: le persone non udenti possono indossare apparecchi acustici che amplificano e chiariscono i suoni e spesso utilizzano una combinazione di lettura labiale e linguaggio dei segni. In caso di emergenza, è necessario un sistema alternativo oltre a segnali acustici;
- **Linguaggio**: le persone con problemi di linguaggio non possono parlare chiaramente o comunicare verbalmente con gli altri occupanti. In genere, questi residenti sono in grado di muoversi da soli e di capire cosa succede nell'ambiente circostante;
- **Cognitivo**: le persone con disabilità cognitive non possono utilizzare o accedere alle funzioni dell'edificio perché non sono in grado di elaborare o comprendere le informazioni necessarie per utilizzarle. Poiché tutti i sistemi di evacuazione richiedono che le persone elaborino e comprendano le informazioni per poter evacuare in sicurezza, questi profili di disabilità sono più problematici in quanto non sono autonomi in nessuna fase del processo di evacuazione.

Oltre agli occupanti con disabilità permanenti o a lungo termine, ce ne sono altri affetti da condizioni temporanee che compromettono le proprie abilità. Individui con fratture, traumi e malattie possono essere compromessi nell'uso e nel movimento dell'ambiente per un breve periodo. Altre condizioni includono malattie del cuore, neurologiche e dei polmoni, che possono ridurre la resistenza fisica. Inoltre, le persone di peso estremi hanno spesso una riduzione dell'autonomia. Non è raro, inoltre, che le persone possano essere affette da disabilità multiple. Per queste ragioni è importante tenere in considerazione tutti questi profili nella progettazione dell'evacuazione in situazioni di emergenza.

Velocità occupanti

Al fine di determinare la velocità degli occupanti in caso di emergenza sono stati analizzati gli studi esistenti in merito alla tematica. La norma ISO/TR 16738:2009 contiene alcuni dei valori delle velocità per le tipologie più comuni di occupanti, tuttavia, quando si analizzano determinate situazioni, come ad esempio gli ospedali, la norma non fornisce tutti i dati necessari. Pertanto, trovare dati da documenti

scientifici e analisi altrettanto validi è compito del progettista. Il primo studio preso in considerazione è quello di Nelson e Mowrer, che nelle loro analisi forniscono un valore di velocità pari a 1,19 m/s su superfici orizzontali, mentre può variare in caso di esodo verticale in base al rapporto tra alzata e pedata. Questo valore si riferisce solo ad occupanti adulti, mentre non è stato trovato alcun riferimento nella norma inerente la velocità di occupanti con età compresa 6 e 14 anni, dato fondamentale per il caso studio, e per questa ragione sono stati analizzati altri studi che riportassero valori affini al caso studio.

Gli studi per determinare la velocità per ogni tipo di occupante è in continua evoluzione. La quantità di dati attualmente disponibili è modesta, tuttavia, molto spesso questi dati sono oggetto di dibattito all'interno della comunità scientifica poiché di difficile interpretazione. Questo accade poiché ci sono infiniti modi in cui, nei vari studi, vengono osservate le prove e misurate le relative velocità. Pertanto, oggi è molto difficile standardizzare questi valori in modo che ogni profilo di occupante abbia un valore unico.

Uno dei pochi studi che analizza la velocità dei bambini è quello condotto da Aldis Run Larusdottir e Anne S. Dederichs, denominato "*A Step towards Including Children's Evacuation Prameters and Behavior in Fire Safe Building Design*". Lo studio ha come obiettivo quello di includerei bambini nella progettazione della sicurezza antincendio introducendo parametri di movimento e un modello empirico che descrivi l'evacuazione dei bambini. Questo perché la maggior parte degli studi inerenti il comportamento in caso di incendio riguardano principalmente gli adulti.

La seguente tabella è stata estrapolata dallo studio e riporta le velocità dei bambini con età compresa tra 0-2 anni e 3-6 anni.

Parameter		Children 0–2 years				Children 3–6 years			
Flow (pers/s)		$0.031\ D^2 + 0.399\ D$				$0.039\ D^2 + 0.622\ D$			
		Mean (m/s)	Min. (m/s)	Max. (m/s)	SD (m/s)	Mean (m/s)	Min. (m/s)	Max. (m/s)	SD (m/s)
Walking	Plane	0.60	0.30	0.98	0.17	0.84	0.42	1.36	0.25
	Stair 1	-	-	-	-	0.58	0.25	1.40	0.31
	Stair 2	-	-	-	-	0.38	0.29	0.48	0.07
	Stair 3	-	-	-	-	0.13	0.08	0.33	0.06
Running	Plane	1.14	0.90	1.80	0.30	2.23	0.83	3.24	0.64

Figura 37. Tabella contenente i parametri del movimento dei bambini durante l'evacuazione

Come si vede dalla tabella, lo studio analizza bambini in una fascia di età più bassa rispetto a quella del caso studio, quindi non è stato possibile prendere tali valori in considerazione.

Al fine di ottenere i valori di velocità necessari è stata analizzata la Tesi di Audrey Di Claudio, che ha come oggetto di studio la scuola Ettore Morelli. Nella tesi è stato adottato il metodo dello studio precedente e i valori delle velocità sono stati presi in modo empirico dalle riprese della prova di evacuazione. Come risultato si è ottenuto che i bambini dagli 8 ai 14 anni camminavano con lo stessa velocità dell'insegnante, mentre i bambini dai 6 ai 7 anni erano più lenti. Di conseguenza, agli studenti dalle terza elementare in poi è stata assegnata la stessa velocità degli adulti pari a 1,19 m/s, derivante dalla norma ISO/TR 16738:2009, mentre agli studenti di prima e seconda elementare una velocità pari a 0,98 m/s. Nel caso di studente disabile su sedia a rotelle è stata assegnata una velocità pari a 0,69 m/s.

4.3. Strumenti utilizzati

Vengono ora analizzati i concetti e le caratteristiche principali sui cui si basano i software utilizzati per creare l'applicativo.

Revit

Per la modellazione tridimensionale è stata utilizzata la versione 2023 di *Revit*, software prodotto da *Autodesk* che consente la progettazione con elementi di disegno e di modellazione parametrica. Revit consente di produrre documenti e progetti completi, coordinati e coerenti, basati su modelli Pertanto, le planimetrie, le sezioni, i prospetti e le viste 3D vengono aggiornati automaticamente. Il programma permette quindi di avere una visualizzazione completa dell'edificio in ogni suo dettaglio. Essendo un'applicazione intelligente, Revit consente a più collaboratori dello stesso team di condividere e salvare il lavoro effettuato sul progetto. La caratteristica principale del programma è che con il suo utilizzo è possibile ottenere un esempio davvero fedele alla realtà di quello che è stato progettato. Tutto questo è possibile attraverso elementi 3D, assonometrici e prospettici. Le caratteristiche principali di Revit includono:

- Percezione fedele della realtà, mediante uso di strumenti intelligenti;
- Possibilità di ottenere i suddetti disegni 3D realistici ad un livello tale che difficilmente può essere ottenuto solo con il disegno manuale;
- Presenza della quarta dimensione integrata nel software.

Unity

In questo paragrafo viene illustrato il programma utilizzato per la creazione del videogioco, Unity. In questo modo i passaggi che verranno descritti successivamente e la terminologia utilizzata saranno maggiormente comprensibili.

Scene View – In questa finestra viene mostrata la scena ed i suoi oggetti. Gli oggetti sono tutti gli elementi del gioco. All'interno di questo pannello sono presenti

telecamere, luci, modelli 3D, elementi dell'interfaccia utente e altri elementi. La scena è il contenitore di tali oggetti, e il progetto può essere composto da una o più scene, che possono essere considerati come i livelli del gioco. All'interno di una scena è possibile importare elementi creati con altri software oppure inserire oggetti direttamente da Unity. Quando viene creato un nuovo progetto la scena è vuota ad eccezione della Main Camera e della Directional Light.

Game View – In questa finestra viene mostrata la scena che viene inquadrata dalla Main Camera e mostra come appare il videogioco effettivo quando verrà utilizzato dall'utente finale. L'obiettivo di questo pannello è quello di mostrare in tempo reale il risultato di ciò che si sta modellando.

Hierarchy Window – La finestra delle gerarchie si trova a sinistra dello Scene Panel, e in esso vengono elencati tutti gli oggetti presenti nella nostra scena. Attraverso il pulsante Create è possibile inserire nuovi oggetti nella scena ed è inoltre possibile creare una relazione di parentela fra due o più oggetti. L'oggetto esterno verrà chiamato "padre", mentre quello interno sarà il "figlio". In questo modo è possibile attribuire agli elementi "figli" le stesse caratteristiche e comportamenti del "padre".

Project Window – In questa finestra sono presenti gli assets, ovvero gli elementi che vengono usati per il gioco: modelli 3d, texture, materiali, scene, script e altri elementi. A sinistra del project panel è presente un menù di navigazione dal quale è possibile accedere alle directory dei files.

Inspector Window – Questo panel occupa il lato destro della schermata. Esso raccoglie tutte le impostazioni e i settaggi relativi all'oggetto selezionato, come gli script, le mesh e gli elementi ad esso correlati.

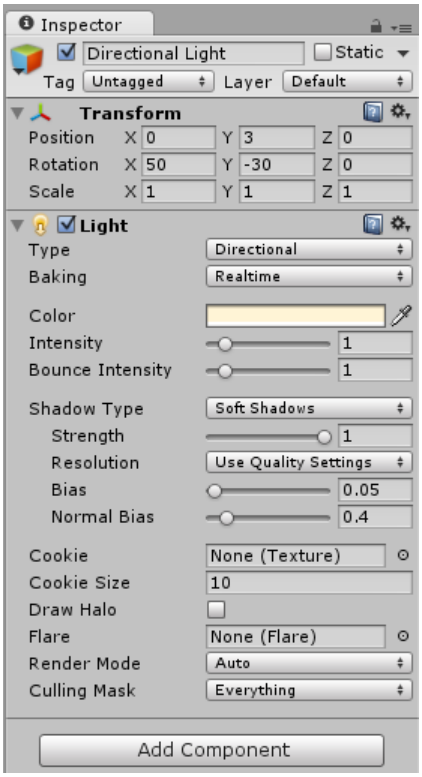


Figura 38. Esempio della finestra Inspector relativa alla Directional Light

Console Window – In questa finestra vengono mostrati gli errori, gli avvisi e altri messaggi generati da Unity.

Concept principali

È importante descrivere ora alcuni elementi fondamentali di Unity per capire alcuni passaggi relativi alla costruzione dell'applicativo. I concetti fondamentali sono tre:

- GameObjects: sono gli oggetti fondamentali in Unity, rappresentano personaggi, oggetti di scena e scenari. Ogni oggetto nel gioco è un GameObject, tuttavia un GameObject non può fare nulla da solo, è necessario dargli delle proprietà, ovvero aggiungere dei component.
- Component: parametri settabili all'interno dell'Inspector. Queste caratteristiche possono essere aggiunte o eliminate ad ogni elemento a seconda degli obiettivi da perseguire. Di seguito si riportano alcuni component:

- Collider: permette la collisione e l'interazione tra gli oggetti presenti nella scena; vi sono classi diverse di Collider come BoxCollider, SphereCollider, CapsuleCollider, MeshCollider e altri;
- Script: consente di attribuire ad un oggetto uno specifico script;
- Transform: determina la posizione, la rotazione e la scala di ogni oggetto nella scena;
- Animator: assegna l'animazione a un oggetto nella scena;
- RigidBody: consente di attribuire caratteristiche della fisica all'oggetto, come la gravità, l'accelerazione e l'attrito.

- Prefabs: il sistema Prefab di Unity consente di creare, configurare e archiviare un GameObject completo di tutti i suoi componenti e proprietà. Il prefab funge da modello da cui è possibile creare nuove istanze prefabbricate nella scena.

Scripting

Gli script sono una parte essenziale in tutte le applicazioni create con Unity. La maggior parte delle applicazioni richiede script per rispondere all'input del giocatore e per fare in modo che gli eventi nel gameplay accadano quando dovrebbero. Inoltre, gli script possono essere utilizzati per creare effetti grafici, controllare il comportamento fisico degli oggetti e implementare sistemi di intelligenza artificiale personalizzati per i personaggi del gioco.

Per scrivere gli script è necessario un linguaggio di programmazione in grado di comunicare con la macchina su cui è in esecuzione il videogioco. Esistono molti linguaggi di programmazione, e quello utilizzato in Unity è C# (denominato C Sharp). Per scrivere il codice è necessario un IDE, ovvero un ambiente di sviluppo integrato in cui è possibile scrivere, modificare e correggere il codice dello script. È stato utilizzato Visual Studio, integrato in Unity.

Caratteristiche degli script

Gli script in Unity sono generalmente considerati come *Components* perché estendono dalla classe MonoBehaviour, che a sua volta estende da Behaviour, che estende da *Component*. La classe MonoBehaviour è la classe base da cui deriva ogni script Unity, di default.
Questa proprietà implica che uno script specifico può essere assegnato a un oggetto del gioco, ad un GameObject, e per fare questa azione si trascina semplicemente lo script sull'oggetto target. Si possono distinguere due tipologie di script:

- Comportamenti: sono script che influiscono sul comportamento degli oggetti e ne definiscono e controllano le azioni. Questa tipologia estende da MonoBehaviour;
- Dati: sono script che vengono utilizzati solo come contenitori di dati e non influiscono sul comportamento dei GameObjects. Queste classi non estendono da MonoBehaviour e quindi nemmeno da Component, quindi non sono assegnate agli oggetti.

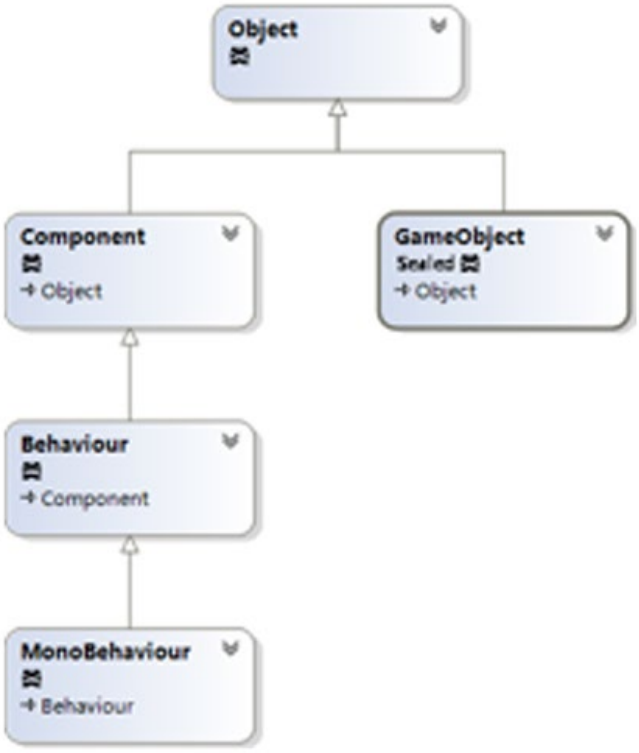


Figura 39. Struttura gerarchica degli oggetti in Unity

Struttura degli script

In Unity gli script che estendono da MonoBehaviour hanno una struttura specifica. Questa struttura è definita da una serie di funzioni, ciascuna delle quali è associata a un preciso istante di tempo in cui deve essere eseguita da parte di Unity Engine. A seguire sono indicate le principali funzioni:

- Awake: la funzione Awake è la prima ad essere richiamata, in particolare nel momento in cui l'oggetto al quale lo script è associato viene caricato;
- Start: la funzione Start e il suo contenuto entrano in esecuzione subito dopo il metodo Awake, nel preciso frame in cui lo script viene attivato (solitamente all'avvio del gioco o quando l'oggetto contenente lo script viene attivato nella scena);
- Update: la funzione Update viene eseguita ad ogni frame del gioco ed è la funzione più comunemente utilizzata per implementare qualsiasi tipo di script di gioco;
- FixedUpdate: la funzione FixedUpdate viene eseguita un determinato numero di volte per frame;
- LateUpdate: la funzione LateUpdate viene eseguita una volta completate tutte le chiamate alle funzioni Update.

Le funzioni descritte non vengono sempre utilizzate nello stesso script, poiché è il programmatore che decide quali utilizzare, a seconda di cosa deve fare lo script e di come questo influisce nel tempo sul comportamento dell'oggetto al quale è associato.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class PrimoScript : MonoBehaviour {
 // Use this for initialization
 void Start () {

 }

 // Update is called once per frame
 void Update () {

 }
}
```

Figura 40. Apertura di un nuovo script su Visual Studio

Variabili in C#

All'interno dello script vanno inserite le variabili, che sono quella parte del codice che consentono di salvare e manipolare i dati. Esistono molti tipi di dati e di conseguenza un numero infinito di tipi di variabili. All'interno dello script la prima cosa da fare è dichiarare le variabili che verranno usate all'interno dello script. Ogni variabile va dichiarata specificandone modificatore di accesso, tipo e nome.

- Modificatore di accesso: un modificatore di accesso è una parola chiave che attribuisce una particolare visibilità alla variabile. I modificatori di accesso più utilizzati sono *public* e *private*, e per scegliere quale attribuire alla variabile è necessario decidere se si vuole che la variabile sia modificabile ed accessibile dall'esterno. Le variabili con modificatore *public* sono effettivamente visibili dall'Inspector Window del componente, e quindi possono essere modificate da esso o da altri script del videogioco. Tuttavia, a volte è bene (o necessario) nascondere le variabili che non devono essere modificate da altre persone o da script. Definire una variabile come *private* significa renderla nascosta e non è possibile modificarne il valore senza accedere direttamente allo script. Il modificatore di accesso può essere omesso, ma in questo caso la variabile viene considerata privata;

- Tipo di dato: il tipo di un dato è la parte più importante della dichiarazione della variabile, infatti consente alla macchina di capire con quale dato sta lavorando e di poterlo gestire opportunamente. Il tipo della variabile, a differenza del modificatore di accesso, va sempre specificato. Non può esistere una variabile che non abbia il proprio tipo chiaramente definito, anche perché Visual Studio inizierebbe a segnalare errori. Esistono molti tipi di dati diversi, e quelli elencati sono i principali:
    - int: indica un numero intero (ad esempio 2, 3, 50);
    - float: indica un numero decimale e per un corretto riconoscimento il numero decimale deve essere separato dalla sua parte intera da un punto e avere una f finale (ad esempio 3.14f);
    - string: indica una stringa, ovvero una sequenza di caratteri, e al suo interno si possono salvare parole, frasi o una sequenza di numeri e lettere;
    - bool: indica una variabile booleana e funziona come un interruttore che può assumere solo uno tra due valori: vero o falso (true o false).
- Nome della variabile: il nome della variabile è l'ultima parte della definizione della variabile, che deve essere univoco, non deve contenere spazi e deve essere diverso dal tipo di dato.

Una volta definiti modificatore di accesso, tipo di dato e nome la dichiarazione di una variabile è conclusa. A questo punto è necessario terminare la definizione con un punto e virgola, che rappresenta un segno molto importante in molti linguaggi di programmazione e deve essere utilizzato alla fine di qualsiasi istruzione eseguita in uno script.

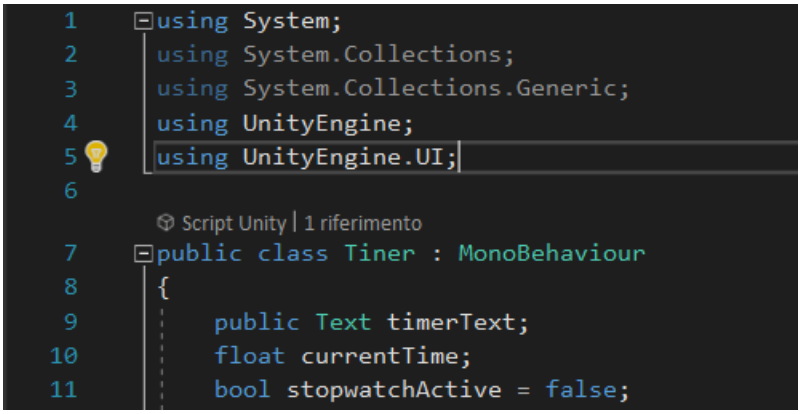
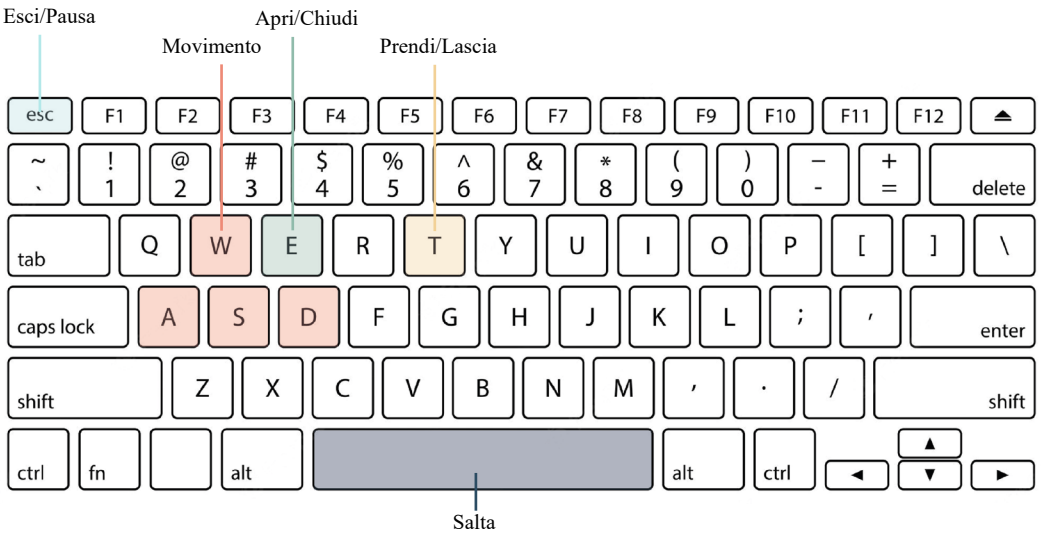


Figura 41. Esempio di dichiarazione varibili in Visual Studio

4.4. Flow-chart dell'applicativo

Successivamente all'importazione del modello BIM della scuola in Unity è stato possibile iniziare con la creazione vera e propria dell'applicativo. Per capire come strutturare il gioco nelle sue fasi è stato costruito un flow-chart nel quale sono inseriti tutti gli step che deve compiere lo studente per uscire dall'edificio in sicurezza. Tutto inizia con il suono della campanella che segnala l'incendio, e quindi l'evacuazione. Il giocatore con il suono della campanella si alza in piedi insieme ai suoi compagni e attende le istruzioni del professore, che si dirigerà poi verso la porta. Il giocatore con il mouse ha la possibilità di guardarsi intorno a 360°, mentre con i tasti della tastiera si può muovere. I comandi sono così rappresentati:



Al giocatore viene data la possibilità di prendere in mano, premendo il tasto T, un libro che si trova sul suo banco. Nel caso in cui il giocatore deciderà di prendere in mano l'oggetto comparirà sullo schermo un segnale di errore, nel quale viene spiegato che durante l'evacuazione non si possono prendere oggetti personali, e che quindi dovrà ricominciare il gioco. In alternativa, se il giocatore lascia l'oggetto sul banco, il gioco prosegue. Dopodiché, il giocatore segue in modo ordinato i compagni e si dirige in corridoio, dove dovrà aspettare il segnale del professore per dirigersi verso l'uscita. Se il giocatore decide di separarsi dal gruppo o inizia a correre superando il professore, anche in questo avrà perso il gioco e lo dovrà

iniziare da capo. Questo serve per far comprendere agli alunni come deve essere eseguita una evacuazione in sicurezza, in modo che durante le prove di evacuazione o una emergenza vera e propria sapranno il comportamento da tenere. Seguendo il professore e scendendo le scale si arriva all'ingresso della scuola, dove bisognerà uscire dalla porta di emergenza e giungere infine al checkpoint. A questo punto sullo schermo comparirà un messaggio di vittoria, poiché si è riusciti a completare l'evacuazione.

Inizialmente si era pensato di creare un livello anche per gli alunni disabili in carrozzina, ma la scuola predispone che su ogni piano, vista l'impossibilità di prendere le scale e il divieto di utilizzare l'ascensore, degli spazi calmi, stanze ignifughe in cui si devono aspettare i soccorsi in caso di incendio. Per questa ragione sono stati creati due livelli, rispettivamente un livello per la scuola primaria, ambientato nella classe 2B situata al primo piano, e un livello per la scuola secondaria di I grado, ambientato nella classe 1F situata al secondo piano. A seguire è stata inserita la raffigurazione del flow-chart di partenza per la creazione dell'applicativo.

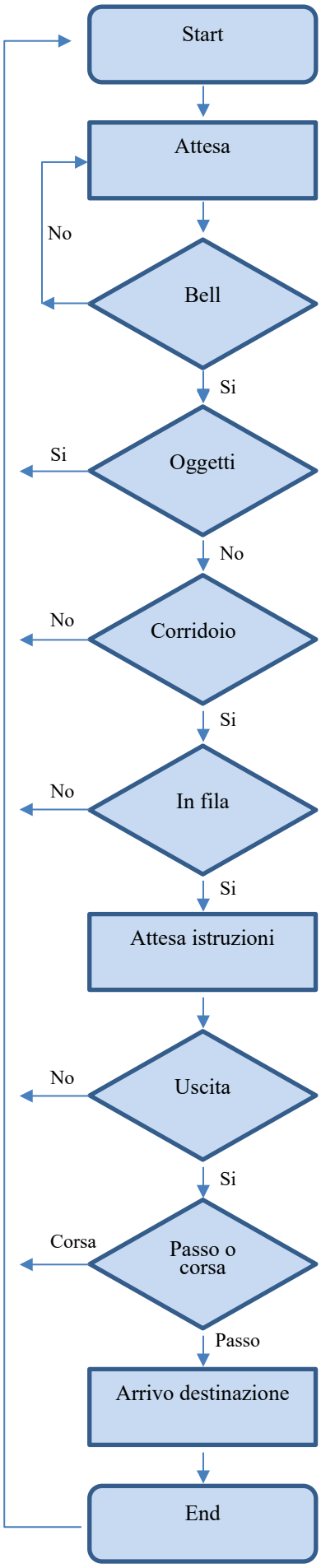


Figura 42. Flow-chart del videogioco

Principi dell'applicativo

L'applicativo sviluppato ha come obiettivo quello di ricreare uno scenario di emergenza antincendio nel quale il giocatore deve evacuare dalla scuola seguendo i segnali di emergenza e le indicazioni del docente. In particolare, sono stati sviluppati due livelli: uno per gli studenti della scuola primaria e uno per gli studenti della scuola secondaria di primo grado. Ciò che differenzia i due livelli sono:

- Posizione della classe: la classe primaria è situata al primo piano, mentre la classe secondaria di primo grado al secondo piano. Questa decisione rende più semplice il percorso che dovrà intraprendere lo studente delle elementari, poiché dovrà scendere solo un piano di scale, e in questo modo avrà meno distrazioni;
- Dimensione studenti: come nella realtà, anche nel gioco sono state differenziate l'altezza e la dimensione degli studenti, infatti quelli delle medie sono più piccoli e più bassi rispetto a quelli delle medie;
- Velocità studenti: essendo gli studenti delle elementari più piccoli, anche il loro passo è più corto, e di conseguenza la loro velocità è più lenta rispetto agli studenti delle medie. Anche la velocità del docente è stata scelta in modo che gli studenti potessero tenere il passo più facilmente, senza farli correre;
- Message Box: per facilitare gli studenti delle elementari, i message box con all'interno le istruzioni da seguire hanno un carattere più grande e il tempo di lettura è stato aumentato in modo da rendere più facile la comprensione del gioco;
- Frecce: all'interno del livello delle elementari sono state inserite delle frecce che indicano la strada da seguire per evacuare la scuola, mentre per il livello delle medie non sono state inserite e come riferimento devono seguire il docente che li guiderà all'esterno dell'edificio.

4.5. Interoperabilità Revit - Unity

Dopo aver implementato il modello BIM della scuola tramite Revit Autodesk 2023, il passo successivo è stato quello di importazione dello stesso all'interno di Unity (la versione utilizzata è la 2019.4, che si è ritenuta più stabile). Per l'importazione del modello della scuola all'interno di Unity è stato indispensabile l'utilizzo di un plugin esterno, SimLab Composer. Innanzitutto è stato aperto il modello della scuola tramite il software Revit Autodesk, è stata settata la vista 3D, ed è stato esportato il modello in formato DWFx.

Successivamente, il file DWFx è stato importato all'interno del plugin SimLab Composer, dove la scena creata è stata "impacchettata" in formato FBX. Questo passaggio ha portato alla creazione di un file zip, che è stato successivamente decompresso, all'interno del quale è presente una cartella che include il file FBX e tutte le textures del modello. Dopo questi step è stato aperto il software Unity, nel quale è stato importato il file FBX all'interno della cartella A*ssets*, ma si tratta di un modello privo di textures.

Cliccando sul modello 3D aggiunto agli assets viene visualizzata una finestra di dialogo, dalla quale, cliccando su "estrai materiali", è stato possibile creare una nuova cartella denominata *Materials*. Le textures presenti nella cartella decompressa sono state quindi selezionate e trascinate nella cartella Materials. Il modello, dopo questi passaggi, ha ora tutte le textures necessarie ed è pronto per essere utilizzato. Nell'immagine si può osservare il risultato finale dell'importazione all'interno di Unity dell'intero modello.

Figura 43. Modello importato in SimLab Composer

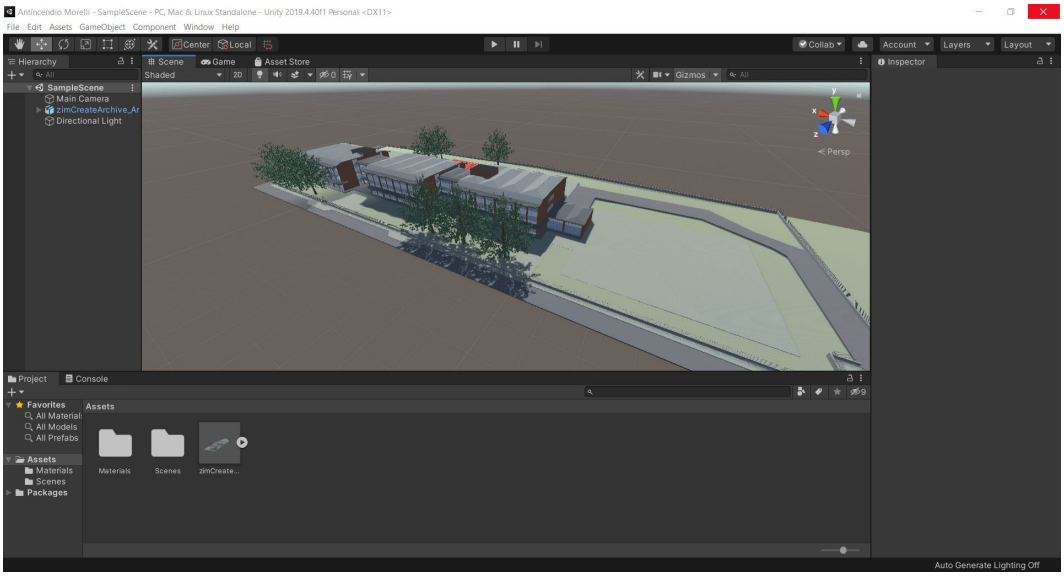


Figura 44. Modello importato in Unity

Interoperabilità

Il modello BIM della scuola è stato realizzato con il software Revit Autodesk 2023 e presenta il grande vantaggio di essere facilmente modificabile e di poter essere importato in altri software. Nel caso studio della scuola Morelli si sono presentate diverse modalità per importare il modello nel software Unity. In particolare, una modalità prevedeva di importare la scena di Revit in 3ds Max tramite l'esportazione del file FBX. Dopodiché, avendo copiato da Revit le trame e i materiali, è stato

possibile attribuire a ciascun elemento tridimensionale uno specifico materiale. Questo metodo è stato provato, ma non ha portato i risultati attesi, poiché alcuni dei materiale e delle caratteristiche dell'edificio venivano a mancare. Il problema che si è presentato durante l'uso di questa metodologia è stato attribuito alle dimensioni del file, troppo grande, e la necessità era quella di poter importare in Unity l'intero edificio, e non solo parti di esso. Per queste ragioni si è preferito adottare la metodologia descritta in precedenza, tramite l'utilizzo del plugin SimLab Composer.

Si tratta di un esempio di interoperabilità, cioè la possibilità di scambiare modelli, applicativi e le informazioni correlate tra diverse piattaforme software. L'interoperabilità tra diverse piattaforme software consente di realizzare scenari di realtà virtuale che riproducono fedelmente edifici, strutture e ambienti reali. La possibilità di mantenere un elevato livello di dettaglio è uno dei maggiori vantaggi dell'interoperabilità tra i software di modellazione BIM, che diventa importante in casi di studio come quello qui presentato, dove il realismo del modello e degli oggetti in esso contenuti è importante.

Figura 45. Foto della scala al secondo piano

Figura 46. Screenshot della scala del secondo piano visualizzata in Revit

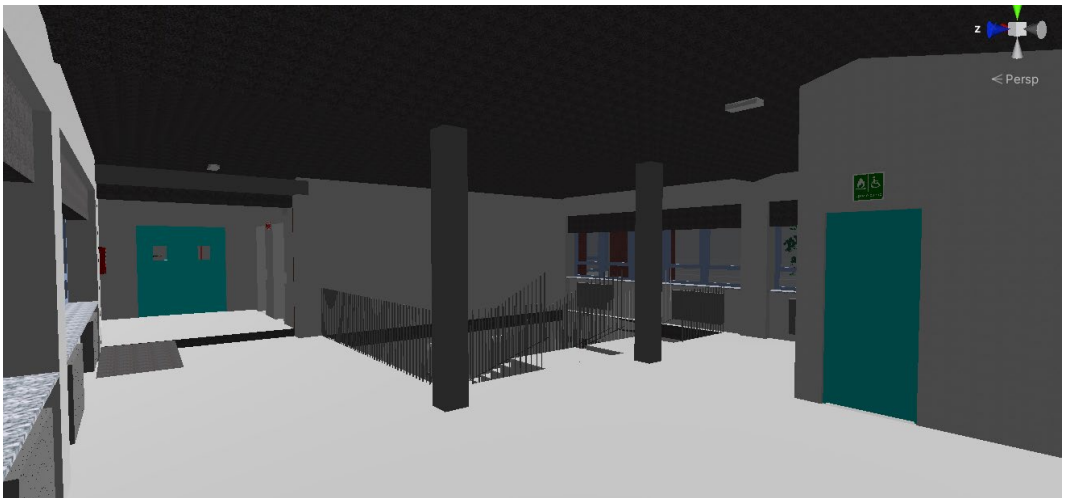


Figura 47. Screenshot della scala del secondo piano visualizzata in Unity

4.6. Sviluppo applicativo

A seguire vengono illustrati i passaggi che hanno portato alla creazione dell'applicativo a partire dalla realizzazione del giocatore, dei personaggi non giocanti e delle azioni che vengono svolte al fine di vincere il gioco.

Level Manager

Il *Level Manager*, visibile all'interno della Hierarchy, è di importanza fondamentale, perché, come dice il nome stesso, è l'oggetto tramite il quale è possibile gestire e modificare il livello del videogioco. Il Level Manager, visualizzato nell'Inspector Window, è collegato con lo script principale denominato InitiScene, codificato su Visual Studio, in cui sono presenti i settaggi del livello. All'interno dello script InitiScene sono presenti variabili private, che possono essere modificate solamente all'interno dello script stesso, e variabili pubbliche, che possono essere modificate e settate all'interno dell'Inspector Window. Questa parte iniziale di codice che viene mostrata contiene le variabili *Npc Settings* e *Player Settings*, che sono pubbliche, e quindi visibili e modificabili nell'Inspector Window. Queste due variabili sono importanti perché all'interno di Player Settings viene collegato il giocatore principale, mentre in Npc Settings vengono collegati i personaggi non giocanti, in particolare, non essendo questi ultimi controllati del giocatore, devono essere settati in modo che seguano un percorso definito ed è possibile impostare la velocità a cui devono camminare.

```
public class InitiScene : MonoBehaviour
{
    [Space]
    [Header("Npc Settings")]
    [Space]

    public List<Animator> studentAnimators;
    public GameObject ProfessorPatrol;
    public float ProfessorSpeed = 0.11f;
    public List<GameObject> allPatrols;
    public float patrolSpeed = 1f;
    public float returnToPatrolTime = 0.3f;

    [Space]
    [Header("Player Settings")]
    [Space]

    public GameObject player;
    //public Collider playerCollider;
    //public GameObject colliderPosition;
```

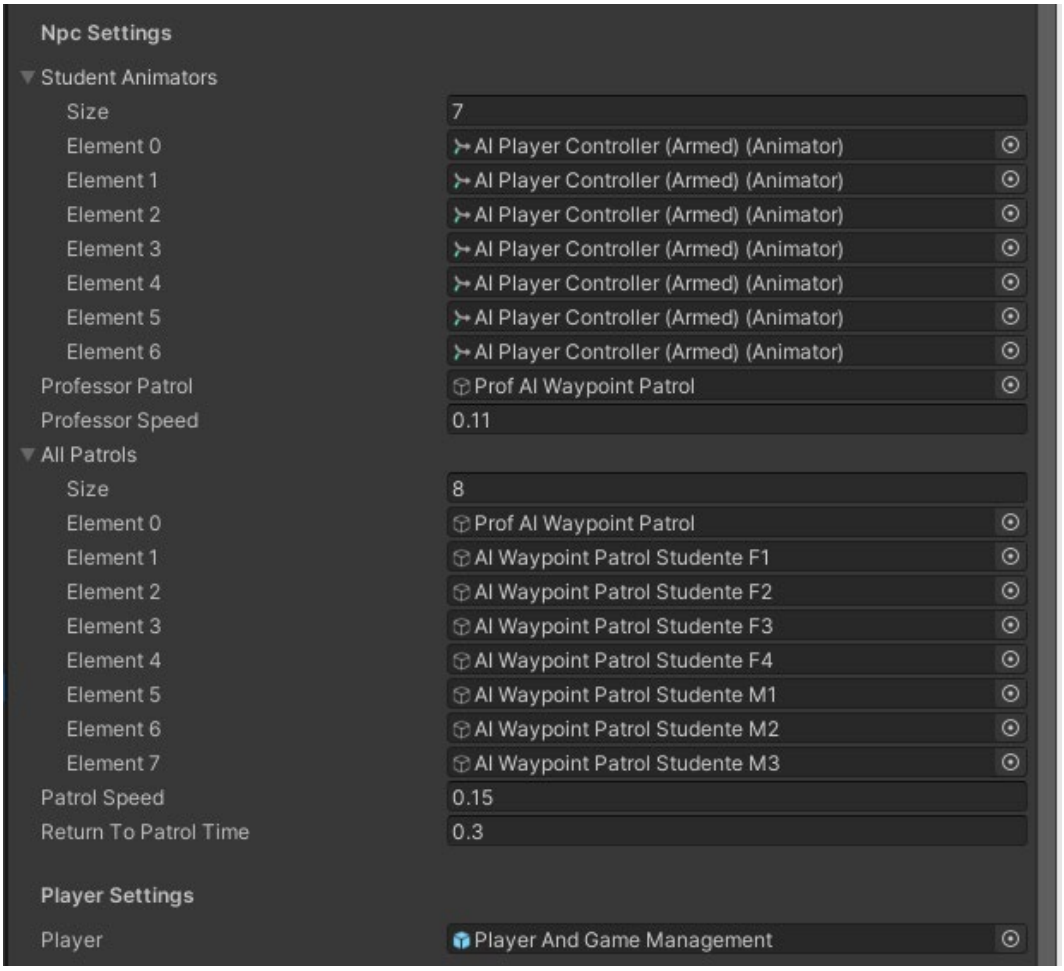


Figura 48. Variabili Npc Settings e Player Settings visibili e modificabili nell'Inspector Window collegate allo script InitiScene

Successivamente, sono stati inseriti i *General Settings*, ovvero le impostazioni generali che possono essere modificate in parte nell'Inspector Window, essendo variabili pubbliche, e in parte solamente all'interno dello script InitiScene, essendo queste variabili private. All'interno dei General Settings ci sono diverse variabili pubbliche che vengono ora descritte:

- **Bell Ring**: questa variabile rappresenta il suono della campanella in situazione di emergenza. È stato importante inserire questa variabile perché il suono della campanella è collegato all'inizio del movimento del professore e degli studenti, che si alzano in piedi, e alla comparsa di un messaggio pop up da parte del professore che da istruzioni agli studenti. All'interno della Hierarchy è stato inserito un GameObject denominato *Bell*

all'interno del quale è stato collegato il suono della campanella, il scaricato da internet in formato mp3. Nell'Inspector Window è inoltre possibile modificare le impostazioni relative al suono della campanella, come ad esempio il volume.

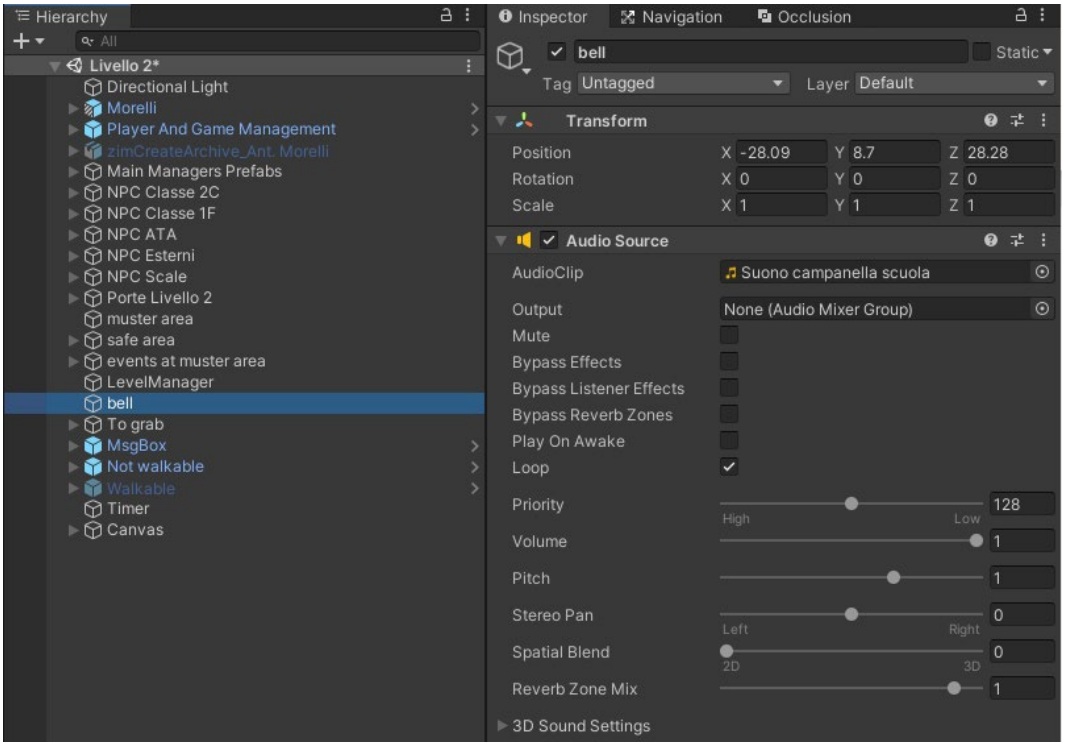


Figura 49. Variabile Bell e relative impostazioni

- **Timer**: all'interno del videogioco è stato inserito un timer per verificare il tempo impiegato degli studenti per uscire dalla scuola. Per la variabile Timer è stato creato uno script che ne descrive il funzionamento dove viene specificato in che momento deve partire, ovvero quando suona la campanella, e quando deve terminare, ovvero quando viene raggiunto il checkpoint fuori dalla scuola. All'interno della Hierarchy è stato creato, come per la campanella, un GameObject, denominato Timer. In questo caso è stato aggiunto anche un canvas che rappresenta la visualizzazione grafica del timer all'interno del videogioco, dove sono stati inseriti il testo Time To Exit e il timer vero e proprio.

Figura 50. Visualizzazione grafica del Timer nel gioco

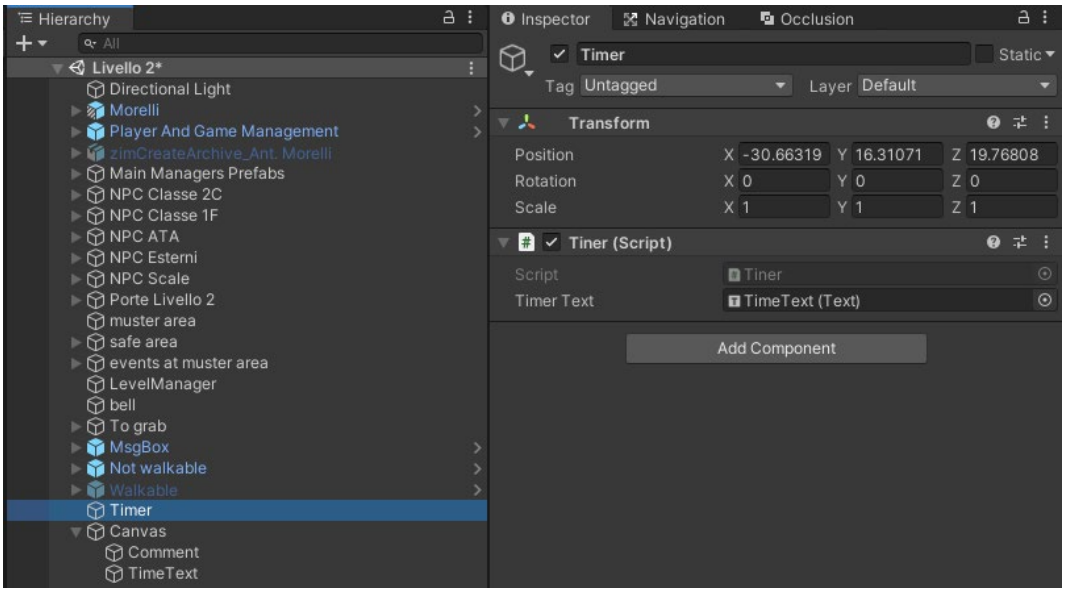


Figura 51. Variabile Timer e relative impostazioni

A seguire lo script che rappresenta il funzionamento del Timer:

```
public class Tiner : MonoBehaviour
{
    public Text timerText;
    float currentTime;
    bool stopwatchActive = false;

    // Start is called before the first frame update
    void Start()
    {
        currentTime = 0;
    }

    // Update is called once per frame
    void Update()
    {
        if (stopwatchActive)
        {
            currentTime += Time.deltaTime;
        }
        TimeSpan time = TimeSpan.FromSeconds(currentTime);
        timerText.text = time.ToString(@"mm\:ss\:fff");
    }

    public void StarStopwatch()
    {
        stopwatchActive = true;
```

```
    }
    public void StopStopwatch()
    {
        stopwatchActive = false;
    }
}
```

- **Pop Up Box e Gui Text (Graphical User Interfaces)**: queste variabili rappresentano il messaggio che compare all'apertura del livello nel quale vengono spiegate le istruzioni del gioco. Anche in questo caso è stato aggiunto un nuovo GameObject, denominato *MsgBox*, all'interno del quale sono presenti due Object: *Panel* rappresenta il pannello con sfondo a tinta unita all'interno del quale è stato inserito il testo, in modo da facilitare la lettura, mentre in *Text* è possibile modificare il testo vero e proprio delle istruzioni del videogioco. Nel Level Manager sono state poi collegate le voci corrispettive, abbinando Panel a Pop Up Box e Text a Gui Text.

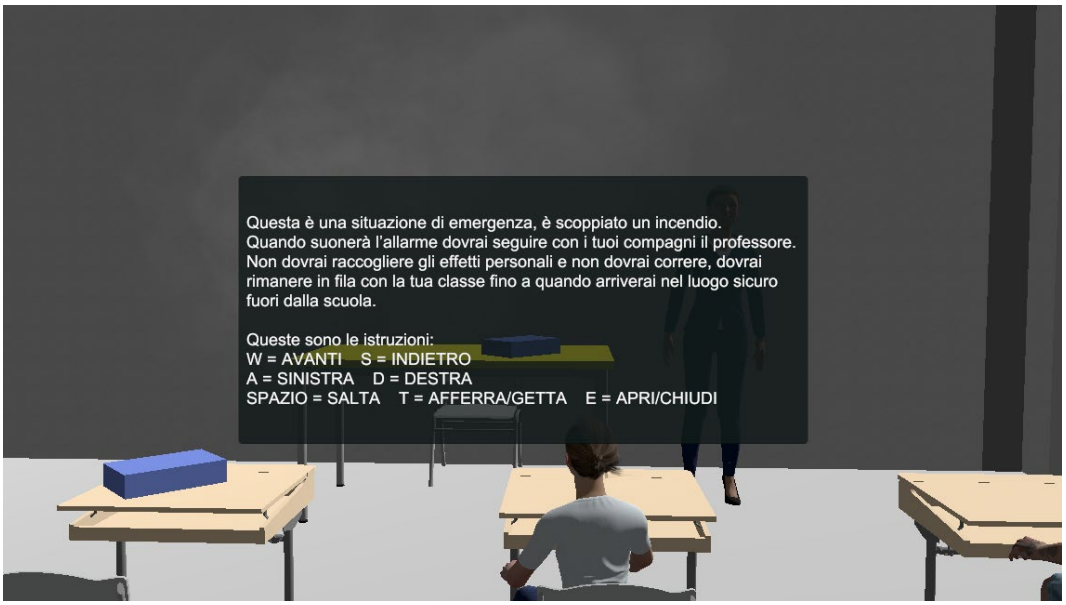


Figura 52. Messaggio iniziale con istruzioni

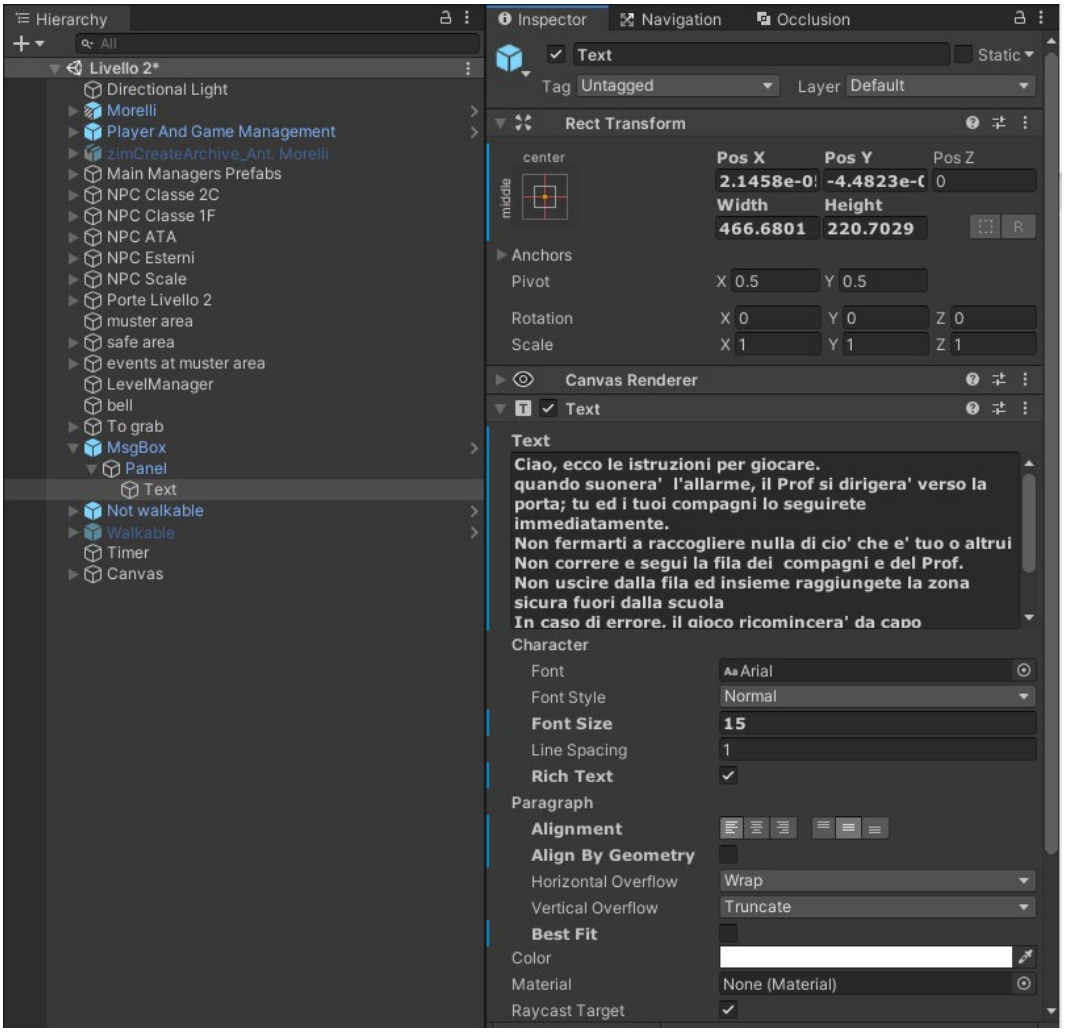


Figura 53. Object Text contenente il testo delle istruzioni

- **Time**: sono presenti quattro variabili riferite al tempo. Di seguito vengono analizzate:

  - Time To Start: questa variabile indica il tempo che si deve aspettare prima che suoni la campanella, in modo che il giocatore abbia il tempo di leggere le istruzioni;
  - Time Prof Is Waiting: questa variabile indica il tempo di attesa del professore davanti alla porta antipanico in corridoio, in modo che possa accertarsi che tutti gli studenti siano presenti;

- Time Prof To Start: questa variabile indica il tempo che il professore aspetta dopo essersi alzato dopo il suono della campanella per dare istruzioni agli studenti;
- Time To Escape: questa variabile stabilisce il tempo massimo in cui il giocatore deve uscire dalla scuola. Se viene superato il tempo stabilito il giocatore dovrà ricominciare il livello dall'inizio.

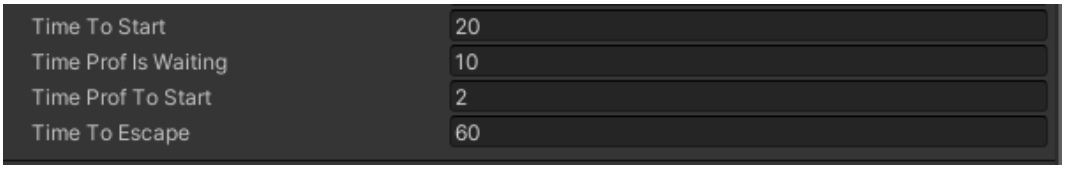


Figura 54. Variabili Time settabili nell'Inspector Window

Muster Area

Questa area, denominata *Muster Area*, è stata inserita all'interno della scena poiché quando viene attraversata vengono attivati degli input per i personaggi non giocanti. Questi eventi sono stati denominati *Events At Muster Area* e a ciascuno è stato collegato un *Event Trigger System*, che è un componente di Unity per il quale l'attivazione di un evento può essere attivata immettendo una condizione specifica per un'interazione. All'interno di Events At Muster Area si trovano:

- Detect Prof Moving: quando il professore si alza in piedi con il suono della campanella viene attivato il timer e i personaggi non giocanti seguono il professore fuori dalla classe;
- Detect Player In Muster Area: quando il giocatore si trova all'interno della Muster Area, ovvero al di fuori della porta della classe, compare un messaggio e il giocatore deve aspettare la partenza del professore che si assicura che tutti gli studenti siano presenti, successivamente il professore aprirà la porta antipanico e gli studenti e il giocatore potranno seguirlo;
- Partenza esterni: con il passaggio del professore in questo punto tutti i personaggi non giocanti presenti nel livello iniziano a camminare seguendo un percorso stabilito per raggiungere il luogo sicuro.

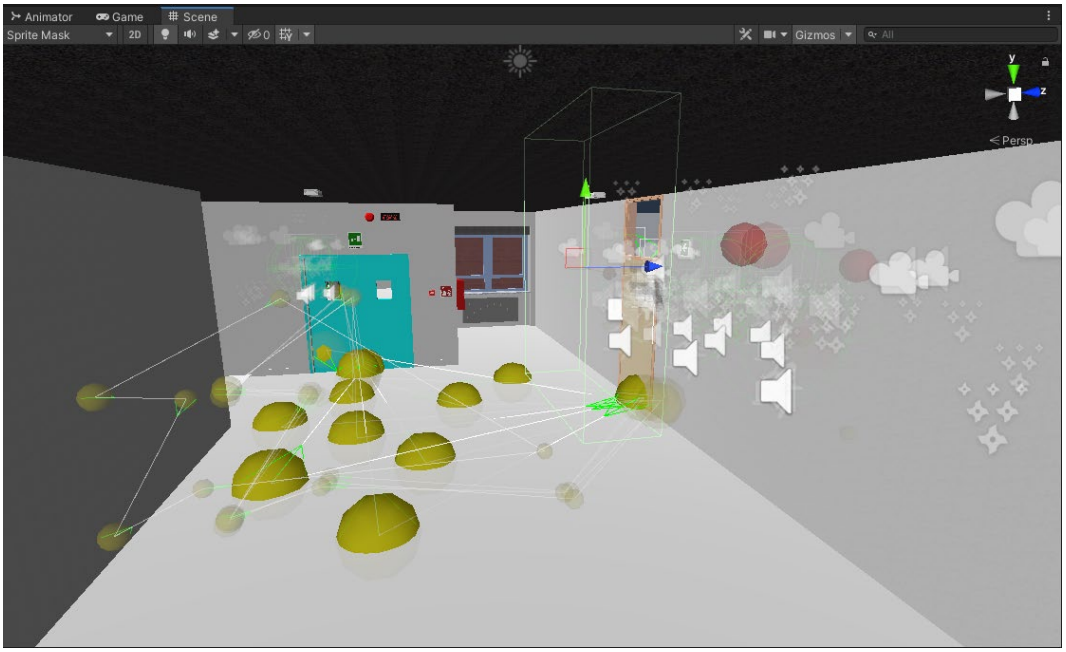


Figura 55. Muster Area rappresentata dal Box Collider dacanti alla porta

Safe Area

La *Safe Area* rappresenta il luogo finale in cui il giocatore deve arrivare per terminare il livello. Per creare questa area sono stati inseriti dei *Mesh Renderer* che delimitano la zona esterna alla scuola. È stato poi collegato un *Event Trigger System*, che è un componente di Unity per il quale l'attivazione di un evento può essere attivata immettendo una condizione specifica per un'interazione. In questo caso con l'arrivo alla Safe Area il Timer viene interrotto, indicando così il temo impiegato per l'evacuazione, la campanella smette di suonare e compare un messaggio di vincita del livello.

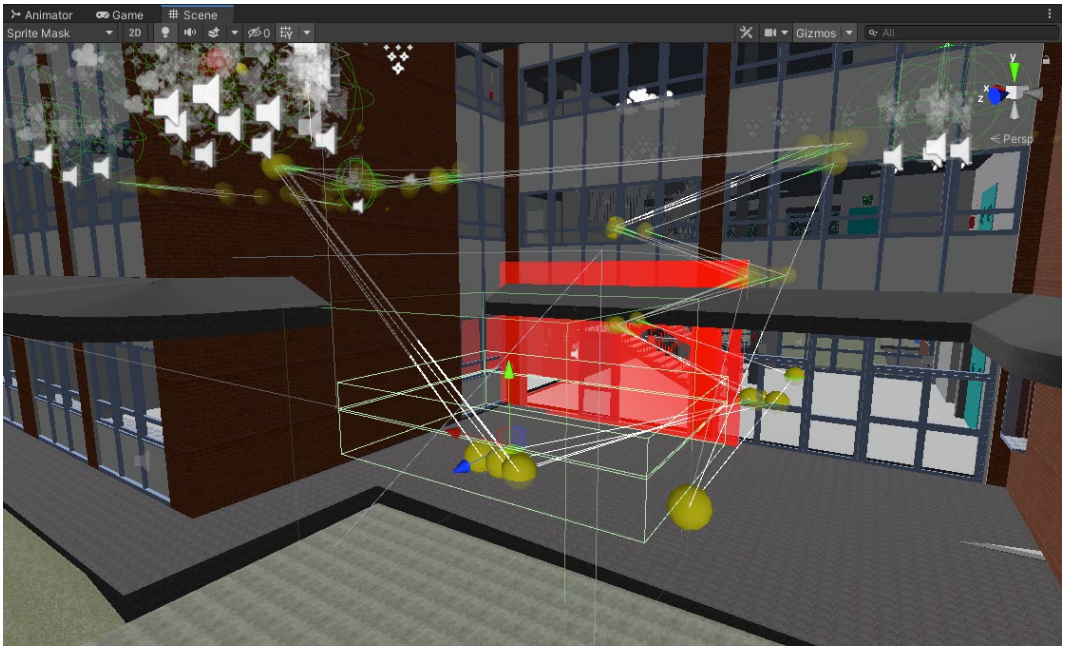


Figura 56. Safe Area

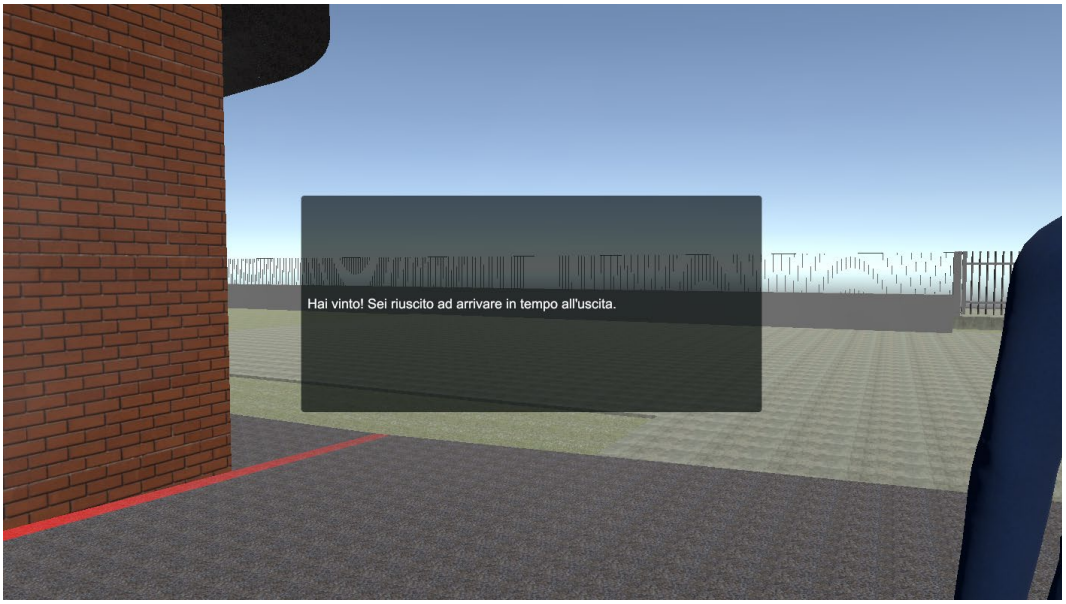


Figura 57. Arrivo alla Safe Area con tempo indicato e messaggio di vittoria

Creazione dei personaggi

Viene ora descritto il procedimento per la creazione del giocatore principale e dei personaggi non giocanti e del loro movimento.

Importazione modelli

Per la creazione del giocatore e dei personaggi non giocanti è necessario scaricare dei modelli. I modelli sono stati scaricati dal *Mixamo*, una società di grafica 3D che fornisce una vasta collezione di personaggi 3D di alta qualità e centinaia di animazioni di personaggi. Una volta scelto il Character idoneo è possibile scaricare il modello nel formato FBX for Unity nella posa "T-pose", ovvero con le braccia aperte. Una volta scaricato il modello questo viene inserito all'interno della cartella Models contenuta negli Assets. Una volta inseriti nella cartella Models il modello scaricato può essere modificato in base alle necessità. Cliccando sul modello si apre l'Inspector Window, dove sono presenti delle sotto-finestre. Nella finestra Model è possibile modificare la scala del modello, in questo caso è stata mantenuta a 1 per gli adulti mentre è stata ridotta per gli studenti; nella finestra Rig è stato assegnato ad Animation Type la caratteristica Humanoid, perché appunto i personaggi sono e si devono comportare come umani; nella finestra Animation tutte le spunte devono essere verdi per assicurare che le animazioni dei personaggi siano attive; infine, nella finestra Materials alla voce Material Creation Mode è stata scelta l'opzione Standard e alla voce Location è stata attribuita l'opzione Use External Materials (Legacy), in modo che i materiali del personaggio fossero quelli del modello scaricato da Mixamo. A questo punto è possibile creare il giocatore e i personaggi che animeranno il gioco.

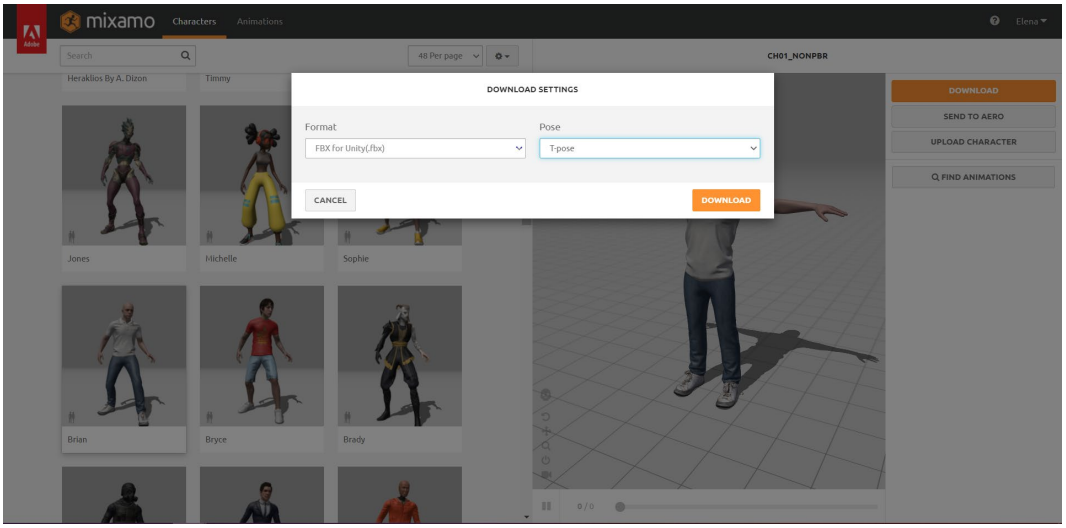


Figura 58. Download del modello da Mixamo

Creazione del giocatore

Il passo successivo dopo aver importato i modelli dei personaggi all'interno di Unity e averli settati correttamente è stato quello di creare il giocatore. Per la creazione del giocatore principale e dei personaggi non giocanti è stato scaricato da Unity Asset Store un engine denominato *Game Kit Controller* (GKC). Questo engine consente di inserire nella scena il giocatore completo di alcune caratteristiche che andranno poi settate in base al videogioco che si vuole creare, in questo caso si tratta di una scuola con all'interno studenti e adulti. Per questo videogioco il giocatore principale è un First Person Controller, che è quindi controllato e visualizzato da una prospettiva in prima persona. Per inserire all'interno della scena il giocatore principale si clicca su Game Kit Controller, poi su Create New Character, in questo modo compare una finestra dove si sceglie il Player Type, ovvero il tipo di giocatore, in questo caso Player, e si trascina il modello scaricato da Mixamo dello studente nella finestra FBX Model, che rappresenta la fisicità del giocatore. A questo punto, cliccando su Create Player viene inserito all'interno della scena il giocatore principale, che però in fase di progettazione non viene visualizzato come il modello scelto ma con sembianze robotiche. In fase di gioco poi verrà visualizzato come il modello definito durante la creazione del giocatore. Il giocatore inserito all'interno della scena può muoversi

attraverso i tasti della tastiera W, A, S e D e può guardarsi intorno muovendo il mouse.

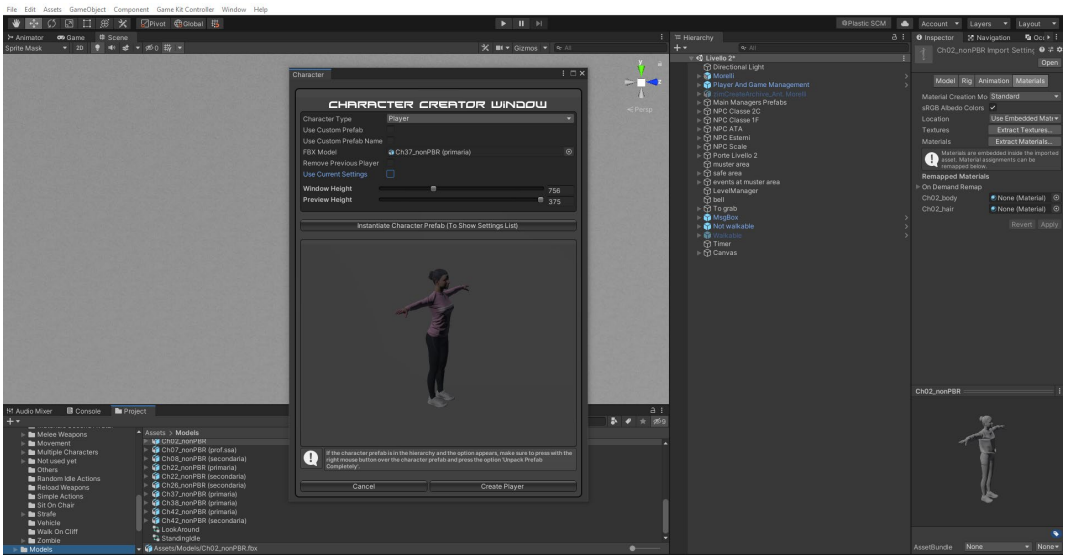


Figura 59. Creazione del giocatore con GKC

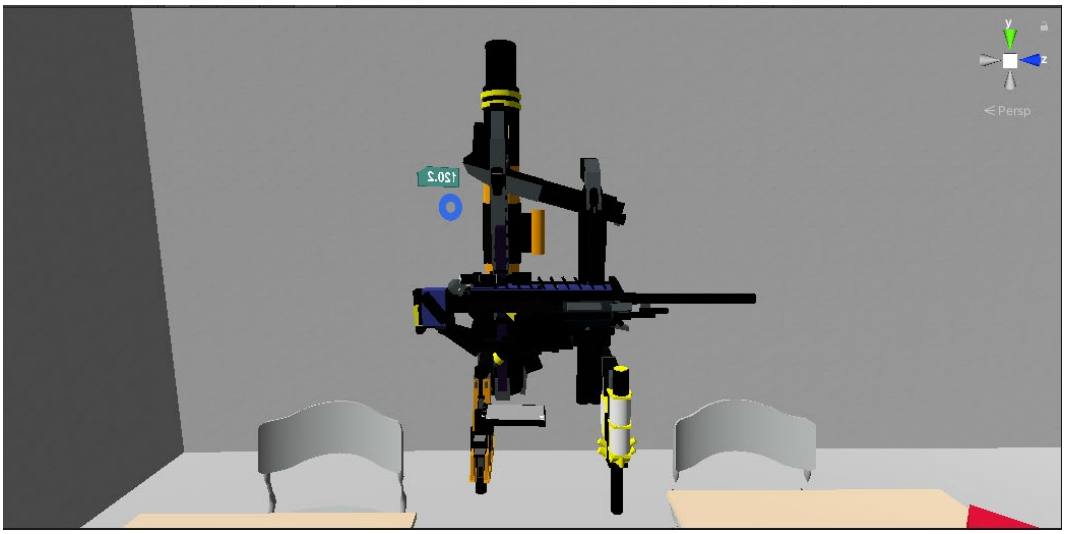


Figura 60. Visualizzazione del giocatore nella progettazione del videogioco

Creazione dei personaggi non giocanti

Successivamente la creazione del giocatore sono stati aggiunti i personaggi non giocanti (NPC, Non-Player Character), cioè i personaggi che non sono sotto il controllo diretto del giocatore, che sono gli studenti che compongono la classe, il professore e il personale ATA. Per la creazione dei giocatori non giocanti è stato seguito lo stesso processo per la creazione del giocatore principale, con la differenza che è stato selezionato Neutral alla voce Player Type, mentre alla voce AI Type è stata selezionata l'opzione Unarmed, ovvero non armati. Infine è stato trascinato nella casella FBX Model il modello del giocatore non giocante prescelto, in questo caso quello del professore.

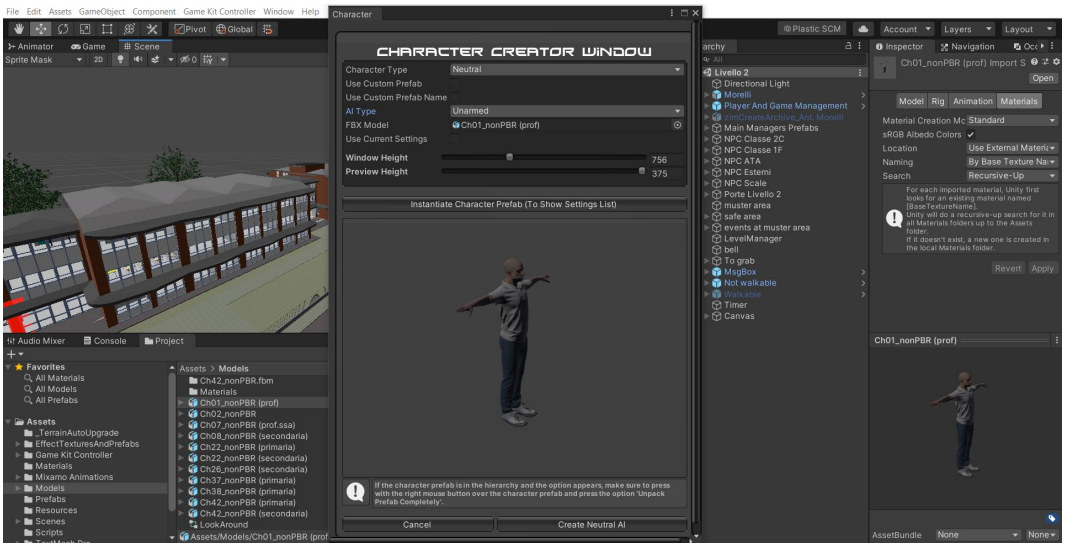


Figura 61. Creazione NPC del professore

Cliccando poi su Create Neutral AI il personaggio non giocante del professore viene inserito all'interno della scena ed è possibile spostarlo dietro la cattedra. Una volta che è stato inserito il NPC è stato necessario intervenire su alcune caratteristiche all'interno del corrispettivo AI Player Controller. Nell'Inspector Window, in particolare, sono stati modificati alcuni settaggi: sotto la finestra Gravity System è stata disabilitata la voce Gravity Power Enabled, poiché la forza di gravità agente nel videogioco viene stabilita dal programmatore; sotto la finestra Health è stata disabilitata l'opzione Use Health Slider, poiché non trattandosi di un combat game,

uno sparatutto o di un gioco in generale dove la salute può diminuire, la barra della vita non si è ritenuta necessaria; infine, sotto la finestra Find Objectives System sono state disabilitate le voci Follow Partner On Trigger e Wander Enabled, poiché non si ha un partner da seguire e non si deve passeggiare dove si vuole, ma si deve seguire un percorso definito. Successivamente, con la stessa metodologia, sono stati creati tutti gli studenti presenti nella classe.

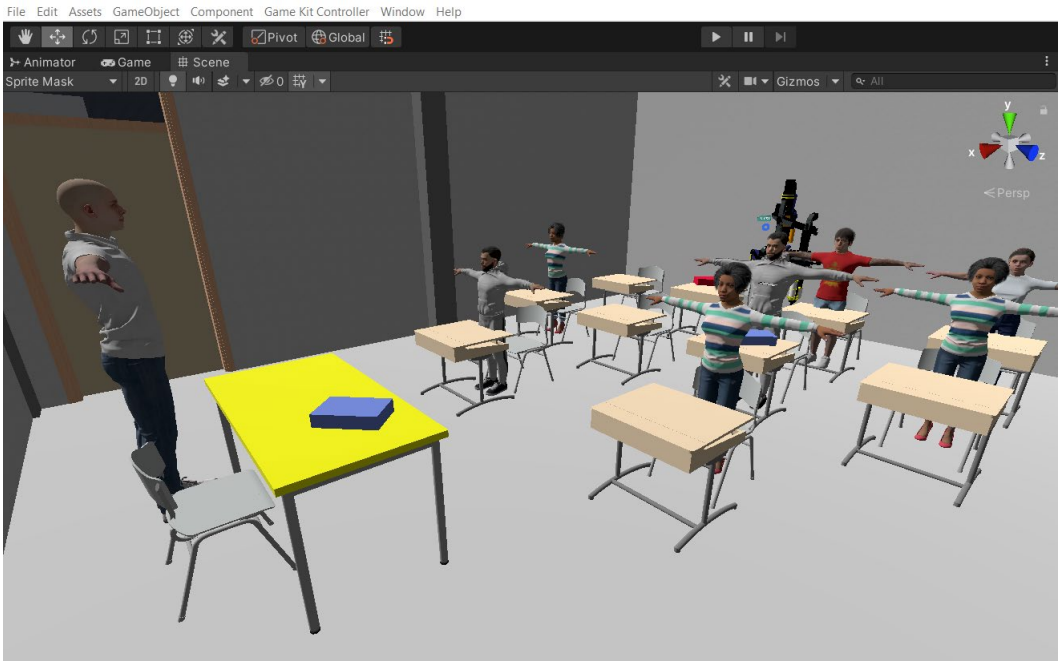


Figura 62. Scena con professore e studenti in classe

Movimento dei personaggi non giocanti

I personaggi non giocanti, come dice il nome, non sono sotto il controllo del giocatore, e quindi il loro movimento è deciso dal programmatore. Per questo caso studio gli NPC devono seguire un percorso preciso che li dirige all'esterno della scuola a un checkpoint definito. Per fare in modo che gli NPC seguano un percorso stabilito sono stati attribuiti dei Patrol System, ovvero una serie di punti in cui il personaggio deve necessariamente passare. Per aggiungere un Patrol System si clicca semplicemente sul personaggio al quale si vuole aggiungere, in modo che si aprirà l'Inspector Window, e basterà selezionare Add Patrol System To AI. In questo modo viene aggiunta una nuova finestra in Hierarchy sotto al personaggio a cui si è aggiunto il Patrol System con la dicitura AI Waypoint Patrol, che contiene

al suo interno una lista di numeri che rappresentano i punti che il personaggio deve raggiungere secondo l'rodine stabilito. Questi punti possono essere aggiunti tramite l'Inspector Window cliccando su Add Point.

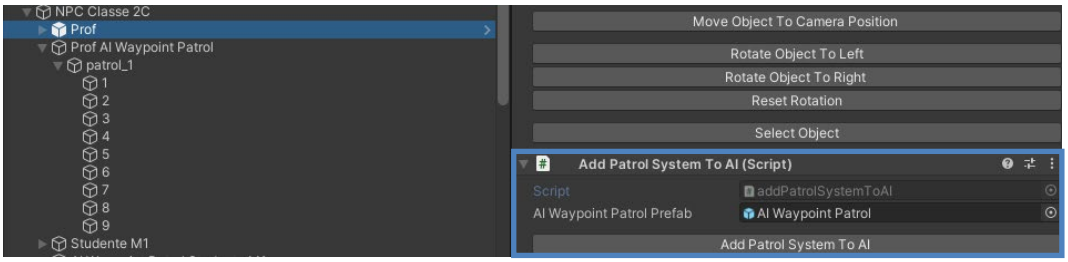


Figura 63. Aggiunta Patrol System

I Patrol System, in alternativa, possono anche essere copiati e incollati a un personaggio dopo che sono stati creati per la prima volta. Se si segue questo metodo di copia e incolla è importante, oltre a rinominare i Patrol System in modo da renderli riconoscibili, che i Components presenti nell'Inspector Window di AI Waypoint Patrol siano correttamente assegnati ai rispettivi Patrol Path, AI Character e Main AI Navmesh. In caso contrario si possono trascinare quelli corretti all'interno delle rispettive caselle.

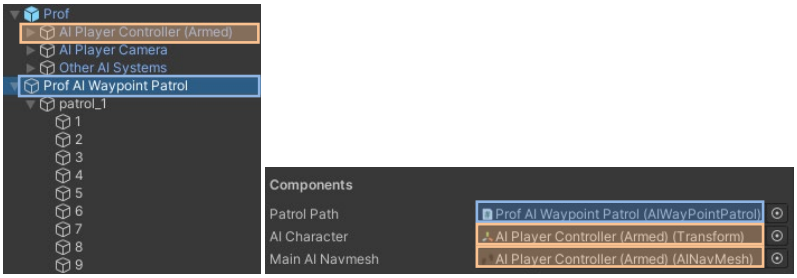


Figura 64. Assegnazione Components

Il passo successivo è quello di posizionare i punti del Patrol System in ordine secondo un percorso definito. Per visualizzare i punti, rappresentati da delle sfere, è necessario attivare i Gizmos, che vengono utilizzati per fornire ausili visivi per il debug o l'impostazione nella scena. I punti sono stati disposti in modo che i personaggi riescano a raggiungerli facilmente senza trovare ostacoli e senza che si intralcino con quelli di altri personaggi. Per posizionare i punti basta spostarli lungo

gli assi x, y e z in modo che siano nel punto desiderato della scena. Quando il gioco è in esecuzione questi punti non sono visibili, per questo è necessario attivare i Gizmos per vederli nella fase di sviluppo. I punti vengono posizionati in mezzo alla porta della classe, in modo che il personaggio ci passi attraverso, davanti alla porta antincendio del corridoio, dove gli studenti devono mettersi in fila e aspettare la partenza del professore, su ogni rampa di scala, così che non escano da percorso stabilito, e infine fuori dalla scuola, dove è presente il checkpoint.

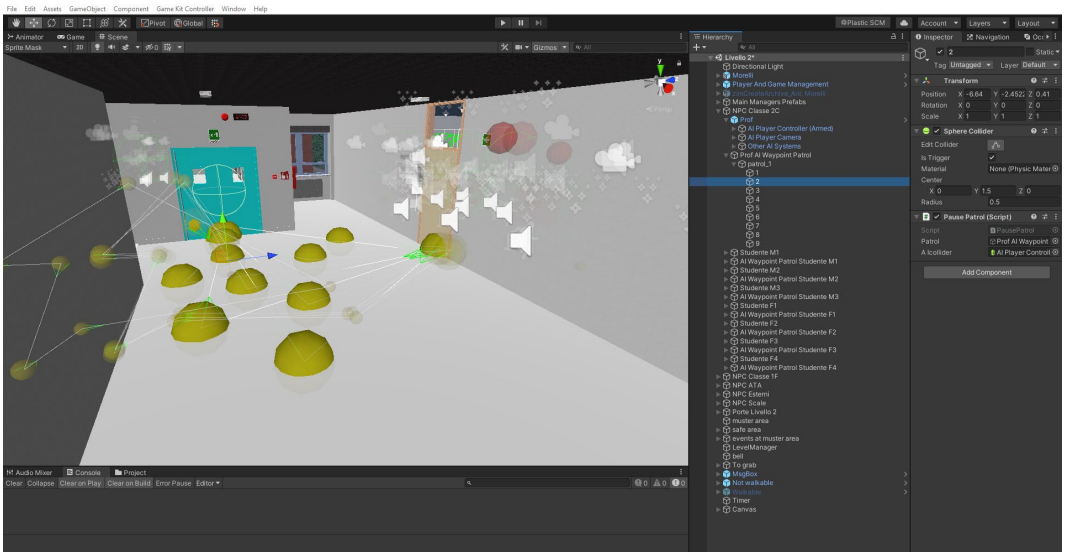


Figura 65. Disposizione dei punti del Patrol System

Il percorso è ora definito, ma bisogna far in modo che i personaggi camminino per raggiungere i punti. Per far muovere i personaggi, cliccando su Level Manager in Hierarchy, si apre la finestra Inspector, nella quale si deve indicare il numero di personaggi con Patrol System e successivamente collegare il rispettivo AI Player Controller. Inoltre, è necessario cliccare su Events At Muster Area, in Hierarchy, e su Detect Prof Moving, in questo modo si apre la finestra Inspector nella quale si seleziona dall'elenco in discesa Runtime Only e si collega l'AI Waypoint Patrol del personaggio che si vuole far camminare, infine, sempre da un elenco a discesa, si seleziona AIPatrolSystem.resumePatrolStateOnAi().

Figura 66. Assegnazione AI Waypoint Patrol a Level Manager

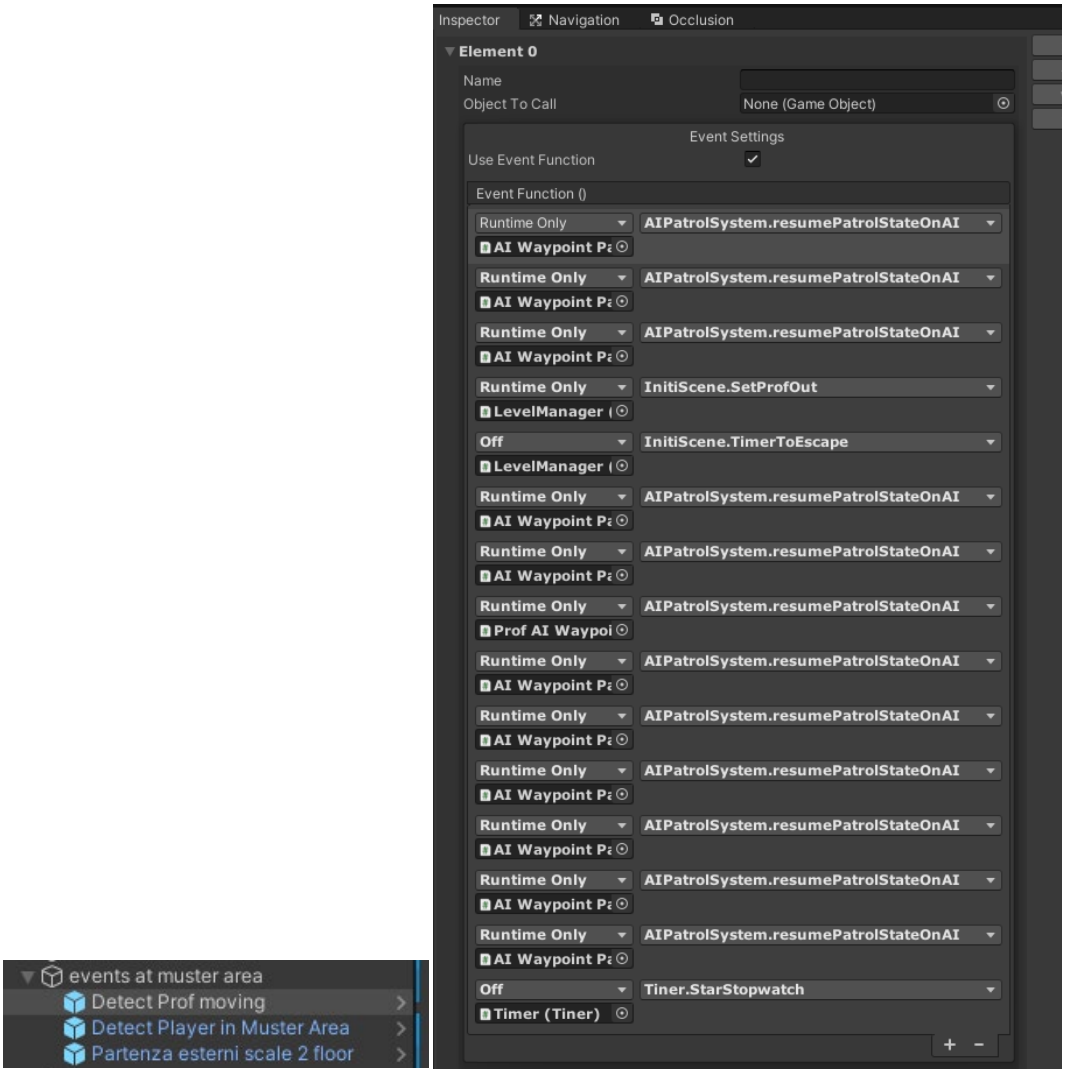


Figura 67. Assegnazione AI Waypoint Patrol a Events At Muster Area

Prendere oggetti

Quando il giocatore si trova in classe e suona la campanella di emergenza ha la possibilità di prendere in mano un libro sul banco premendo il tasto (T) della tastiera. All'Object libro è collegato lo script Grab Physical Object System, che consente al giocatore di prendere in mano il libro. Nel caso in cui il giocatore prende in mano il libro comparirà un messaggio di perdita del livello poiché il giocatore ha prelevato il libro e non avrebbe dovuto, dato che in situazioni di emergenza bisogna lasciare tutti gli oggetti. Per far terminare il livello è stato inserita una parte di codice all'interno dello script InitiScene. Una volta visualizzato il messaggio il giocatore potrà ricominciare il livello dall'inizio.

```
public void ObjectGrabbed() //controllo raccolta oggetti
    {

        messageToShow = "Ti sei attardato a raccogliere oggetti, potresti
non arrivare in tempo all'uscita!\nRiprova";
        ShowMessage(messageToShow);
        StartCoroutine(GenericTimer(5));
```

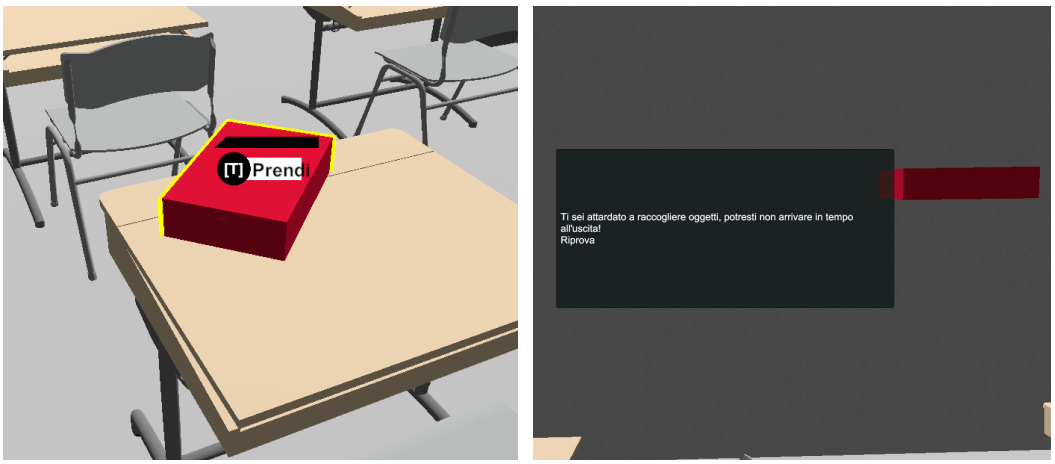


Figura 68. Possibilità di prendere il libro e messaggio di perdita del gioco dopo aver preso il libro

Animazione apertura porte

Quando il professore si alza al suono della campanella di emergenza si dirige verso la porta della classe e, successivamente, verso le porte antipanico del piano e quella dell'uscita dalla scuola, queste si aprono. Per l'apertura delle porte inizialmente è stato provato ad applicare una animazione che consentisse l'apertura. Purtroppo, non è stato possibile seguire questo metodo, poiché il modello della porta è composto dall'anta e dal telaio, e non è stato possibile separarli. Quindi, applicando l'animazione alla porta si apriva non solo l'anta, ma anche il telaio. Per risolvere questo problema è stato necessario copiare la geometria della porta e creare un Object che rappresentasse la sola anta e inserire all'interno di questa un cardine che ne permettesse l'apertura nel senso dell'esodo. Alla porta è stato collegato lo script Door System, che di impostare alcuni parametri come la velocità di apertura della porta e l'assegnazione ad essa di un suono di apertura e di chiusura. Alla porta è stata poi assegnata una Allowed Tag List, ovvero una lista con inseriti i personaggi che possono aprire la porta.

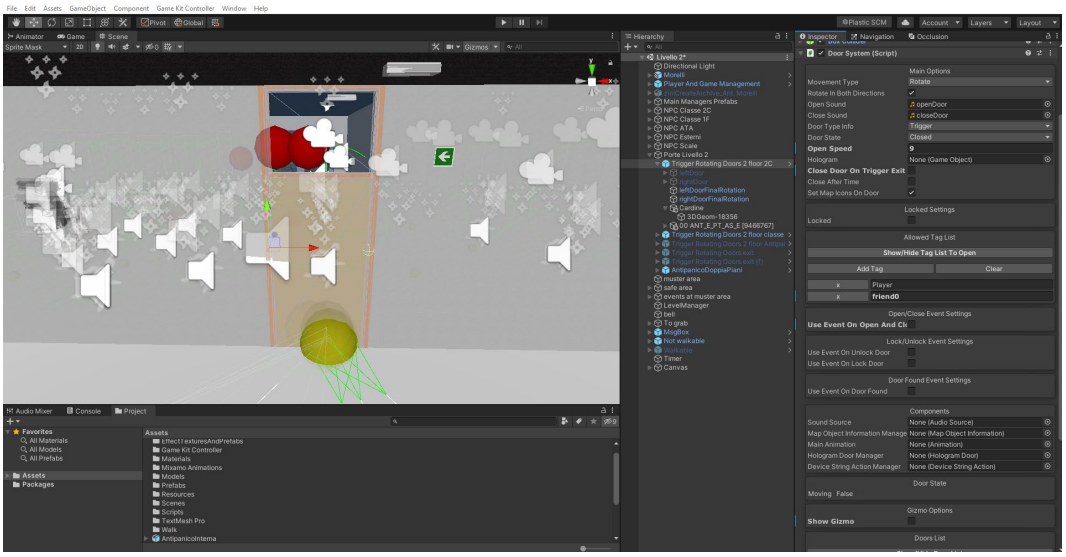


Figura 69. Creazione animazione porta della classe

La stessa metodologia è stata seguita anche per le porte antipanico presenti lungo il corridoio della classe e al piano terra che consentono l'uscita dalla scuola. In questo caso però, essendo la porta antipanico doppia, quindi composta da due ante, il

problema era si è presentato è che le due ante componevano un unico blocco. Per risolvere il problema è stato necessario aprire il modello della porta sul software Revit per dividere le ante e poi reimportare nuovamente il modello in Unity. Infine è stato possibile aggiungere i due cardini ai lati della porta in modo che le due ante potessero aprirsi nel senso dell'esodo.

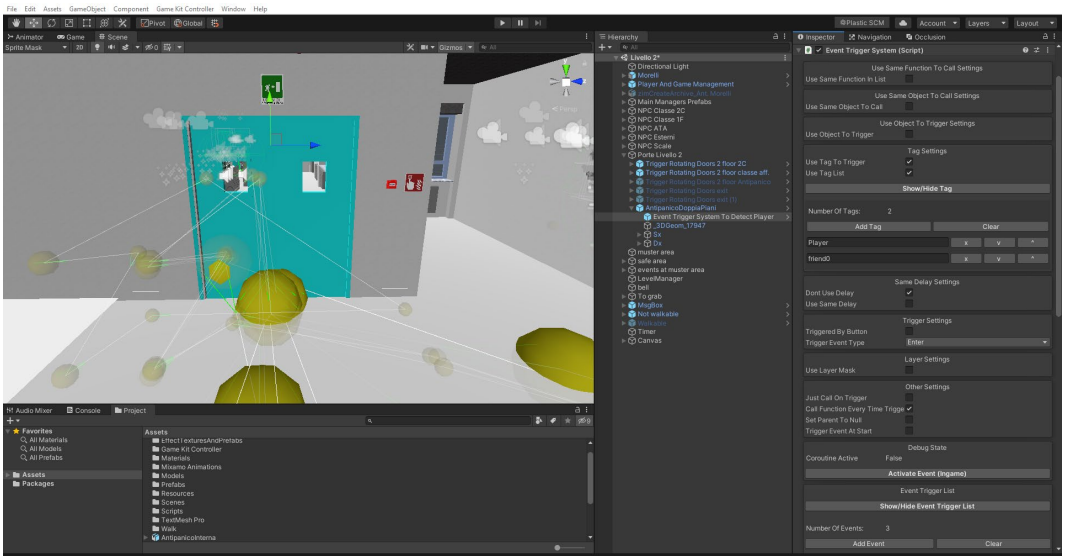


Figura 70. Creazione animazione porta antipanico

Inserimento dei fumi

All'interno dell'applicativo è stato inserito il fumo per rendere l'esperienza più realistica. Per farlo si sono presentate due possibilità: scaricare un asset dall'Unity Asset Store o crearlo tramite l'inserimento di un sistema di particelle (*Particle System*). Il primo metodo prevedeva il download di un asset dall'Asset Store, ma la maggior parte di questi erano a pagamento, mentre quelli gratuiti non erano soddisfacenti. È stato quindi creato il fumo da zero, tramite l'aggiunta di un Particle System, un sistema di particelle, all'interno della scena. Per fare in modo che il Particle System assomigli a fumo reale, è necessario scaricare e inserire all'interno della scena una immagine che rappresenti il fumo e creare un nuovo materiale, che verrà poi trascinato nella cartella Material all'interno del Renderer del Particle System.

Dopodiché vengono modificati i valori di base si possono modificare i valori secondari per mettere a punto la forma, il volume e il movimento delle particelle e dell'emettitore. Infine, è possibile scegliere la dimensione e la posizione del fumo all'interno della scena.

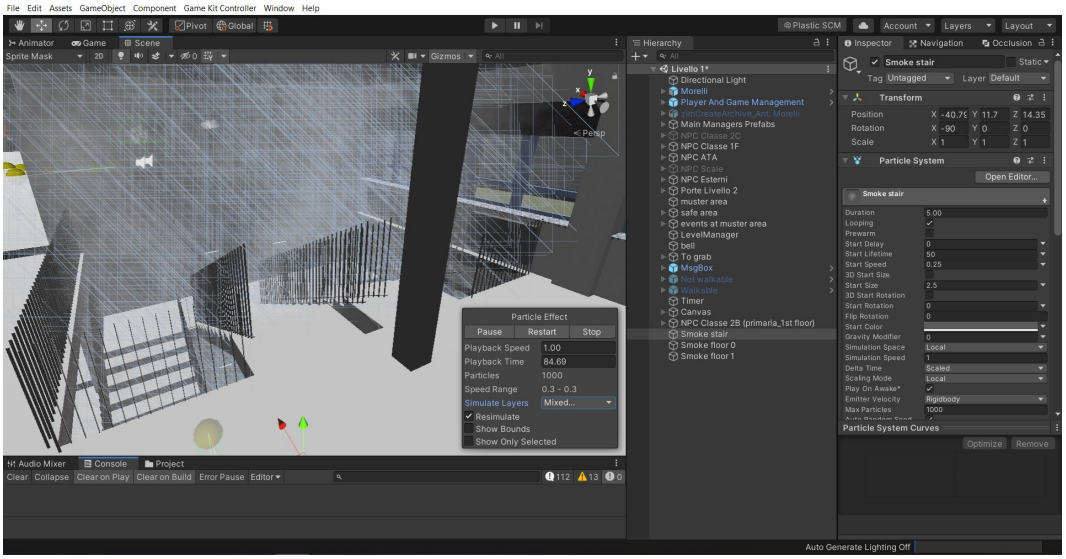


Figura 71. Inserimento dei fumi nella scena

Menù interattivo

Dopo aver creato le scene rappresentative dei due diversi livelli, si è creato il Menù interattivo che compare una volta avviato il gioco che consente di scegliere il livello al quale si vuole giocare. Il Menù rappresenta la scena principale perché è la connessione tra tutte le scene presenti. Per creare il Menù è stata creata una nuova scena, denominata Home Menù. Per visualizzare la scena è necessario passare alla vista 2D. L'immagine di copertina del Menu è stata scaricata da Internet e poi inserita all'interno della cartella Resources presente negli Assets. Successivamente, l'immagine è stata trascinata all'interno dell'Inspector Window in Raw Image. Dopo aver inserito lo sfondo è stato scritto in Title il titolo del videogioco e dopodiché si è passati alla creazione del Menu vero e proprio, in cui il giocatore può scegliere tra quattro opzioni:

- Livello Elementari: consente il caricamento della scena del primo livello;
- Livello Medie: consente il caricamento della scena del secondo livello;

- Impostazioni: consente il caricamento della scena in cui il giocatore può modificare le impostazioni del videogioco;
- Esci: consente il caricamento della scena in cui viene chiesto al giocatore se vuole uscire, il quale può scegliere tra Si o No.

Il *Livello Elementari* e il *Livello Medie*, se cliccati, portano alla scena del gioco prescelto. Per fare che questo accada è necessario collegare, all'interno dell'Inspector Window, sotto la voce Button, la scena corrispondente al livello selezionato.

All'interno del Menù, oltre ai bottoni che portano ai livelli del gioco, sono stati inseriti anche il pulsante *Impostazioni* e il pulsante *Esci*, che se cliccati portano a due schermate differenti. Cliccando Impostazioni si apre un nuovo menù in cui il giocatore può scegliere tra alcune opzioni in base alle preferenza di gioco. È possibile modificare il volume del gioco, la sensibilità del mouse e invertire la direzione dei movimenti e della camera. Cliccando su Esci, invece, comparirà una schermata che chiede al giocatore se vuole uscire: in caso negativo il gioco proseguirà, mentre in caso affermativo il gioco si chiuderà e si tornerà alla schermata principale del desktop.

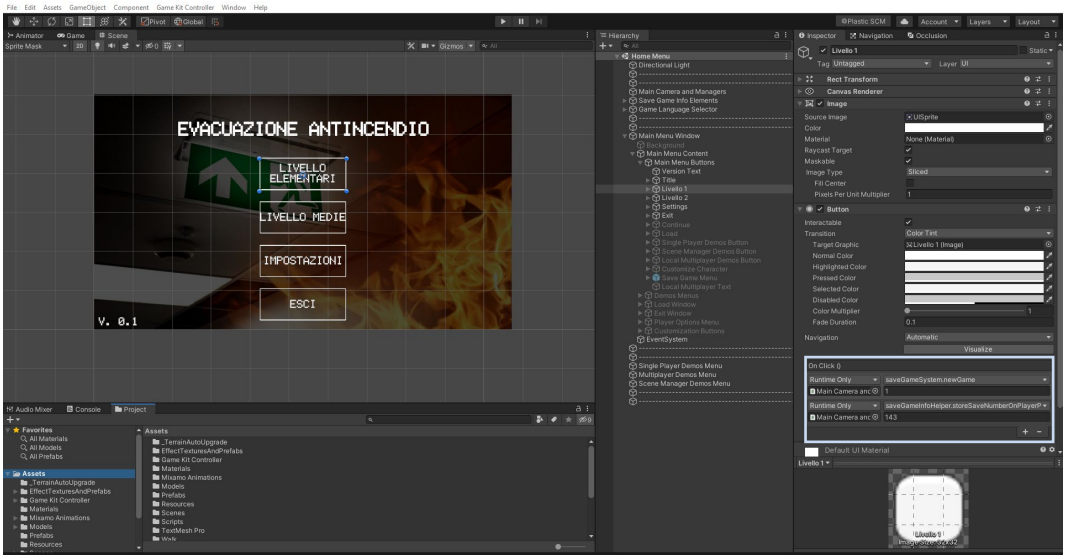


Figura 72. Collegamento del Livello alla scena corrispondente

Figura 73. Home Menu all'avvio dell'applicativo

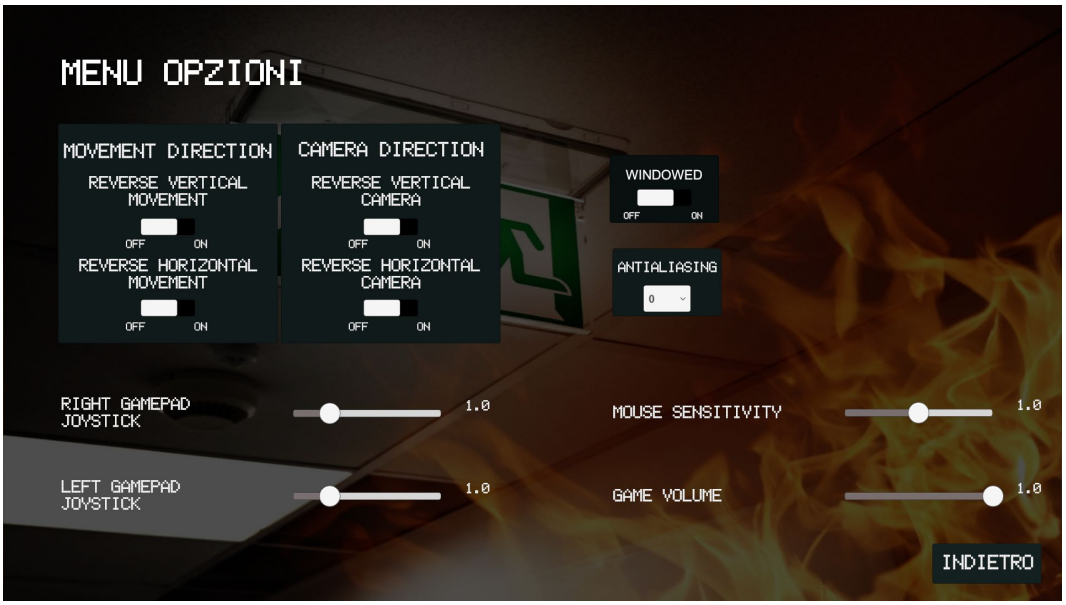


Figura 74. Menu Opzioni dell'applicativo

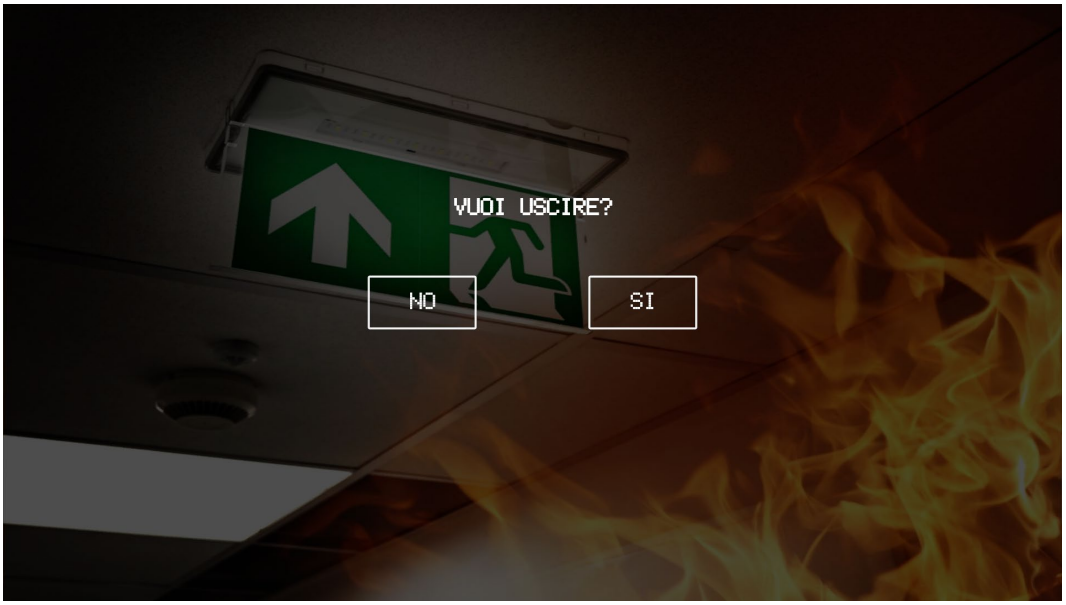


Figura 75. Menu di Uscita dell'applicativo

Build

L'ultima fase è la creazione vera e propria dell'applicativo, ovvero del videogioco utilizzabile dallo studente. Con Unity è possibile creare l'applicativo per piattaforme diverse. Per selezionare la piattaforma e le impostazioni e avviareil processo di compilazione si accede a *Build Settings* cliccando su File dal menu principale di Unity. Unity produce due tipi di build:

- Release build: include solo ciò che è necessario per eseguire l'applicazione ed è un tipo di build predefinito;
- Development build: include lo scripting debug e il profiler. In questo caso viene abilitato un set aggiuntivo di opzioni.

Quando si vuole creare l'applicativo Unity unisce tutte le scene in *Scenes in Build* e le scene vanno messe in ordine e si possono aggiungere o eliminare dall'elenco. Per l'applicativo le scene inserite sono quelle, in ordine, dell'Home Menu, del Livello Elementari e del Livello Medie.

Ogni build deve avere una piattaforma di destinazione e il riquadro *Platform* elenca tutte le piattaforme, che sono diverse da computer a computer. Quando viene scelta

la piattaforma Unity mostra un elenco di opzioni per la modifica della compilazione. Per questo applicativo la piattaforma scelta è il PC.

Per creare l'applicazione, infine, è possibile scegliere tra due opzioni:

- Build: viene creato l'applicativo e viene aperto nella cartella selezionata in fase di salvataggio;
- Build and Run: viene creato l'applicativo che viene aperto immediatamente una volta finita la compilazione.

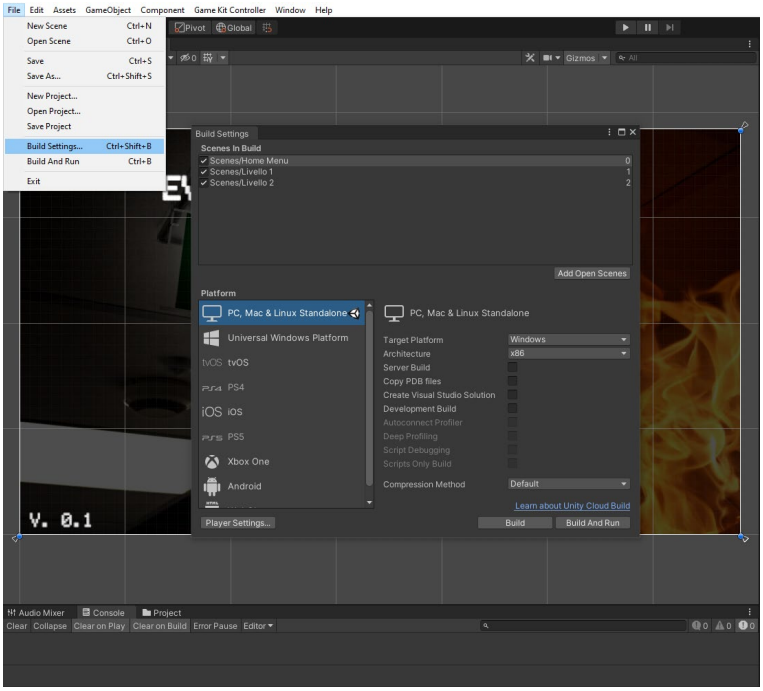


Figura 76. Build Settings

Script

Vengono ora inseriti tutti gli script per lo sviluppo dell'applicativo.

InitiScene:

```
using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;

public class InitiScene : MonoBehaviour
{
    [Space]
    [Header("Npc Settings")]
    [Space]
    public List<Animator> studentAnimators;
    public GameObject ProfessorPatrol;
    public float ProfessorSpeed = 0.11f;
    public List<GameObject> allPatrols;
    public float patrolSpeed = 1f;
    public float returnToPatrolTime = 0.3f;
    [Space]
    [Header("Player Settings")]
    [Space]
    public GameObject player;
    //public Collider playerCollider;
    //public GameObject colliderPosition;
    //public// float timeInMusterArea = 5; sostituito dal delay nel trigger
    [Space]
    [Header("General Settings")]
    [Space]
    public AudioSource bellRing;
    public GameObject timer;
    public GameObject popUpBox;
    public List<GameObject> objectToDisable;
    public Text guiText;
    public float timeToStart = 20f;
    //public float instructionTime = 20f;
    public float timeProfIsWaiting = 10;
    public float timeProfToStart = 2f;
    public float timeToEscape = 60;
    string instructions = "Questa è una situazione di emergenza, è scoppiato
       un incendio.\nQuando suonerà l'allarme dovrai seguire con i tuoi
      compagni il professore.\nNon dovrai raccogliere gli effetti personali
      e non dovrai correre, dovrai rimanere in fila con la tua classe fino a
       quando arriverai nel luogo sicuro fuori dalla scuola.\n\nQueste sono
      le istruzioni:\nW = AVANTI    S = INDIETRO\nA = SINISTRA    D = DESTRA
      \nSPAZIO = SALTA    T = AFFERRA/GETTA    E = APRI/CHIUDI";
    private bool winGame = false;
    private string messageToShow;
    private bool mustRestart = false;
    private  bool bellIsRinging = false;
    float currentTime;
    bool watchIsOn = false;
    private bool profIsOut = false;
    float currentTimeProfToStart;
```

```
48      //inputManager input;
49      AIPatrolSystem patrolProf;
50      menuPause restart;
51      //playerInputManager movement;
52      // Start is called before the first frame update
53      void Start()
54      {
55          //playerCollider = player.GetComponent<Collider>();
56          //input = player.GetComponent<inputManager>();
57          restart = player.GetComponentInChildren<menuPause>();
58          patrolProf = ProfessorPatrol.GetComponent<AIPatrolSystem>();
59          //movement = player.GetComponentInChildren<playerInputManager>();
60          currentTime = 0;
61          Activation();
62      }
63      void Activation()
64      {
65          SitDown();
66          ShowTimeMessage(instructions, timeToStart);
67          StartCoroutine(Waiting());
68      }
69      public void SetProfOut()
70      {
71          profIsOut = true;
72      }
73      public void SetWalk(float speed)
74      {
75          //playerController playercontroller =
              player.GetComponent<playerController>();
76          player.GetComponentInChildren <playerController>().setWalkSpeedValue
              (speed);
77          if (speed > 0.01)
78              player.GetComponentInChildren<playerController>
                  ().setIncreaseWalkSpeedEnabled(true);
79          else
80              player.GetComponentInChildren<playerController>
                  ().setIncreaseWalkSpeedEnabled(false);
81      }
82      private void SitDown()
83      {
84          foreach(Animator anim in studentAnimators)
85          {
86              int layerID = anim.GetLayerIndex("Base Layer");
87              anim.Play("Sit On Chair", layerID);
88          }
89      }
90      private void StandUp()
91      {
92          foreach (Animator anim in studentAnimators)
93          {
94              int layerID = anim.GetLayerIndex("Base Layer");
95              anim.Play("Get Up From Chair", layerID);
96          }
```

```
97      }
98      public void StartStopwatch()
99      {
100         currentTime = 0;
101         watchIsOn = true;
102     }
103     void RestartPatrols()
104     {
105         // quando gli npc si trovano fuori dalla classe ed arriva il
              giocatore parte un'attesa di timeInMusterArea
106         foreach (GameObject npc in allPatrols)
107         {
108             AIPatrolSystem patrol = npc.GetComponent<AIPatrolSystem>();
109             if (patrol)
110             {
111                 patrol.patrolSpeed = patrolSpeed;
112                 patrol.returnToPatrolSpeed = returnToPatrolTime;
113                 patrol.resumePatrolStateOnAI();
114             }
115         }
116         patrolProf.patrolSpeed = ProfessorSpeed;
117         foreach (GameObject item in objectToDisable)
118         {
119             item.SetActive(false);
120         }
121     }
122     public void TimerToEscape() //avvia timer tempo per fuggire
123     {
124             StartCoroutine(CheckEscapeTimer(timeToEscape));
125     }
126     // 3 controlli error checking
127     public void TimeOver()   //controllo tempo per fuggire
128     {
129         messageToShow = "Hai perso troppo tempo e non sei riuscito a
              raggiungere l'uscita nel tempo previsto.\nRiprova";
130         // show message
131         ShowMessage(messageToShow);
132         StartCoroutine(GenericTimer(5));
133     }
134     public void ObjectGrabbed() //controllo raccolta oggetti
135     {
136         messageToShow = "Ti sei attardato a raccogliere oggetti, potresti
              non arrivare in tempo all'uscita!\nRiprova";
137         ShowMessage(messageToShow);
138         StartCoroutine(GenericTimer(5));
139     }
140     public void ColumnLeaving() // controllo fila
141     {
142         if (profIsOut)
143         {
144             messageToShow = "Hai lasciato la fila dei tuoi compagni,
                  potresti non arrivare in tempo all'uscita!\nRiprova";
145             ShowMessage(messageToShow);
```

```
146             StartCoroutine(GenericTimer(5));
147         }
148     }
149     public void WaitInCorridor()
150     {
151         messageToShow = "Attendi qui la ripartenza del gruppo.\nPremi Invio
              per chiudere questo messaggio";
152         ShowMessage(messageToShow);
153     }
154     public void WinGame(bool win) // controllo vittoria
155     {
156         if (win && !mustRestart)
157         {
158             winGame = true;
159             messageToShow = "Hai vinto! Sei riuscito ad arrivare in tempo
                  all'uscita.";
160             ShowMessage(messageToShow);
161             StartCoroutine(QuitLevel());
162         }
163     }
164     /*
165     public void ResumeInputOnPlayer(bool resume)
166     {
167         movement.inputEnabled = resume;
168     }
169     */
170     public void ShowMessage(string msg)
171     {
172         SetWalk(0);
173         StartCoroutine(Popup(msg));
174     }
175     public void ShowTimeMessage(string msg, float time)
176     {
177         SetWalk(0);
178         StartCoroutine(PopupTimer(msg, time));
179     }
180     // coroutines
181     IEnumerator Popup(string msg) //messaggi popo up
182     {
183         guiText.text = msg;
184         popUpBox.SetActive(true);
185         //yield return new WaitForSeconds(10);
186         yield return new WaitUntil(() => Input.GetKey(KeyCode.Return));
187         popUpBox.SetActive(false);
188     }
189     IEnumerator PopupTimer(string msg, float time) //messaggi popo up
190     {
191         guiText.text = msg;
192         popUpBox.SetActive(true);
193         yield return new WaitForSecondsRealtime(time);
194        // yield return new WaitUntil(() => Input.GetKey(KeyCode.Return));
195         popUpBox.SetActive(false);
196     }
```

```
197     IEnumerator Waiting() // inizio gioco
198     {
199         // attende timeToStart secondi reali prima di iniziare
200         yield return new WaitForSecondsRealtime(timeToStart);
201         bellRing.Play();
202         StandUp();
203         timer.GetComponent<Tiner>().StarStopwatch();
204         TimerToEscape();
205         //input.enabled = true; spostato in trigger in muster area o porta
              classe
206         if (patrolProf)
207         {
208             //Debug.Log("patrol found");
209             //patrolProf.resumePatrolStateOnAI();
210             bellIsRinging = true;
211             messageToShow = "Mantenete la calma e seguitemi.";
212             ShowTimeMessage(messageToShow, timeProfToStart);
213         }
214     }
215     IEnumerator GenericTimer(float seconds) // timer generico
216     {
217         yield return new WaitForSecondsRealtime(seconds);
218         mustRestart = true;
219     }
220     IEnumerator CheckEscapeTimer(float seconds) // timer fuga
221     {
222         yield return new WaitForSecondsRealtime(seconds);
223         if (!winGame)
224         {
225             TimeOver();
226         }
227     }
228     IEnumerator QuitLevel()
229     {
230         yield return new WaitForSecondsRealtime(5);
231         SceneManager.LoadScene(0);
232     }
233     private void Update()
234     {
235         if (mustRestart)
236         {
237             restart.restart();
238         }
239         if (watchIsOn)
240         {
241             currentTime += Time.deltaTime;
242             if (currentTime >= timeProfIsWaiting)
243             {
244                 watchIsOn = false;
245                 RestartPatrols();
246                 SetWalk(1f);
247             }
248         }
```

```
249         if (bellIsRinging)
250         {
251             currentTimeProfToStart += Time.deltaTime;
252             if(currentTimeProfToStart >= timeProfToStart)
253             {
254                 bellIsRinging = false;
255                 patrolProf.resumePatrolStateOnAI();
256             }
257         }
258     }
259 }
260
```

AIPatrolSystem

```
1  using UnityEngine;
2  using System.Collections;
3  using System.Collections.Generic;
4
5  public class AIPatrolSystem : MonoBehaviour
6  {
7      [Header ("Main Settings")]
8      [Space]
9      public bool paused;
10     public float minDistanceToNextPoint = 0.6f;
11     public float patrolSpeed = 0.2f;
12     public float returnToPatrolSpeed = 1;
13     public bool loop = true;
14     [Space]
15     [Header ("Patrol Time Settings")]
16     [Space]
17     public bool useGeneralWaitTime = true;
18     public float generalWaitTimeBetweenPoints;
19     public bool moveBetweenPatrolsInOrder = true;
20     public float fixedTimeToChangeBetweenPatrols;
21     [Space]
22     [Header ("Random Patrol Settings")]
23     [Space]
24     public bool changeBetweenPointRandomly = true;
25     [Range (0, 10)] public int changeRandomlyProbability = 1;
26     public bool useTimeToChangeBetweenPointRandomly;
27     public bool useRandomTimeToChangePatrol;
28     public Vector2 randomTimeLimits;
29     [Space]
30     [Header ("Debug")]
31     [Space]
32     public bool returningToPatrol;
33     public bool AIIsDestroyed;
34     [Space]
35     [Header ("Gizmo Settings")]
36     [Space]
37     public bool showGizmo;
38     public float gizmoRadius;
39     [Space]
40     [Header ("Components")]
41     [Space]
42     [Tooltip ("An AIWayPointPatrol path in your scene that this AI should
         follow. To create one add a GameObject with AIWayPointPatrol component
          and edit the Way Points then drag the GameObject here.")]
43     public AIWayPointPatrol patrolPath;
44     public Transform AICharacter;
45     public AINavMesh mainAINavmesh;
46     bool patrolAssigned;
47     Transform currentWayPoint;
48     int currentPatrolIndex = 0;
49     int currentWaypointIndex = 0;
50     Coroutine movement;
51     bool settingNextPoint;
```

```
52      float lastTimeChanged;
53      float distanceToPoint;
54      Transform closestWaypointTransform;
55      void Start ()
56      {
57          if (AICharacter == null) {
58              AICharacter = transform;
59          }
60          if (mainAINavmesh == null) {
61              mainAINavmesh = AICharacter.GetComponent<AINavMesh> ();
62          }
63          if (patrolPath) {
64              setClosestWayPoint ();
65          }
66      }
67      void Update ()
68      {
69          if (!paused && patrolAssigned && !settingNextPoint)
70          {
71              if (AIIsDestroyed) {
72                  enabled = false;
73                  return;
74              }
75              if (!mainAINavmesh.isPatrolPaused ())
76              {
77                  if (AICharacter == null) {
78                      AIIsDestroyed = true;
79                      return;
80                  }
81                  distanceToPoint = GKC_Utils.distance (AICharacter.position,
                      currentWayPoint.position);
82                  if (distanceToPoint < minDistanceToNextPoint) {
83                      if (returningToPatrol) {
84                          mainAINavmesh.setPatrolSpeed (patrolSpeed);
85                          returningToPatrol = false;
86                      }
87                      bool setRandomWayPoint = false;
88                      if (changeBetweenPointRandomly) {
89                          int changeOrNotBool = Random.Range (0,
                            (changeRandomlyProbability + 1));
90                          if (changeOrNotBool == 0) {
91                              setRandomWayPoint = true;
92                              //print ("random waypoint");
93                          }
94                      }
95                      if (setRandomWayPoint) {
96                          setNextRandomWaypoint ();
97                      } else {
98                          setNextWaypoint ();
99                          //print ("in order");
100                     }
101                 }
102                 if (changeBetweenPointRandomly) {
```

```
103                     if (useTimeToChangeBetweenPointRandomly) {
104                         if (useRandomTimeToChangePatrol) {
105                         } else {
106                             if (Time.time > fixedTimeToChangeBetweenPatrols
                        + lastTimeChanged) {
107                                 lastTimeChanged = Time.time;
108                                 setNextRandomWaypoint ();
109                                 //print ("random waypoint");
110                             }
111                         }
112                     }
113                 }
114             }
115         }
116     }
117     public void pauseOrPlayPatrol (bool state)
118     {
119 //      print ("patrol paused");
120         paused = state;
121     }
122     public bool isPatrolPaused ()
123     {
124         return paused;
125     }
126     public Transform closestWaypoint (Vector3 currentPosition)
127     {
128         float distance = Mathf.Infinity;
129         for (int i = 0; i < patrolPath.patrolList.Count; i++) {
130             for (int j = 0; j < patrolPath.patrolList [i].wayPoints.Count; j
                ++) {
131                 float currentDistance = GKC_Utils.distance (currentPosition,
                    patrolPath.patrolList [i].wayPoints [j].position);
132                 if (currentDistance < distance) {
133                     distance = currentDistance;
134                     currentPatrolIndex = i;
135                     currentWaypointIndex = j;
136                 }
137             }
138         }
139         closestWaypointTransform = patrolPath.patrolList
              [currentPatrolIndex].wayPoints [currentWaypointIndex];
140         return closestWaypointTransform;
141     }
142     public void setNextWaypoint ()
143     {
144         if (movement != null) {
145             StopCoroutine (movement);
146         }
147         settingNextPoint = false;
148         if (mainAINavmesh.isPatrolPaused ()) {
149             return;
150         }
151         movement = StartCoroutine (setNextWayPointCoroutine ());
```

```
152     }
153     IEnumerator setNextWayPointCoroutine ()
154     {
155         mainAINavmesh.removeTarget ();
156         settingNextPoint = true;
157         if (useGeneralWaitTime) {
158             yield return new WaitForSeconds (generalWaitTimeBetweenPoints);
159         } else {
160             yield return new WaitForSeconds
                (patrolPath.waitTimeBetweenPoints);
161         }
162         if (!mainAINavmesh.isPatrolPaused ()) {
163             currentWaypointIndex++;
164             if (currentWaypointIndex > patrolPath.patrolList
                [currentPatrolIndex].wayPoints.Count - 1 && loop) {
165                 currentWaypointIndex = 0;
166                 if (moveBetweenPatrolsInOrder) {
167                     currentPatrolIndex++;
168                     if (currentPatrolIndex > patrolPath.patrolList.Count -
                    1) {
169                         currentPatrolIndex = 0;
170                     }
171                 }
172             }
173             else
174             {
175                 //end of patrol
176             }
177             currentWayPoint = patrolPath.patrolList
                [currentPatrolIndex].wayPoints [currentWaypointIndex];
178             setCurrentPatrolTarget (currentWayPoint);
179         }
180         settingNextPoint = false;
181     }
182     public void setNextRandomWaypoint ()
183     {
184         if (movement != null) {
185             StopCoroutine (movement);
186         }
187         settingNextPoint = false;
188         if (mainAINavmesh.isPatrolPaused ()) {
189             return;
190         }
191         movement = StartCoroutine (setNextRandomWayPointCoroutine ());
192     }
193     IEnumerator setNextRandomWayPointCoroutine ()
194     {
195         mainAINavmesh.removeTarget ();
196         settingNextPoint = true;
197         if (useGeneralWaitTime) {
198             yield return new WaitForSeconds (generalWaitTimeBetweenPoints);
199         } else {
200             yield return new WaitForSeconds
```

```
                (patrolPath.waitTimeBetweenPoints);
201         }
202         if (!mainAINavmesh.isPatrolPaused ()) {
203             int currentWaypointIndexCopy = currentWaypointIndex;
204             int currentPatrolIndexCopy = currentPatrolIndex;
205             int checkBucle = 0;
206             if (patrolPath.patrolList.Count > 1) {
207                 while (currentPatrolIndexCopy == currentPatrolIndex) {
208                     currentPatrolIndex = Random.Range (0,
                        patrolPath.patrolList.Count);
209                     checkBucle++;
210                     if (checkBucle > 100) {
211                         //  print ("bucle error");
212                         break;
213                     }
214                 }
215             }
216             checkBucle = 0;
217             while (currentWaypointIndexCopy == currentWaypointIndex) {
218                 currentWaypointIndex = Random.Range (0,
                      patrolPath.patrolList
                      [currentPatrolIndex].wayPoints.Count);
219                 checkBucle++;
220                 if (checkBucle > 100) {
221                     //print ("bucle error");
222                     break;
223                 }
224             }
225             //print ("Next patrol: " + (currentPatrolIndex+1) + " and next
                  waypoint: " + (currentWaypointIndex+1));
226             currentWayPoint = patrolPath.patrolList
                  [currentPatrolIndex].wayPoints [currentWaypointIndex];
227             setCurrentPatrolTarget (currentWayPoint);
228         }
229         settingNextPoint = false;
230     }
231     public void setClosestWayPoint ()
232     {
233         if (paused) {
234             return;
235         }
236         patrolAssigned = true;
237         currentWayPoint = closestWaypoint (AICharacter.position);
238         setCurrentPatrolTarget (currentWayPoint);
239         mainAINavmesh.setPatrolSpeed (patrolSpeed);
240     }
241     public void setCurrentPatrolTarget (Transform newTarget)
242     {
243         mainAINavmesh.setPatrolTarget (newTarget);
244         mainAINavmesh.setPatrolState (true);
245     }
246     public void setReturningToPatrolState (bool state)
247     {
```

```
248         returningToPatrol = true;
249         if (returningToPatrol) {
250             mainAINavmesh.setPatrolSpeed (returnToPatrolSpeed);
251         }
252     }
253     public void resumePatrolStateOnAI ()
254     {
255         pauseOrPlayPatrol (false);
256         setClosestWayPoint ();
257     }
258     public void pausePatrolStateOnAI ()
259     {
260         pauseOrPlayPatrol (true);
261         mainAINavmesh.setPatrolState (false);
262     }
263     void OnDrawGizmos ()
264     {
265         if (!showGizmo) {
266             return;
267         }
268         if (GKC_Utils.isCurrentSelectionActiveGameObject (gameObject)) {
269             DrawGizmos ();
270         }
271     }
272     void OnDrawGizmosSelected ()
273     {
274         DrawGizmos ();
275     }
276     void DrawGizmos ()
277     {
278         if (showGizmo) {
279             Gizmos.color = Color.red;
280             Gizmos.DrawSphere (patrolPath.patrolList
                  [currentPatrolIndex].wayPoints
                  [currentWaypointIndex].transform.position, gizmoRadius);
281         }
282     }
283 }
```

PausePatrol:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PausePatrol : MonoBehaviour
{
    public GameObject patrol;
    public Collider AIcollider;
    private void OnTriggerEnter(Collider other)
    {
        if(other == AIcollider)
        {
            patrol.GetComponent<AIPatrolSystem>().pausePatrolStateOnAI();
            if (other.gameObject.CompareTag("friend0"))
            {
                Animator anim = other.GetComponentInParent<Animator>();
                if (anim)
                {
                    int index = anim.GetLayerIndex("Base Layer");
                    Debug.Log("anim found" + index);
                    anim.Play("Stand Half Turn Left", index);

                }
            }
        }
    }
    // Start is called before the first frame update
    void Start()
    {
    }
    // Update is called once per frame
    void Update()
    {
    }
}
```

CheckIsolated:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CheckIsolated : MonoBehaviour
{
    public float maxTimeIsolated = 3;
    public GameObject levelManager;
    int friendCounter = 0;
    float emptyTime = 0;
    float startTime;
    // Start is called before the first frame update
    void Start()
    {
        startTime = Time.time;
        //Debug.Log("collider enabled");
    }
    private void OnTriggerEnter(Collider other)
    {
        if (other.gameObject.CompareTag("friend0") ||
          other.gameObject.CompareTag("friend1"))
        {
            friendCounter++;
            Debug.Log("entered " + friendCounter);
        }
    }
    private void OnTriggerExit(Collider other)
    {
        if (other.gameObject.CompareTag("friend0") ||
          other.gameObject.CompareTag("friend1"))
        {
            friendCounter--;
            Debug.Log("exited " + friendCounter);
        }
    }
    // Update is called once per frame
    private void FixedUpdate()
    {
        if (friendCounter == 0)
        {
            emptyTime = Time.time - startTime;
            Debug.Log("tempo " + emptyTime + "counter " + friendCounter);
            if (emptyTime > maxTimeIsolated)
            {
                levelManager.GetComponent<InitiScene>().ColumnLeaving();
            }
        }
        else
        {
            startTime = Time.time;
    }
  }
}
```

Timer:

```
using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class Tiner : MonoBehaviour
{
    public Text timerText;
    float currentTime;
    bool stopwatchActive = false;

    // Start is called before the first frame update
    void Start()
    {
        currentTime = 0;
    }

    // Update is called once per frame
    void Update()
    {
        if (stopwatchActive)
        {
            currentTime += Time.deltaTime;
        }
        TimeSpan time = TimeSpan.FromSeconds(currentTime);
        timerText.text = time.ToString(@"mm\:ss\:fff");
    }
    public void StarStopwatch()
    {
        stopwatchActive = true;
    }
    public void StopStopwatch()
    {
        stopwatchActive = false;
    }
}
```

MessageBox:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class MyMessageBox : MonoBehaviour
{
    // call with
    // MessageBox mb = new MessageBox("This is a test message!");
    public MyMessageBox(string msg)
    {
        GameObject obj = (GameObject)MonoBehaviour.Instantiate(Resources.Load
          (@"MsgBox"));
        obj.GetComponentInChildren<Text>().text = msg;
    }
}
/* inserire nello script di call
void Start()
{
    StartCoroutine("Do");
}
IEnumerator Do()
{
    MessageBox mb = new MessageBox("This is also a test message!");
    yield return new WaitUntil(() => GlobalVariables.MsgBoxClicked == true);
    //GlobalVariables is a separate class containing some static variables.
    Debug.Log("Done!");
}
*/
```

CONCLUSIONI

In questo lavoro di tesi è stato usato un approccio basato sui Serious Games per addestrare gli occupanti di un edificio alle abilità personali di sicurezza antincendio. Il gioco si concentra non solo sull'acquisizione da parte dei giocatori delle conoscenze necessarie per evacuare un edificio, ma più in generale consente loro di apprendere e praticare alcune delle azioni e delle procedure richieste nelle emergenze antincendio.

Come sviluppo sarebbe interessante far provare il gioco all'interno del plesso Ettore Morelli e avere un feedback da parte degli studenti. Un altro sviluppo futuro riguarderebbe come rendere il giocatore più consapevole dello stato fisiologico del suo personaggio (ad esempio, forza, infortuni,...) e dell'ambiente (ad esempio, calore, fumo altamente tossico,...) nel gioco. In alcuni giochi tali dimostrazioni sono state ottenute ad esempio oscurando le immagini, incapacità di eseguire movimenti precisi, display tremante. Lo stesso modo può essere utilizzato per simulare la perdita di coscienza a causa di fumi tossici.

Un'altra possibile analisi potrebbe essere quella di valutare quanto e in che modo il gioco riesca a coinvolgere i giocatori. Per fare ciò, si potrebbe utilizzare un questionario.

Sarebbe inoltre utile valutare quanto la conoscenza acquisita nel gioco si trasferisca nel mondo reale. Per fare ciò, si potrebbero organizzare simulazioni dal vivo di alcuni scenari di gioco nell'edificio reale. Infine, sarebbe interessante di sperimentare livelli di gioco multiutente. In questo caso, oltre al vantaggio di generare situazioni più realistiche, gli scenari potrebbero anche considerare ruoli diversi per i giocatori (professore, personale ATA).

BIBLIOGRAFIA E SITOGRAFIA

Bibliograifa

[1] Sime J. D., 1984, *Escape Behaviour in Fires : 'Panic' or Affiliation?*, University of Surrey

[2] Julien T.U.S, Shaw C.D., 2003, *Firefighter command training virtual environment*, Proceedings of the Conference on Diversity in Computing, Atlanta, Georgia

[3] Tanoni I., 2003, *Videogiocando s'impara. Dal divertimento puro all'insegnamento-apprendimento*, Erickson

[4] Santos G., Aguirre B.E., 2004, *A critical review of emergency evacuation simulation models*, in: NIST Workshop on Building Occupant Movement during Fire Emergencies

[5] Backlund P., Engstrom H., Hammar C., Johannesson M., Lebram M.,2007, *Sidh – a Game Based Firefighter Training Simulation*, 11th International Conference Information Visualization

[6] Trenholme D., Smith S.P., 2008, *Computer game engines for developing first-person virtual environments*

[7] Tavares R.M., 2008, *Evacuation Processes Versus Evacuation Models: "Quo Vadimus"?*, Fire Technol

[8] Woksepp S., Olofsson T., 2008, *Credibility and applicability of virtual reality models in design and construction*, Advanced Engineering Informatics

[9] Harteveld C., 2011, *Triadic Game Design – Balancing Reality, Meaning and Play*, Springer, London

[10] Nesti R., 2012, *Frontiere attuali del gioco. Per una lettura pedagogica*, Unicopli

[11] Larusdottir A.R., 2013, *Evacuation of Children, Focusing on daycare centers and elementary schools*, Technical University of Denmark

[12] Penzentcev A. 2015, *Architecture and implementation of the system for serious games in Unity 3D*, Masarykova Univerzita

[13] Nesi L., 2017, *Il videogioco: un'opportunità per l'apprendimento*, Università degli Studi di Firenze

[14] Zanut S., Ronchi E., Cosi F., 2017, "*Fuori tutti, c'è un incendio a scuola"... ma è solo una simulazione* virtuale, Rivista antincendio

[15] Cereda E. 2018, *Strumenti di Fire Safety Engineering nella progettazione antincendio: dai modelli di simulazione dell'esodo alla virtual reality*, Politecnico di Torino

[16] Di Claudio A., 2019, *FSE: La gestione delle emergenze nell'edilizia scolastica*, Politecnico di Torino

[17] Viale V. 2019, *La gestione della manutenzione dei presidi antincendio tramite la metodologia BIM*, Politecnico di Torino

[18] Jona S. – Mutani G., *Linee guida per i piani di emergenza ed evacuazione per edifici civili, storico-artistici ed industriali*, Piemonte, Regione Piemonte

[19] Malizia M., *Il Codice di prevenzione incendi*

Sitografia

[1] https://www.consulenteantincendio.it/ [data di accesso: 16/06/22]

[2] https://www.vernicifirewall.it/ [data di accesso: 25/06/22]

[3] https://www.vigilfuoco.it/sitiVVF/ascolipiceno/ [data di accesso: 20/09/22]

[4] https://www.puntosicuro.it/ [data di accesso: 12/11/22]

[5] https://vrforcad.com/ [data di accesso: 28/06/22]

[6] https://docs.unity3d.com/Manual/index.html [data di accesso: 21/08/22]

[7] https://forum.unity.com/ [data di accesso: 28/08/22]

[8] https://assetstore.unity.com/ [data di accesso: 25/07/22]

[9] https://www.mixamo.com/#/ [data di accesso: 27/07/22]

[10] https://discord.com/channels/504429042825429023/50626322104529716 [data di accesso: 26/09/22]

[11] https://www.testo-unico-sicurezza.com/un-gioco-multimediale-sulla-salute-e-sicurezza-sul-lavoro.html [data di accesso: 28/10/22]

[12] https://www.savethechildren.it/ [data di accesso: 15/08/22]

[13] https://www.sicurezza3d.com/ [data di accesso: 20/07/22]

[14] https://www.testo-unico-sicurezza.com/index.html [data di accesso: 30/06/22]

[15] https://www.ictorino2.edu.it/il-nostro-istituto [data di accesso: 5/06/22]

[16] https://www.nfpa.org/ [data di accesso: 20/06/22]

NORMATIVE

D.M. 18/12/1975 – Norme tecniche aggiornate relative all'edilizia scolastica

D.M. 26/08/1992 – Norme di prevenzione incendi per l'edilizia scolastica

D.M. 24/07/1998, n. 331 – Disposizioni concernenti la riorganizzazione della rete scolastica

ISO/TR 16738:2009 "Fire-safety engineering – Technical information on methods for evaluating behaviours and movement of people

D.P.R. 20/03/2009, n. 81 – Norme per la riorganizzazione della rete scolastica

D.P.R. 01/08/2011 n. 151 – "Scuole" con oltre 100 persone presenti sono "attività soggette" a controllo VVF (att. n. 67)

D.M. 3/08/2015 - Approvazione di norme tecniche di prevenzione incendi, ai sensi dell'articolo 15 del decreto legislativo 8 marzo 2006, n. 139

D.M. 7/08/2017 – "RTV Scuole" del Codice P.I. - riguarda att. n. 67 del DPR n. 151/2011 esclusi "asili nido"

Legge 21/9/2018, n. 108 – proroga adeguamento al 31 dicembre 2018 per "scuole" esistenti