UNIVERSITÀ DEGLI STUDI DI MILANO

FACOLTÀ DI SCIENZE E TECNOLOGIE

CORSO DI LAUREA MAGISTRALE IN INFORMATICA

GHOST 3.0: A GHOST STORY WRITER

Relatore:
Ing.
Laura Anna Ripamonti

Candidato:
Alfio Giuliano Faro
Mat. 908958

Correlatore:
Prof.
Francesco Tissoni

ANNO ACCADEMICO 2019/2020

Indice

Capitolo 1

Introduzione

1.1 Breve descrizione del lavoro

Il progetto consiste nella terza iterazione del tool, sviluppato presso il laboratorio PONG, per l'engine Unity 3D, Ghost. L'idea principale del tool è l'implementazione di un supporto al designer, o a colui che si occupa della narrazione durante lo sviluppo di un videogioco, permettendogli di strutturare attraverso degli aiuti visivi, dei "nodi", una possibile sceneggiatura per il proprio gioco.
Sarebbe possibile, altresí utilizzare il tool non solo nell'ambito applicativo per cui é stato pensato e sviluppato, ma anche per altre opere che comprendono una qualsivoglia sceneggiatura.
Nel lavoro svolto, si é dato particolare valore alla struttura multiatto delle produzioni cinematografiche e videoludiche moderne, cercando di uscire, senza peró trascurarla, dalla struttura aristotelica classica a tre atti.

1.2 Ambito applicativo: i videogiochi

1.2.1 Definizione di videogioco

Un videogioco é un software, gestito da un dispositivo elettronico, che consente, per mezzo di una grafica a schermo, piú o meno sofisticata ed articolata, di simulare situazioni di carattere ludico, permettendo a uno o piú utenti di interagire tra loro o con il computer.

1.2.2 Cenni di storia dei videogiochi

Nel 58 abbiamo la nascita di *Tenis for Two*, questo era visualizzato su un oscilloscopio, possiamo considerarlo un precursore dei famoso *PONG* ma il campo da gioco veniva mostrato lateralmente
Il gioco poteva essere giocato da due persone tramite due plance con un tasto per il lancio della palla e una manopola per controllarne la traiettoria.
Nel 1961 vediamo la diffusione di Spacewar®, gioco che simulava una battaglia tra due astronavi, ideato da un ricercatore del MIT.

Nel 1972 abbiamo l'esordio di Pong, primo grande successo della neonata Atari, che diventerá la compagnia di riferimento nel mondo dei videogiochi in quegli anni.
La Atari sará la prima compagnia ad investire nei sistemi di *home gaming*. Nel 77 introduce la prima console ad utilizzare cartucce, una grandissima novità per l'epoca, in quanto il mercato videoludico era composto solamente da console al cui interno poteva esistere solamente un gioco, per la quale la console o il cabinato era stato creato.
L'anno successivo un' altra pietra miliare nel mondo dei videogiochi viene scagliata: Space Invaders. Il successo di questo titolo cominció a spingere sempre piú produttori ad entrare nel mercato dei videogiochi, i quali diedero vita ad una competizione a cui assistiamo ancora oggi con moltissimi studi sparsi per

il mondo.

Solo negli 80 il gaming, peró, raggiunge il grande pubblico. Il protagonista della svolta ha un nome: Nintendo.
Nel 1985 la console nipponica, grazie alla qualitá di titoli come Super Mario Bros, Final Fantasy, The Legend Of Zelda, Metroid e molti altri titoli, sará la dominatrice incontrastata del mercato, soprattutto americano, sebbene non fosse l'unica console presente sul mercato. Alla fine del suo ciclo vitale Il Nintendo Enterteinment System raggiunse quasi le 62 milioni di unitá vendute contribuendo in maniera estremamente significativa alla definitiva diffusione di questo neonato medium.

La decade successiva, gli anni 90, furono caratterizzati da tantissime innovazioni. Passiamo salla grafica in 2D a quella 3D, ed assistiamo anche alla nascita del mercato portatile, sempre con Nintendo, che nel 1989 rilascia sul mercato il Game Boy. Il mercato in questo periodo si sposta definitivamente nelle case degli utenti, che abbandonano sempre piú spesso le sale giochi in favore della comoditá delle loro abitazioni. In questo particolare periodo, un'azienda accetta e scommette su un nuovo tipo di supporto fisico per la propria console, il CD-ROM, scelta rivelatasi poi azzeccata, in quanto permette a Sony, con la sua PlayStation di imporsi in un mercato dove Nintendo era padrona incontrastata da quasi un decennio.
Dagli anni 2000 in poi, i videogiochi accrescono sempre di piú la loro popolaritá arrivando ad essere l'industria che conosciamo ai giorni nostri, con un valore complessivo di circa 160 miliardi di dollari l'anno[1].

1.2.3 *Storytelling* nei videogiochi

Aristotele é considerato il primo vero esperto di narrazione. Egli conosceva l'importanza di dover dare una struttura alla storia per poter raggiungere aiutare la comprensione e poter godere di una storia, uno spettacolo o un poema.

In poetica egli analizzó il metodo per creare spettacoli e poemi, che erano la forma dominante della narrazione nella Grecia antica, enfatizzando il ruolo del protagonista le cui peripezie si svolgono davanti al pubblico.

Nei videogiochi, la storia, la narrativa e l'ambientazione sono importanti tanto quando il gameplay, il game design, la direzione artistica, la programmazione o la musica. Un gioco é un importante strumento per veicolare una storia, al pari di un libro. Un gioco narra, solamente lo fa in una maniera differente da quello che potrebbe essere un mezzo narrativo canonico, come appunto, un libro o un film.

Il videogioco non può essere considerato un media convenzionale, in quanto il grado di coinvolgimento del fruitore è decisamente maggiore rispetto al leggere un romanzo o il vedere un film, poichè l'utente finale interagisce direttamente con il mondo di gioco.

Il videogioco come media narrativo riprende molto dalle strutture dei classici media, ma ne amplia aggiungendo proprio il coinvolgimento del giocatore[35]. Nei videogiochi, infatti, possiamo esplorare liberamente il mondo di gioco, dobbiamo sottostare alle regole che i designer hanno creato per noi, possiamo interagire direttamente con i protagonisti della storia, potendo addirittura modificare le scelte e gli esiti in base alle nostre azioni.

Peró non tutti i giochi sono buoni strumenti di narrazione. Generalmente, la storia, é piú importante nei titoli altamente realistici e rappresentativi,che su quelli piú astratti. La struttura dei videogiochi al suo interno é composta da elementi formali e drammatici, in particolare sono questi ultimi il punto di incontro tra narrazione e videogioco, infatti sono questi elementi che coinvolgono emotivamente e intellettualmente il giocatore. Possiamo riassumere questi elementi in 5 punti, definiti da Fullerton[32]:

- **Character**

  Gli individui che agiscono nella nostra storia, possono essere principali o comprimari.

- **Play**

  Le azioni che lo spettatore vede, ad esempio quando a teatro gli attori si scambiano battute o si muovono sul palco

- **Premise**

  L'idea di fondo che esprime il plot in termini semplici

- **Story**

  La serie temporale di eventi che vanno a definire ciò che effettivamente lo spettatore vede

- **Challenge**

  Gli ostacoli che i personaggi devono affrontare per arrivare al loro obiettivo

1.3 Obiettivo del lavoro

A supporto della scrittura di elementi di narrazione è possibile trovare sul mercato molti strumenti di supporto, ma nulla che intervenga nella fase di definizione della vera e propria struttura narrativa.
Per questo motivo, durante l'anno accademico 2015/2016 nasce GHOST: A Ghost Story Writer da un lavoro di tesi, vederemo nel par. 3.1.1 e 3.1.2 in dettaglio i lavori precedenti a questa iterazione.

L'obiettivo prefissato per quel progetto era quello di fornire uno strumento utile, a chi incaricato di occuparsi della narrazione di un videogioco, per la definizione di uno schema narrativo immediato, senza preoccuparsi della scrittura degli elementi narrativi stessi.
L'obiettivo fu raggiunto attraverso l'implementazione di un grafico aciclico diretto in grado di descrivere una struttura in 3 atti.
Nella seconda iterazione del plugin, la struttura fu ampliata per poter dareun supporto completo alla descrizione dei personaggi, in particolare definendo un arco di trasformazione degli stessi nel corso degli eventi, aggiungendo anche le descrizioni delle possibili relazioni tra i personaggi.
Il principale obiettivo del mio lavoro, la terza edizione del plugin, é l'ampliamento della struttura narrativa, non focalizzandosi esclusivamente sulla struttura aristotelica classica a 3 atti, ma aggiungendo la possibilitá di una struttura non lineare a piú atti e piú archi narrativi.
Inoltre si é data particolare enfasi e attenzione alla "lore" del gioco, andando a definire un nodo del grafo appositamente per questo motivo, e alle storie secondarie, sub narrazioni che si distaccano spesso dagli eventi principali del gioco, ma che vanno a definire piacevoli intermezzi per il giocatore finale, le quali molto spesso, non vengono particolarmente caratterizzate, ponendole come accessori del gioco stesso e non come mezzo molto potente per veicolare delle informazioni aggiuntive.

Capitolo 2

Storytelling

2.1 Narrativa

Lo storytelling é l'atto di narrare, disciplina che usa i principi della retorica e della narratologia.
É una metodologia che usa la narrazione come mezzo creato dalla mente per inquadrare gli eventi della realtá e spiegarli secondo una logica, cercando di dargli un senso compiuto.

Lo storytelling é definibile come una e vera propria arte, che implica ricerca pianificazione e competenze per poter ritrarre eventi reali o fittizi attraverso parole, immagini, suoni.
É uno strumento attraverso il quale puó avvenire una forma di comunicazione efficace: coinvolge contenuti, emozioni, intenzionalitá e i contesti. La storia raccontata ha una connotazione emotiva poichè cerca di coinvolge le persone cercando di andare a toccare la loro emotività.
Lo storytelling si sviluppa a partire dall'assunzione di due principi fondamentali: l'organizzazione delle esperienze umane che avviene grazie ai racconti e alla narrazione. É un processo che dota le persone di una sensibilitá culturale che li mette in grado di attivare processi riflessivi e formativi, soprattutto nei gruppi.

Lo storytelling puó, teoricamente, essere fatto su qualsiasi cosa abbia la necessitá di essere sostenuta dal punto di vista comunicativo: un'azienda, un brand, un prodotto, una persona o un evento.

Ci sono 5 regole di base che devono essere seguite per realizzare una buona narrazione:

- Chiarire fin da subito chi é il soggetto della storia.
- L'utilizzo di una trama decisa a priori é fondamentale
- La contestualizzazione temporale degli eventi narrati é necessaria per facilitarne la comprensione all'audience di riferimento
- É fondamentale specificare i luoghi in cui si sviluppa la storia, in modo da aiutare il target a comprenderla al meglio.
- É importante esplicitare al meglio quali sono i fattori che motivano il succedersi degli eventi della toria narrata.

Abbiamo anche diverse tecniche con cui si puó ottenere un efficace narrazione. Le piú diffuse sono:

- **In media Res**: tecnica che consiste nel cominciare la storia raccontando il suo momento piú avvincente per por poi fare un salto nel passato, utilizzando flashback, per spiegare l'origini e le cause di quanto accade.
- **Il viaggio dell'eroe**: questa modalitá narrativa, probabilmente la piú utilizzata in campo videoludico, consiste nell'incentrare il racconto sulle avventure del protagonista: dal tema del superamento degli ostacoli, a quello della perdita.
- **Show don't tell**: questa tecnica prevede una descrizione ricca di dettagli di ció che sta accadendo raccontando scena per scena. Sono i dettagli della scena a raccontare la storia.

- **Cliffhanger**: La narrazione cliffhanger é caratterizzata da una fine brusca con picchi di suspense che creano nell'immaginario del lettore il bisogno di conoscere l'epilogo, spingendolo cosí a cercare di scoprire cosa accadrá dopo.

- **Racconto Ciclico**: questa tecninca fa sí che la storia inizi e finisca nello stesso modo; la narrazione parte con la descrizione di un accadimento e si conclude parlando di nuovo dello stesso

2.2 Narrativa nei vari generi

Una narrativa avvincente é cruciale al gameplay, specialmente per alcuni particolari generi di giochi.
In franchise di grandissimo successo come Tomb Raider(Crystal Dinamics, 2013)[2], Uncharted (Naughty Dog 2007)[3], God of War(Santa Monica Studio)[4], The Last of us(Naughty Dog, 2013)[5] il giocatore controlla il proprio personaggio all'interno di mondi pericolosi, alla ricerca di tesori, risolvendo enigmi e combattendo i nemici.
Il giocatore conosce perfettamente la situazione in cui il proprio avatar si muove, e ció lo spinge a prendere decisioni migliori in fase di gameplay.
In particolare Naughty Dog, con la serie *Uncharted* ha segnato un vero e proprio anello di congiunzione tra cinematografia e videogioco, con un approccio appartente piú all'industria holliwoodiana che a quella videoludica. Nathan Drake, il nostro protagonista, infatti si muove in mabienti ostili usando degli scenici angoli di camera e creando continuitá tra gameplay e scene di intermezzo.
In uno shooter, giochi in cui l'azione predominante è quella di sparare con armi dalla distanza, possono essere da fuoco o no, ai vari nemici che popolano il livello, é importante che il giocatore conosca il perché sia coinvolto nel conflitto, inoltre la narrativa aiuta a giustificare la ricca gamma di tipi di nemici, armi, e obiettivi del giocatore
Sicuramente il genere piú dipendente da una buona narrazione é il gioco di

ruolo. Questo discende direttamente dai giochi di ruolo paper and pen come *Dungeons & Dragons* e dalle avventure testuali come *Zorc*. Il genere é caratterizzato da un enorme mondo di gioco, e bisogna riuscire a catturare il giocatore, sia con una storia ben esposta e raccontata, sia con una "lore" del mondo di gioco coinvolgente, che permetta al giocatore di immergersi perfettamente in un mondo immaginario.

Questi sono alcuni dei generi dove la narrativa ricopre un ruolo molto importante, se non cruciale. Ma non a tutti i giochi hanno bisogno di una componente narrativa. Nelle simulazioni sportive, ad esempio, il coinvolgimento di un giocatore non é dettato da una storia avvincente, ma nel permettere all'utente di giocare con le loro squadre preferite.

Attualmente, peró, anche questo in questo genererevediamo la comparsa di un elemento di narrazione, ad esempio possiamo vedere in *Fifa* o *Nba 2k21* nuove modalitá in cui il giocatore deve impersonare un giovane atleta agli albori della propria carriera, cercando di farsi notare dall'allenatore, scegliere i propri collaboratori o trattare il proprio contratto.

Un altro genere in cui la narrativa non é fondamentale é quello del Racing game, dove il giocatore impersona un pilota a bordo del suo bolide, o lo Strategico dove il giocatore conosce veramente poco sul suo ruole di "comandante". Anche in questi casi, peró, l'evoluzione del gioco ha portato all'aggiungere una forte componente narrativa anche in generi in cui non é necessaria. L'esempio precedentemente citato di *Fifa* non é l'unico, abbiamo anche dei puzzle game con una forte componente data dalla storia, l'esempio piú conosciuto e popolare é sicuramente *Portal*, dove il giocatore viene sfidato da una intelligenza artificiale chiama GLaDOS e deve completare i puzzle usando una particolare pistola.

Un altro grande esempio di evoluzione della narrativa in generi non molto famosti per questa caratteristica, la serie di *Starcraft*[6] o di *Command & Conquer*[7] sono strategici in tempo reale con storie profonde e personaggi vividi. In particolare *Starcraft* é sviluppata da Blizzard , una compagnia che ha un grande focus nel creare dei mondi ampi, personaggi complessi in cui il giocatore

possa immedesimarsi, uno dei piú famosi e articolati si lega ad un altro genere in cui non é necessaria una grande storia per avere un gameplay divertente, il gioco di ruolo online. Blizzard ha speso un grande quantitativo di tempo e risorse per arricchire la "lore" del mondo di Azeroth. Ogni missione é scritta per essere ben miscelata nel mondo di gioco ed ogni singola avventura scrive un pezzo di storia, storia che non viene neppure notata dal giocatore in quanto non narrata o non palesata.

2.3 Elementi della narrativa

Dall'inizio del linguaggio, l'atto di narrare stoire é servito per insegnare, unire eccitare e calmare. L'essere umano ha presto imparato ad abbellire le sue storie, aggiungendo elemeti per attirare l'attenzione dell'ascoltatore. Ma le storie non sono altro che composizioni di frasi, la cui struttura puó essere espressa come:

$$Soggetto + Verbo + Oggetto = Frase[8]$$

e la struttura base della storia, come detto precedentemente in 1.2.3, puó essere espressa come:

$$Protagonista + Scopo + Conflitto + Ostacoli + Risoluzione = Storia[8]$$

Con questa semplice formula si potrebbe definire qualunque storia mai narrata. Andiamo adesso ad analizzare in maniera specifica gli elementi che compongono la storia.

2.3.1 Protagonista

Nello storytelling classico, il protagonista é l'eroe della storia, il personaggio che si imbarca nell'avventura.
L'eroe è il personaggio che deve più empatizzare con il pubblico, cercando di canalizzarne l'attenzione, . Il protagonista nei videogiochi, essendo un media interattivo, di solito é il giocatore stesso. Il protagonista é il giocatore e viceversa.
Romanzi, film, serie tv e altri media, hanno spesso piú protagonisti, ad esempio, nel famoso show degli anni '90 *Friends* non possiamo identificare un singolo protagonista, ma ogni personaggio é l'attore principale della propria storia, dipendente dall'arco narrativo dell'episodio o dell'intera stagione.
I giochi solitamente hanno un solo protagonista, poiché potrebbe comportare delle difficoltá per il giocatore tenere traccia di tutte le vicende di avatar multipli.

2.3.2 Obiettivo

Tutte le storie parlano di un desiderio, anche se questo é semplicemente il desiderio di sopravvivenza o di essere lasciati da soli. Se manca questo fondamentale tassello la narrazione perde di interesse e significato.Riusciamo a immedesimarci in queste storie, che esse siano di pura finzione o meno, poiché riguardano problemi che affrontiamo anche noi, anche se a diverse scale di grandezza.
Alcuni protagonisti vogliaono cambiare la loro vita, alcuni vogliono affrontare i problemi per trovare un poco di serenitá, ma alla base di tutto, come detto é presente un desiderio, che é anche la chiave della storia che stiamo andando a raccontare.
Senza un chiaro obiettivo, non abbiamo una direzione, ne per il giocatore ne per il protagonista. Storie avvincenti, mettono a confronto due fazione l'una contro l'altra dove solamente una puó prevalere sull'altra. Quindi per ogni protagonista dobbiamo avere un' antagonista.

2.3.3 Conflitto

Il conflitto é ció che rallenta il protagonista durante la sua avventura. Senza il conflitto non avremmo interesse o coinvolgimento nella storia.
Esso puó essere creato dall'ambiente, dall'antagonista, potrebbe anche essere un conflitto interiore che lo stesso protagonista sente, come una colpa o dei dubbi.
Il piú classico dei conflitti é sicuramente con l'antagonista della storia, che viene rappresentato come un malvagio, un malfattore, come una figura negativa, che il nostro eroe é portato ad affrontare nel corso degli eventi. Un'altra importante caratteristica di un antagonista é che esso pensa di avere ragione, e che siano gli altri, protagonista compreso, a trovarsi nel torto.

2.3.4 Ostacoli

L'eroe della storia deve affrontare sfide ed ostacoli, ma a differenza del giocatore, l'eroe affronta gli ostacoli perché deve e non per divertimento.
Per una storia che riesca a intrattenere, l'eroe deve affrontare moltissimi ostacoli durante la narrativa.
Il protagonista in uno show TV, un romanzo o un film dovrebbe reagire quasi con disperazione.
Queste situazioni cosí intense scatenano una tensione drammatica nell'audience, che porta a tifare per il successo dell'eroe. In ogni caso si vede come gli ostacoli cercano di fare deviare il percorso, spaventano l'eroe e, nei videogiochi, é il giocatore a dover superare quegli ostacoli, i quali generalmente sono nemici o puzzles che creano anche il contenuto di gameplay.

2.3.5 Risoluzione

Delle buone storie hanno anche bisogno di un finale soddisfacente. L'investimento emotivo che lo spettatore, il giocatore, il fruitore in generale ha messo nell'odissea del protagonista deve essere risolta in qualche modo, che non deve per forza essere felice, ma sicuramente soddisfacente e chiudere le varie linee narrative descritte durante la narrazione, cercando di non lasciare buchi nella trama per non arrecare un senso di incompiutezza nello spettatore.

2.4 Struttura ad atti

Chi si occupa di storytelling, in qualsiasi ambito, deve conoscere, ovviamente, la teoria che sta dietro ad esso. Questo é possibile grazie alla **narratologia** termine coniato nel 1969 dal filosofo Tzvetan Todorov per indicare la branca di studi delle strutture narrative.
Non bisogna, peró, confondere la struttura con la trama, la quale cambia in base al progetto che si sta andando a creare, bensí é uno strumento utile e versatile, che serve da modello narrativo e che puó essere applicato in molteplici narrazioni.
La prima struttura ad atti fu formalizzata da Aristotele nel 300 A.C [31], e ad oggi, quella struttura é ancora tra le piú diffuse e utilizzate nell'ambito delle scienze narratologiche e sugli studi di sceneggiatura drammatica.
La struttura a 3 atti é stata superata da altri tipi di strutture, che hanno usato quest'ultima come ossatura, per poi espandere determinati aspetti rispetto ad altri.
Vediamo in dettaglio quali tipi di strutture possiamo ritrovare, le quali sono state utilizzate per la creazione del mio progetto finale.

2.4.1 Struttura a 3 atti

Le tre fasi di questa struttura narrativa coincidono con le tre fasi della dialettica aristotelica: Tesi, Antitesi e Sintesi. Per fare il parallelismo con la dialettica la Tesi rappresenta il primo atto, ovvero l'inizio, l'antitesi é lo svolgimento, quindi la parte centrale della narrazione, ed infine la conclusione che rappresenta la sintesi.
Vediamo nel dettaglio i 3 atti.

- **Atto 1: Inizio**

  La prima parte della struttura di ogni narrazione serve principalmente per mettere in scena tutta la struttura temporale e a presentare i personaggi che vedremo nella storia. Serve anche a presentare il luogo in cui ci troviamo e il tempo in cui la storia é ambientata. In questo atto troviamo l'incipit della storia, il background dei personaggi o della scenografia in cui ci troviamo.

  Alla fine di quest'atto troviamo Un evento scatenante che ci porta negli eventi del secondo atto, che comprende la gran parte della storia.

- **Atto 2: Svolgimento**

  Il nostro protagonista comincia a compiere il suo viaggio per raggiungere il proprio obiettivo e subito deve affrontare i primi ostacoli, i quali devono essere numerosi e fastidiosi o non avremmo il lato drammatico.

  Il secondo atto é la metá della storia e mostra la maggior parte della drammaticitá dell'azione : il confronto. Molto spesso la fine del secondo atto é il punto piú "basso" nella storia dei nostri eroi, quando il protagonista si rende conto che l'antagonista che sta affrontando é troppo forte persino con tutti gli sforzi che sta facendo, dove tutto ormai sembra perduto, in pratica la peggior situazione in cui il nostro protagonista si possa trovare.

- **Atto 3: Conclusione**

  In quest'ultimo atto il protagonista si trova faccia a faccia con l'antagonista, e finalmente abbiamo la conclusione della linea narrativa e della storia a cui stiamo assistendo. Una volta finita ques'ultima fatica, il nostro eroe potrá tornare alla normalitá, portandosi dietro il bagaglio di esperienza acquisito, che potrebbe essere l'incipit per un nuovo episodio delle sue avventure o la fine di tutto. Questo struttura a 3 atti molto lineare viene spesso utilizzata nelle grandi produzione videoludiche, che cercano di emulare i tre atti dei blockbuster Hollywoodiani. molto spesso anche nella stesura dei giochi, la tecnica utilizzata é proprio quella della scrittura delle scenggiature dei film, con un discors, una sinossi, un prova della scena e quindi lo sceneggiato.

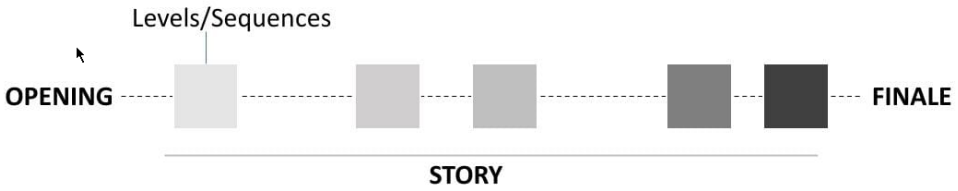

Figura 2.1: Struttura lineare a 3 atti[8]

2.4.2 Struttura a 4 atti

Nonostante la struttura a 3 atti sia di fatto la piú conosciuta, la piú studiate ed emulata, la struttura a 4 atti rappresenta quella piú utilizzata soprattutto in ambito cinematografico.
Molto spesso gli sceneggiatori parlano di **"Midpoint"** una via di mezzo nel secondo atto, che potrebbe essere un colpo di scena, ad esempio un alleato che diventa un nemico o viceversa.
Una buona sceneggiatura con questa struttura spesso é molto ben bilanciata, anche a livello di numero di pagine e di minutaggio le sceneggiature hanno regole molto stringenti.
In particolare nel caso della struttura a 4 atti, il primo e il quarto atto (inizio e conclusione) hanno uguale lunghezza e il secondo ed il terzo atto hanno lunghezza doppia del primo e dell'ultimo.
Il "Midpoint" é definibile come un intervallo, qualcosa di importante che succede al plot o ai personaggi, se non addirittura ad entrambi. Un esempio di questo particolare artefatto narrativo nei videogiochi, lo possiamo riscontrare in *BioShock*, quando il protagonista Jack impara cosa é successo nella cittá subacquea di Rapture, dove la storia si svolge e scopre di essere stato manipolato dall'unico amico che pensava di avere.

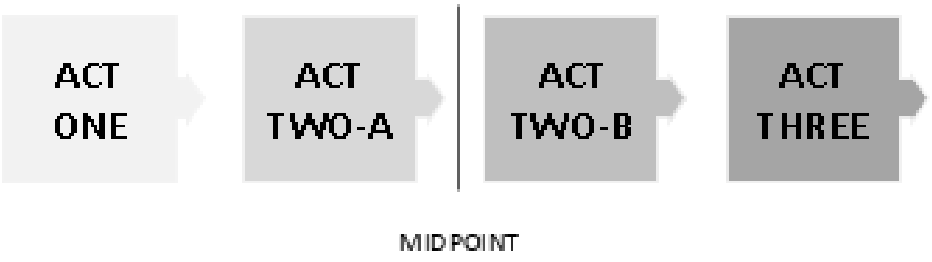

Figura 2.2: Struttura a 4 atti[8]

2.4.3 Struttura a 5 atti

Shakespeare era solito dividere i suoi spettacoli in 5 atti, e questa struttura viene utilizzata molto spesso in ambito televisivo, soprattutto Serie tv come *Lost*. Gli atti si suddividono in:

- **Introduzione e presentazione di un conflitto pre esistente**
  L'idea in questo atto é quella di presentare al fruitore, nel nostro caso il giocatore, dell'opera il contesto in cui vede, o deve, svolgere il suo compito, cercando di spiegare il tutto in maniera concisa e immediata.

- **Una svolta che peggiora il conflitto**
  In questa parte ci vengono mostrati gli attori nella nostra storia, antagonisti e non, e viene fatta un' ulteriore digressione su ció che ci si para di fronte.

- **Colpo di scena**
  Il terzo atto in una struttura che ne comprende 5, é molto simile a quanto visto nel "Midpoint" della struttura 4 atti. In questo momento i piani del nostro eroe vengono smantellati e devono essere ripensati.

- **La spirale degli eventi**
  Nel quarto atto l'azione cresce in modo drastico e repentino, e tutto ció che succede adesso é un preparativo per la conclusione della storia ormai imminente.

- **Climax e risoluzione**
  Il punto in cui tutte le linee si chiudono e dove la storia ha un crescita fino al suo punto piú alto, per poi scendere con la risoluzione di tutti i punti lasciati in sospeso.

2.4.4 Struttura a 8 atti

L' approccio in sequenza, anche conosciuta come struttura a 8 atti, viene spesso utilizzata nella struttura di una storia.
Come ampiamente esposto, tutte le strutture non sono altro che emulazioni della struttura a 3 atti, in particolare questa versione, la quale non fa altro che prendere i classici 3 atti e dividerli in 8 sotto sequenza, ognuna di loro con una propria risoluzione che permettono di arrivare alla risoluzione finale della storia.
Ogni sequenza ha la propria evoluzione drammatica, il proprio contesto, come se fosse a sua volta una piccola struttura a 3 atti, con inizio, sviluppo e fine.
Molti film d'azione, ma soprattutto molti videogames, utilizzano questo paradigma, un sempio potrebbe essere la storia che coinvolge John McClane un poliziotto di New York che si trova a Los Angeles per passare le vacanze natalizie con la famiglia, nel film cult del 1988 *Die Hard:Trappola di Cristallo*.
Ogni volta che il nostro eroe , in questa pellicola, si trova a dover affrontare i membri dell'organizzazione , i quali tengono in ostaggio i dipendenti della compagnia che si trova all'interno del grattacielo dove è ambientata la storia, è uno sviluppo a se, e gli epiloghi di queste piccole vicende portano al epilogo finale con la classica vittoria del bene sul male.
Il parallelismo con il mondo dei videogiochi è estremamente semplice in questo caso. Infatti ogni volta che il nostro protagonista affronta un nuovo livello, con relativo epilogo, è un mattone che va a comporre tutta l'esperienza videoludica e ci porta al livello finale e quindi a chiudere l'arco narrativo.
Particolare importanza in questo tipo di struttura riveste il filo logico della storia, non può succendere B se prima non è successo A, ovviamente.

Figura 2.3: Struttura sequenziale a 8 atti[8]

2.4.5 Narrativa ramificata

Nel mondo dei videogiochi, sempre più spesso , vediamo titoli che non hanno un proseguo della storia lineare.
Se prendiamo ad esempio i capolavori di *Quantyc Dream* e del loro Game Designer *David Cage*, questi giochi hanno basato sulla narrativa e sulla sceneggiatura il loro successo. La peculiarità di questi giochi è che a partire da un unico problema, un unico livello, la storia in base alle scelte che compiamo, le risposte che diamo ad un dialogo o l'interazione con qualche oggetto, cambia, dando vita ad un'esperienza unica con moltissimi possibili finali.

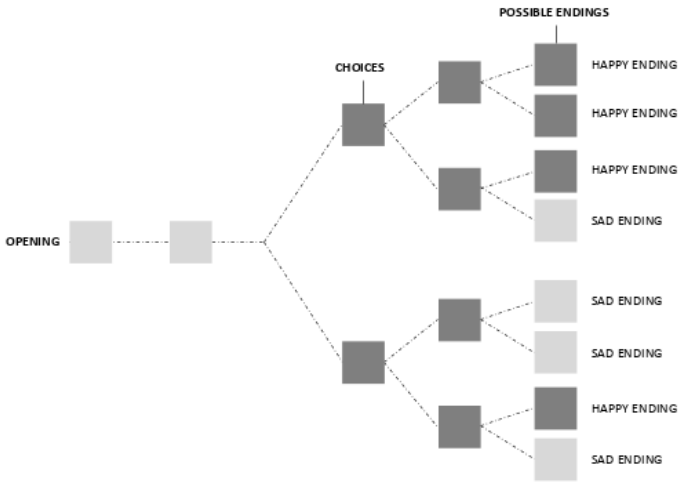

Figura 2.4: Struttura ramificata[8]

Questo tipo di struttura, però, crea molti interrogativi sia nel giocatore che nello sviluppatore. Mettiamo, ad esempio, che il giocatore affronti moltissime ore di gioco per arrivare ad un determinato finale, gli sviluppatori a questo punto, quando vanno a implementare questa caratteristica dei finali mulitpli, tendono a creare finali simili tra di loro, cambiando alcuni dettagli, per non scontentare il giocatore, il quale dovrebbe ricompletare l'avventura per poter assistere ad un altro epilogo.

Questa caratteristica da al videogioco una grande rigiocabilità, in quanto i giocatori più attenti al completamento totale di un gioco, tenderanno a rivivere l'avventura più e più volte, fino a quando non avranno scoperto ogni piccola cosa del gioco.

Capitolo 3

Metodologia

La metodologia usata per questo progetto è basata sul modello a cascata classico. Questo è il più classico modello di cliclo di vita del software, strutturato in sequenze lineari di passi.

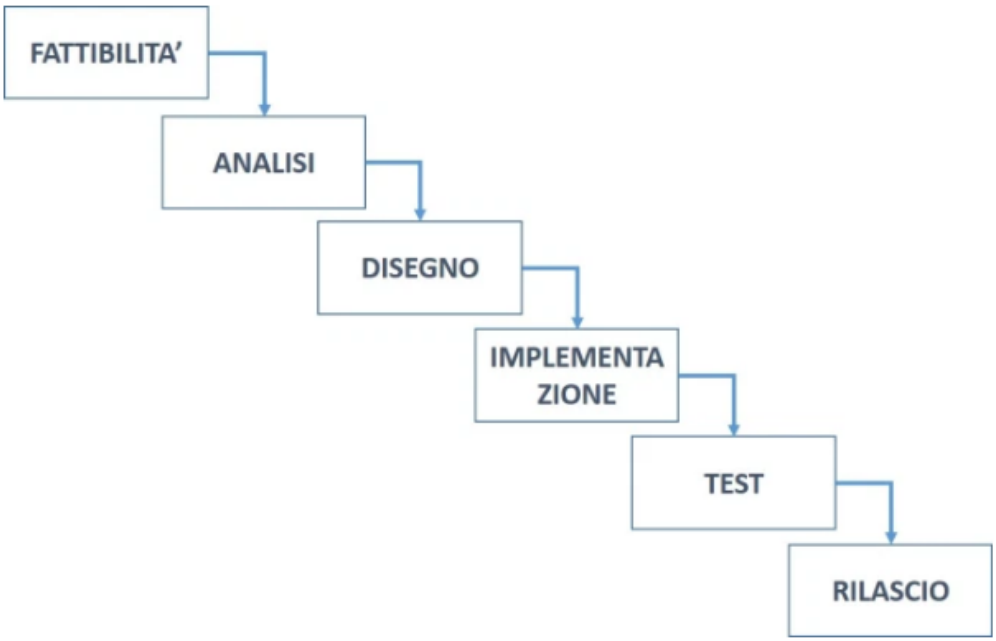

Figura 3.1: Modello a cascata[9]

I passi evidenziati anche in figura 3.1 sono:

- **Analisi dei requisiti**

  L'attività preliminare allo sviluppo, atta a definire i requisiti funzionali che il nuovo software deve avere.

- **Disegno**

  La definizione formale di come i requisiti analizzati e trovati, prenderanno forma e verranno soddisfatti.

- **Implementazione**

  La parte effettiva di sviluppo del nuovo software.

- **Test**

  L'esecuzione dei test, prima unitari fatti da me, ed infine una sessione di testing con utenti esterni.

- **Rilascio**

  Il rilascio della funzionalità una volta finita la fase di test e gli sviluppi.

Essendo un approccio superato, ho compiuto questa iterazione più volte, in modo incrementale per ogni *feature* sviluppata.
Ci sono stati dei ricicli nelle fasi di

- Analisi

  Per ogni nodo da creare, l'analisi era ovviamente divesa, in quanto bisognava definire i tipi di connesioni e quante queste potessero essere in relazione con gli altri nodi presenti.

- Disegno

  Per ogni nodo è stata data una definizione formale dei valori in *input* e dei valori in *output* e quali dovessero essere le sue finalità.

- Implementazione

  L'implentazione effettiva del nodo in questione

- Test

  La fase di test per ogni nuovo nodo creato, per vederne le connessioni ed il comportamento generale nella *canvas*.

La fase di test conclusiva, invece, è stata eseguita su due diversi tipi di utenti, il primo con esperienze pregresse in ambito di sviluppo, e il secondo tipo senza alcuna esperienza in ambito di sviluppo.
L'obiettivo era quello di raccogliere quanti più *feedback* riguardanti l'usabilità e la fruibilità del software da parte di chi aveva già avuto esperienza in questo campo, con parallelismi anche rispetto ad altri software presenti sul mercato, mostrati in 3.2, e i secondi per verificare se possa essere un effettivo aiuto per i neofiti che vogliono cimentarsi in questo ambito.

3.1 Planning e Analisi dei requisiti

Nella fase iniziale vi è stato uno studio approfondito e mirato per capire quali fossero le modifiche da fare per andare a migliorare ed ampliare il progetto.
La prima cosa che è stata fatta, dunque, è stata l'analisi dei precedenti lavori sviluppati.

3.1.1 Ghost

Ghost[29] è un *plugin* di Unity3D nato come lavoro di tesi durante l'anno accademico 2015/2016 e poi modificato e ampliato, sempre come lavoro di tesi nel 2016/2017 .
Sviluppato sulla base del framework per Unity3D *Node Editor*, il suo intento è quello di aiutare chi ha il compito nel progetto, di sviluppare una narrativa per qualsiasi genere di videogioco, dandogli una struttura immediata su cui poi basare l'effettiva storia.
Ghost è basato su grafi diretti aciclici, i cui nodi determinano una parte di struttura narrativa che si vuole descrivere.

Con questa prima versione del plugin si permetteva al fruitore di:

- Creare la struttura narrativa in 3 atti
- Creare personaggi combinando 4 nodi che ne descrivevano le peculiarità
- Creare sidequest
- Utilizzare le carte prese dal gioco da tavolo *Once Upon a Time* per utilizzarle in maniera descrittiva nei vari nodi.
- Esportazione in Html

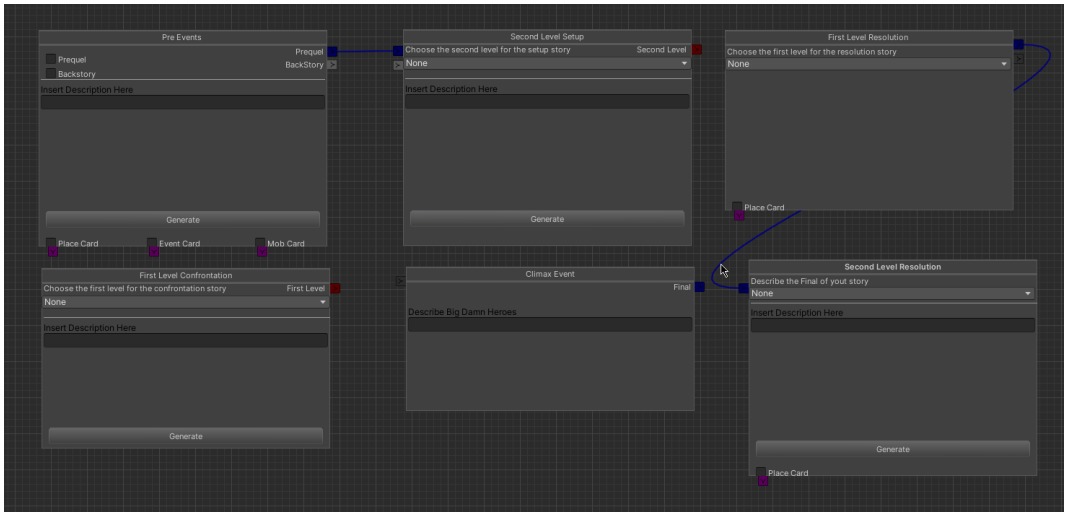

Figura 3.2: Nodi per la definizione della storia nella prima versione di Ghost

Tra i vari nodi vengono definiti dei collegamenti colorati, che permettono di creare una connessione logica all'interno del grafo.
Ogni nodo non rappresenta solamente ciò che il nome stesso descrive, infatti molti nodi possiedono dei selettori di scelta o dei pop-up che crea un alto grado di personalizzazione del progetto.

Oltre la struttura narrativa, come detto in precedenza, Ghost permette anche la personalizzazione dei personaggi e delle loro relazioni. Questa prima versione definisce i nodi:

- ***CharacterArchetype***

  L'archetipo del personaggio descritto (e.s "Eroe", "Aiutante", "Comprimario").

- ***CharacterRole***

  Descrizione del ruolo del personaggio nella storia.

- ***CharacterModifier***

  Un modificatore riguardante una caratteristica del personaggio.

- ***CharacterCharacteristic***

  Indica le catteristiche del personaggio.

- ***CharacterResult***

  Nodo che riassume tutte le caratteristiche del personaggio appena creato.

Non era ancora presente. però, un nodo descrittivo del *background* del personaggio e delle relazioni che possono crearsi tra di loro.
Per ottenere questo risultato è stato utilizzato Node Editor Framework che vedremo in 4.2.

3.1.2 Ghost 2.0

La seconda versione del plugin è del 2017, sempre sviluppata come lavoro di tesi.
Sono state apportate diverse migliorie e la creazione di nuovi nodi per descrivere nel modo migliore possibile la struttura narrativa.
Le modifiche più significative sono state:

- A differenza della prima versione, dove Node Editor permetteva solamente la definizione di un *canvas* generico, nella seconda vediamo uno specifico *canvas* con regole studiate proprio per la versione 2.0.
  Per *canvas* intendiamo l'area di lavoro all'interno del quale mostrare ed inserire l'informazione che vogliamo. come vedremo sempre in 4.2.
  Nello specifico, per quanto riguarda il lavoro precedente sono state utilizzate le proprietà nome e implementata una nuova logica di attraversamento del grado e sovrascrittura del metodo di controllo di conformità del nodo(per permettere solamente ai nodi di una determinata lista di poter essere inseriti nella *canvas*)

- Da una generica logica di attraversamento del grafo, ne è stata creata una nuova appositamente studiata proprio per questa iterazione.
  Attraverso l'estensione della classe *NodeCanvasTraversal* di Node Editor, è stato possibile dettagliare la logica di attraversamento, permettendo alle *callback* successive di calcolare dinamicamente lo stato del nodo.

- *Refactoring* della struttura in 3 atti precedente per supportare le nuove funzionalità.

- Rimane invariata la feature di esportazione in HTML e salvataggio del caricamento per il formato di Node Editor.

- Viene disaccoppiato il core di Node Editor, permettendone l'aggiornamento senza bisogno di un *hard refactoring* di Ghost.

La modifica più grande, però, è stata fatta nell'ambito della descrizione in termini di tratti, archetipi ruolo e trasformazione del personaggio[36].
Abbiamo l'introduzione del nodo *CharacterDescriptor*, mostrato in figura 3.3, che definisce la descrizione generale di un personaggio con la possibilità di inserirne le informazioni personali.

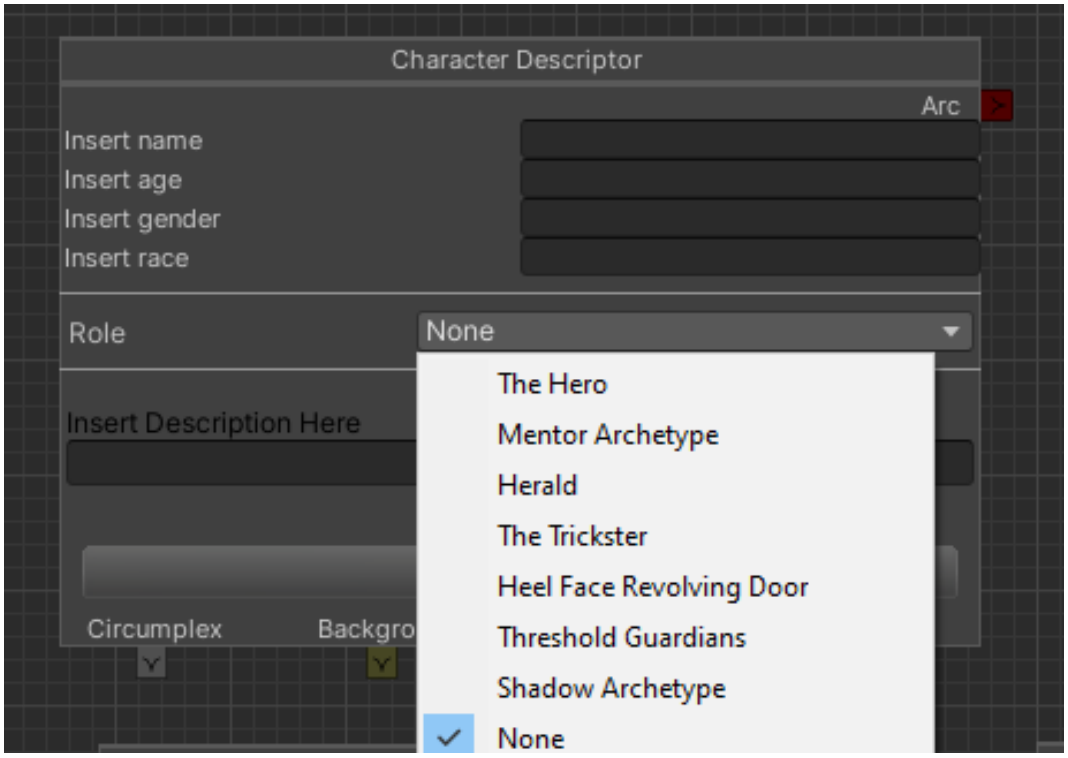

Figura 3.3: Nodo character descriptor

Oltre la normale descrizione anagrafica possiamo anche inserire un suo ruolo nella storia.
In più possiamo descrivere un vero arco di trasformazione nel tempo del personaggio, dove ogni nodo rappresenta un momento particolare nella storia che il nostro personaggio sta vivendo.

Per definire questo arco narrativo abbiamo l'introduzione dei seguenti nodi:

- **Transformational Arc Start**

  L'inizio dell'arco narrativo e di trasformazione

- **First Turning Point**

  Il primo punto di svolta nella storia del personaggio

- **Mid point**

  Il maggior punto di tensione nell'arco narrativo del personaggio

- **Second Turning Point**

  Il punto precedente alla risoluzione della storia, dove il personaggio deve avere un'ulteriore svolta per poter permettere la risoluzione della sfida e dar vita al finale del suo arco narrativo.

- **Resolution**

  Nodo finale che mostra se il nostro personaggio è riuscito a superare le sfide che ha dovuto affrontare durante la sua trasformazione.

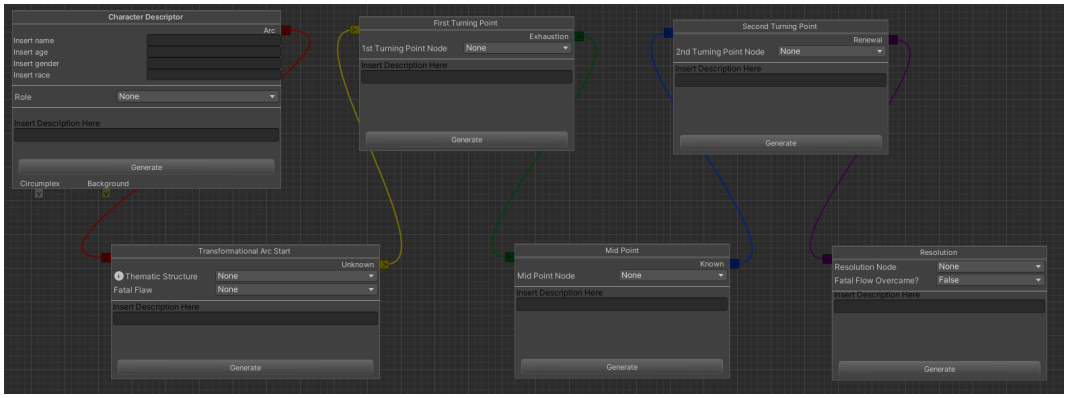

Figura 3.4: Arco di trasformazione del personaggio completo

Abbiamo poi i nodi *Background*, il quale permette di inserire un testo sul background del personaggio e il nodo *InterpersonalCircumplex*, che crea un grafico dove poter mostrare una rappresentazione del rapporto tra due o più personaggi, completano le nuove aggiunte al lavoro.

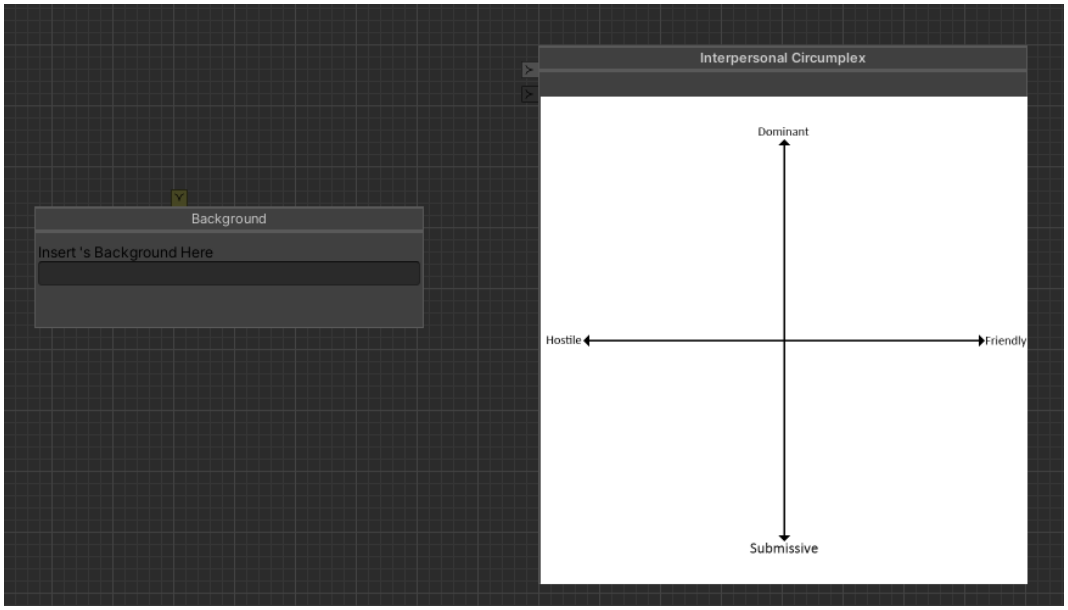

Figura 3.5: Nodi *Background* e *InterpersonalCircumplex*

Il nodo *InterpersonalCircumplex*, in particolare, una suddivisione in due parti:

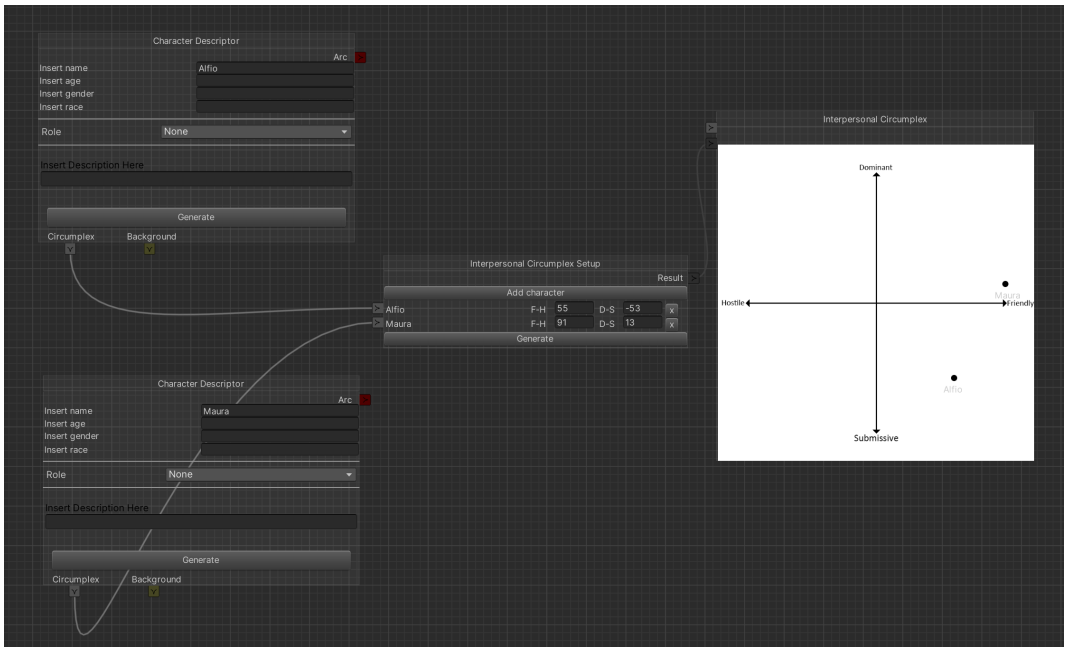

Figura 3.6: Funzionamento nodo

Come vediamo in figura 3.6, abbiamo due nodi che descrivono due personaggi, i quali sono connessi al nodo *Interpersonal Circumplex Setup*, che definisce le coordinate sugli assi dei personaggi connessi, mostrando poi i nomi nel nodo *InterpersonalCircumplex*.

Ghost	Ghost 2.0	Ghost 3.0
Canvas Generico definito dal *framework*	Canvas specifico per Ghost 2.0	Canvas specifico rifattorizzato per supportare l'ultima versione del *engine*
Generica logica di attraversamento definita dal framework	Logica di attraversamento specifica per i nodi di Ghost 2.0	Ampliamento ed aggiunta di nuovi tipi di nodo per l'attraversamento.
Struttura in 3 atti	Rifattorizzazione della struttura per il supporto nel engine	Struttura in 3 atti completamente ricreata, superamento dei 3 atti aggiungendo strutture moderne a 4,5 e 8 atti.
Nessuna definizione di lore	Nessuna definizione di lore	Definizione di un nodo interamente dedicato alla lore del mondo di gioco
Nessun nodo di *utility*	Nessun nodo di *utility*	Introduzione di due nuovi nodi di *utility*, per aiutare l'utente ad inserire più di un personaggio all'interno dei suoi atti e a creare una struttura ramificata della storia
Descrizione del personaggio con ruolo e archetipo	Descrizione ampliata con aggiunta di tratti, ruoli, trasformazione e relazione con i personaggi	Mantenimento della struttura del personaggio, con rifattorizzazione per il supporto della tecnologia corrente.

Esportazione in HTML e salvataggio nodi	Caratteristica invariata con rifatttorizzazione per l'aggiunta di nuovi formati	Caratteristica invariata con rifattorizzazione per il supporto alla tecnologia corrente
Nessun nodo per *sidequest*	Aggiunta nodo per sidequest	Aggiunta label in nodo sidequest per la connessione con il nodo plot.

Tabella 3.1: Differenze tra le tre versioni di Ghost

I nodi descrittivi sui personaggi sono rimasti invariati, come mostra la tabella 3.1, e vengono attualmente utilizzati per descriverli.
La maggioranza dei cambiamenti è basato su un supporto massiccio alle differenti e moderne strutture narrative, con l'introduzione delle moderne strutture a più atti, un' attenzione alla costruzione della storia del mondo di gioco e una particolare enfasi nella costruzione delle *sidequest*.

3.2 Stato dell'arte

Il passo successivo allo studio dei lavori precedenti è stato, logicamente, quello di andare ad analizzare lo stato dell'arte, andando ad individuare i principali strumenti che vengono utilizzati in ambito letterario e videoludico.
I più noti e famosi sono *Twine* e *XMind*, entrambi fruibili in modo gratuito.
Gli strumenti che sono stati esaminati per la mia analisi sono:

- ***Twine***[10]
- ***XMind***[11]
- ***Fungus***[12]
- ***Scrivener***[13]
- ***Final Draft***[14]
- ***Inform***[15]

3.2.1 Twine

Twine è uno strumento *open-source* gratuito per la narrazione interattiva, non lineare di storie.
Viene in aiuto delle persone che non hanno esperienza nello scrivere codice, ma possono essere usati i paradigmi logici di quest' ultima come variabili, condizioni logiche, importazioni di immagini etc.
Twine esporta direttamente in HTML, rendendo molto flessibile il lavoro, in quanto basta semplicemente una connesione ad internet per poter operare da qualsiasi parte, in qualsiasi momento.
Twine è stato creato da *Chris Klimas* nel 2009 e viene mantenuto attualmente da volontari che lavorano su differenti repositories. Twine, quindi, è uno strumento per la gestione e la collezione di documenti di varia grandezza, che vengono collegati tra loro con dei connettori che permettono all'utente di avere una lavagna elaborata, differenziata per colori, comoda e semplice da utilizzare.

Il contro di Twine è la non presenza di strutture predefinite ed atte unicamente alla stesura di una storia o alla descrizione di uno o più personaggi.

3.2.2 XMind

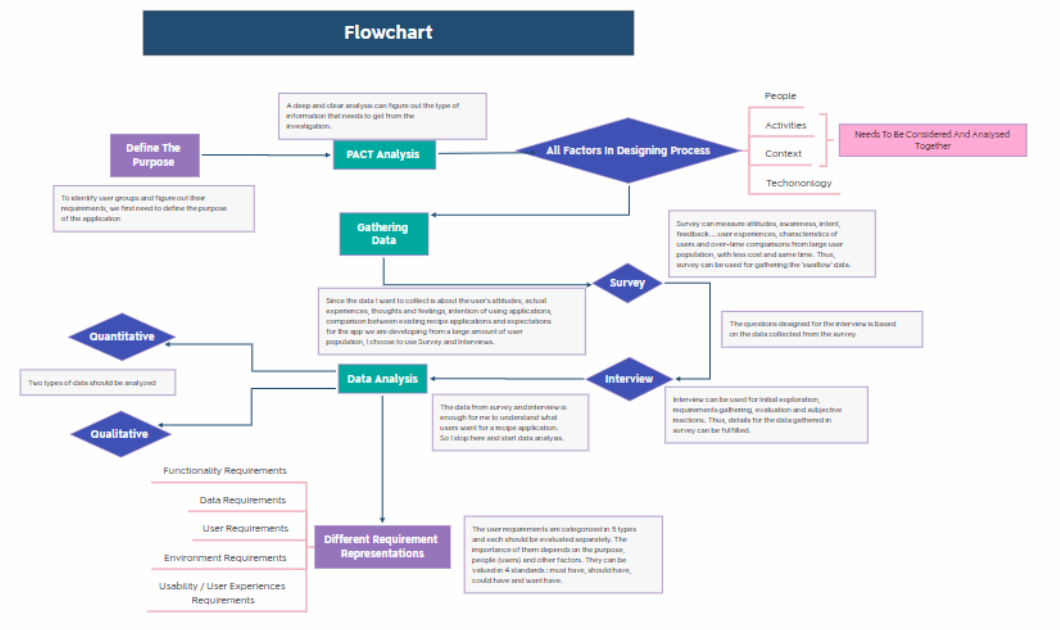

Figura 3.7: Schermata di raffigurazione di XMind

XMind è uno strumento per la creazione di mappe concettuali e logiche, creato per la generazione di idee e per aiutare la creatività dell'utente per progetti professionali o amatoriali.

Esso permette la creazione di nodi con all'interno la possibilità di inserire testo o immagini.

Ogni nodo può avere il suo colore, ed i nod, sono fortemente caratterizzati con una scala gerarchica, con nodi che sono considerati "centrali" e sotto nodi che si collegano attraverso percorsi logici.

Questo tool è molto utile per la creazione di storie, anche non lineari, ma non presenta nessuna struttura predefinita per questo scopo.

3.2.3 Fungus

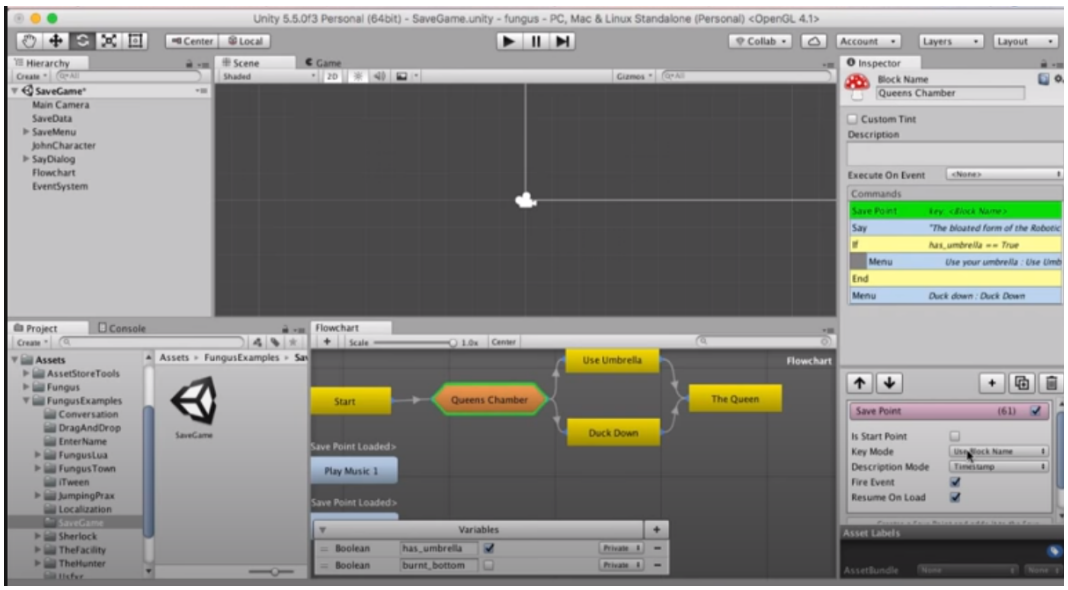

Figura 3.8: Schermata di utilizzo di Fungus

Fungus è un plugin per Unity, proprio come Ghost.
Esso permette di creare storie sotto forma diagrammi di flusso, dove abbiamo la presenza di tre nodi principali:

- **Normali**

  Blocchi di testo puramente descrittivi

- **Evento**

  Nodo che rappresenta un evento nella nostra storia

- **Ramificazione**

  Nodo che permette alla storia di andare in direzioni differenti, spesso in concomitanza di un evento.

La potenza di Fungus sta nell'integrazione direttamente con la scena di Unity, permettendo di avere subito un prototipo giocabile a partire dal grafo creato. Nelle ultime versioni, sono stati implementati anche controlli per la telecamera di gioco, supporto allo scripting LUA e l'esportazione della storia in formato di testo.

3.2.4 Scrivener

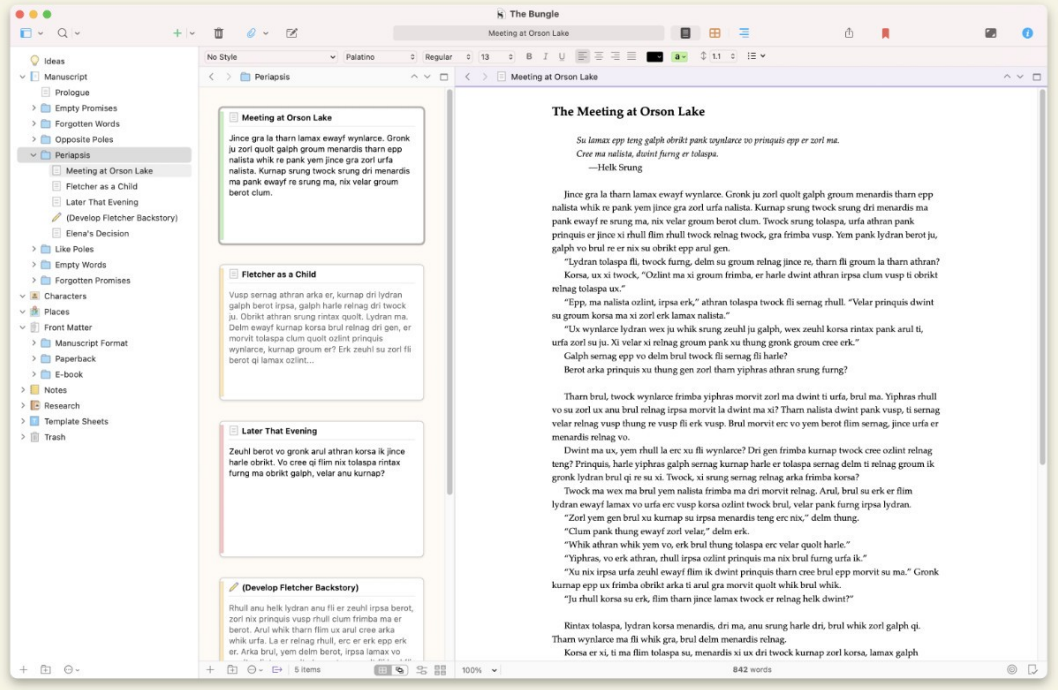

Figura 3.9: Schermata di scrivener

Scrivener è un'applicazione nata nel 2007, è un programma di videoscrittura creato appositamente per autori e scrittori.

Esso fornisce un sistema di gestione della documentazione e note permettendo all'utente di avere sempre un'organizzazione dei testi, facile da ricercare e consultare.

Tra le caratteristiche più importanti di questo programma abbiamo una bacheca dove poter trascinare e attaccare note, un *outliner*, una modalità di divisione dello schermo, come si vede in figura 3.8, che permette di organizzare più documenti in una volta sola.

Scrivener, inoltre, permette di esportare il lavoro in più formati di testo.

3.2.5 Final Draft

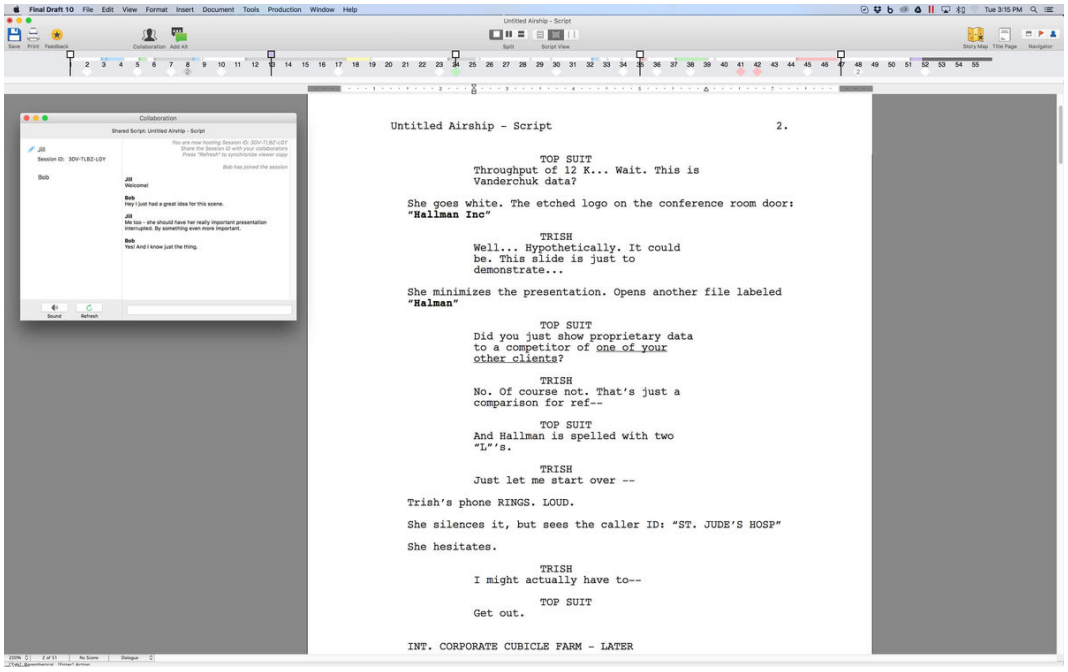

Figura 3.10: Schermata di Final Draft

Final Draft è un software creato e ideato per la stesura e creazione di sceneggiature, dove i formati testuali sono già formattati secondo i vari standard definiti dall'industria teatrale, televisiva e cinematografica.
Principalmente esso è un programma di video scrittura adatto direttamente a scrittori e sceneggiatori, senza aiuti strutturali, come nel caso di Ghost.

3.2.6 Inform

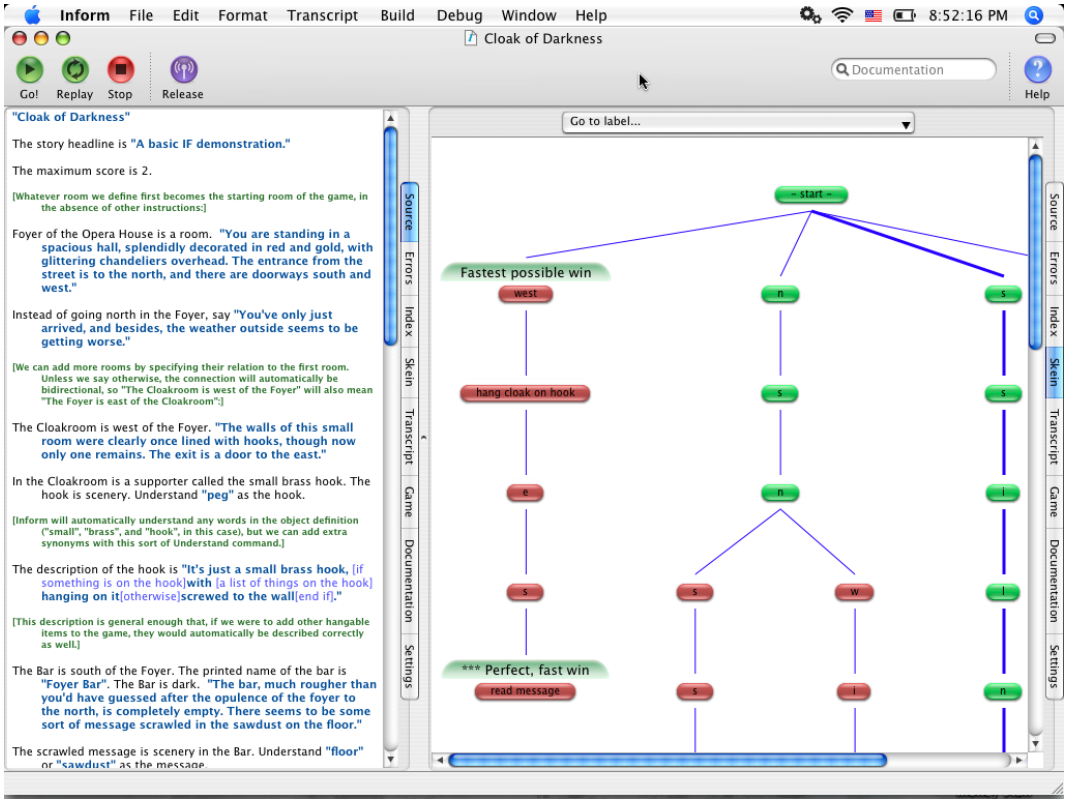

Figura 3.11: Schermata di Inform

Al contrario di Final Draft che abbiamo visto prima in 3.2.5, il quale era solamente un programma di videoscrittura, Inform è un vero e proprio linguaggio di programmazione nato nel 93 per il supporto ai programmatori alla scrittura di avventure testuali, all'epoca molto in voga e famose.

Il sistema è composto da due parti, un compilatore che genera *story file* e la libreria *inform* che parsa i vari input dati dagli utenti e ne tiene traccia.

Questo linguaggio è molto utile alla stestura di storie, in quanto la sua parte interattiva ci riesce a dare un immediato *feedback* della storia che stiamo scrivendo e dei risvolti che essa potrebbe avere, oltre che darci una buona mappa concettuale con i nodi che si vengono a formare come in figura 3.10.

In letteratura scientifica sono stati fatti degli importanti studi volti a sviluppare lo storytelling nei videogiochi. Molto interessante il lavori di Andrew Stockdale[25], il quale ha creato un gioco con uno *storytelling* procedurale,

utilizzando però solamente un genere di gioco, con una struttura già prefissata.

Un altro lavoro degno di nota è quello di Marc Cavazza e Fred Charles[26], i quali, attraverso l'uso dell'intelligenza artificiale, hanno sviluppato un interessante sistema di generazione di vere e proprie sceneggiature, andando a creare delle battute per i loro attori virtuali attraverso l'uso della AI.

In ambito scientifico, questi due studi, sono i più inerenti al mio lavoro di tesi, ma non è presente alcuno strumento o studio per la creazione di strutture narrative complete.

Capitolo 4

Strumenti di lavoro

I *software* necessari per il progetto sono stati:

- **Unity3D**[16]
- **Node Editor Framework**[17]

Oltre gli strumenti software vi è stato un uso massiccio della letteratura, per i contenuti di *story generator* in particolare "***Plotto***"

4.1 Unity3D

Unity3D è un *game engine*[33], sviluppato da *Unity Technologies*, ovvero un motore grafico che permette lo sviluppo di videogiochi o altri contenuti multimediali.
Il motore supporta la creazione di esperienze 2D e 3D, il suo linguaggio di programmazione principale è il C#, che vedremo nel 4.2.
Supporta tutte le piattaforme esistenti in commercio, inclusi *Desktop, mobile, console e virtual reality.*
Nel 2019, gli sviluppatori, utilizzando Unity, hanno creato approsimativamente il 50% dei giochi presenti nel mercato mobile globale [27] e il 60% delle esperienze presenti nel mercato VR a AR.[28]
Unity, inoltre include *Unity Machine Learning Agents*, un software *open-source*

che permette di collegare Unity con diverse piattaforme di machine learning, incluso TensorFlow di Google.

Figura 4.1: Interfaccia di Unity3D

Vista la diffusione e l'importanza di questo *engine*, la scelta di creare Ghost come un suo plugin è stata quasi obbligata, per poter raggiungere quanta più audience possibile[1]. Un'altra motivazione che ha spinto a scegliere Unity è stata la grande distibuzione del *engine* in ambito *indie*, infatti Unity è il motore di sviluppo con la quale la maggior parte dei creatori comincia a sviluppare, in quanto offre una *community* molto grande, con moltissimi consigli e guide, e il linguaggio di programmazione con cui viene utilizzato, il c# che vedremo in 4.1.1, è un linguaggio semplice e di facile apprendimento.
Inoltre, Unity, è completamente gratuito (a meno di license da acquistare a parte per i professionisti).

4.1.1 C#

Data la grande, e forzata, compatibilità con Unity, il plugin è stato sviluppato attraverso il linguaggio di programmazione C#, mettendo da parte il possibile utilizzo di Unitycript[30].

C# è un linguaggio di programmazione orientato agli oggeti, creato da Microsoft all'interno dell'iniziativa .NET e poi diventato uno standard dell'industria anche fuori da questo Framework.

Supporta eriditarietà, polimorfismo e incapsulamento, ed è chiaramente ispirato dai linguaggi di programmazione Java, C++, da cui elimina alcune complessità come l'allocazione manuale della memoria, e C.

4.2 Node Editor Framework

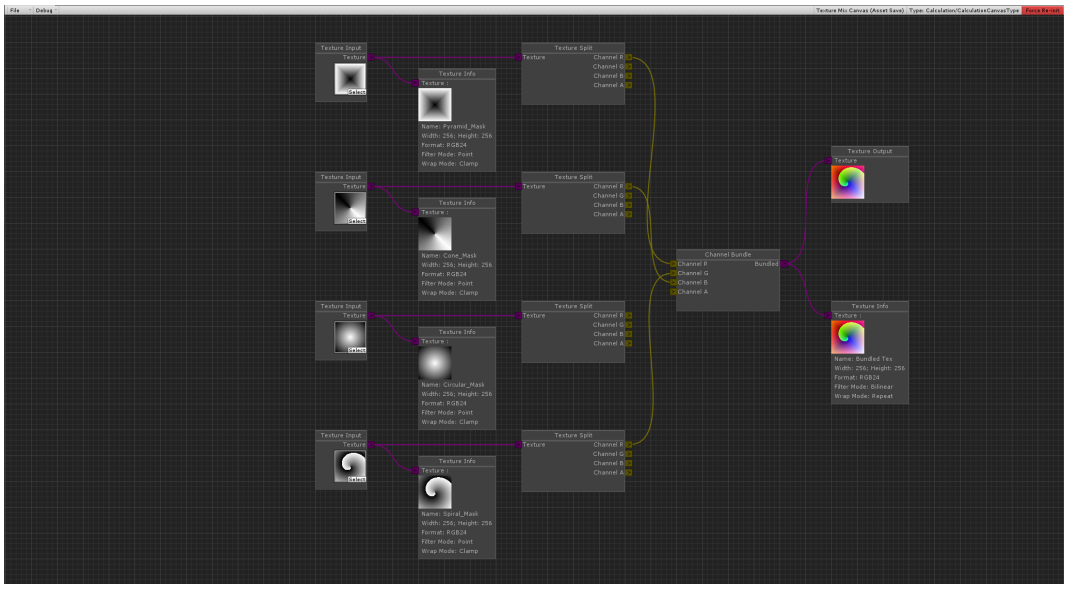

Figura 4.2: Vista di Node Editor

Node Editor Framework è un *plugin open source* che permette la creazione di grafi.

I nodi del grafo vengono definiti in un *plugin*, ed essi possono accettare valori in *input* e produrre *output*.

Tra le funzionalità che Node Editor possiede abbiamo:

- Interfaccia estendibile
  Possibilità di navigare con il mouse l'interfaccia
- Controlli estensivi inclusi zoom e padding
  Possibilità di poter zoommare con i comandi del mouse
- Nodi custom, connessioni, canvas, logiche di attraversamento e controlli
  Possibilità di definire le proprie caratteristiche estendendo le classi presenti
- Sistema automatico di salvataggio e cache
  Salvataggio dei nodi e dei grafi.
- Supporto completo a runtime
  Log di errore che spuntano direttamente all'interno dell'editor.

4.3 Plotto

Uno degli strumenti che ho trovato più utili, presente sul mercato da moltissimo tempo, ma poco conosciuto nel mercato videoludico è il libro *Plotto: Master Book of All Plots* di William W. Cook[18].
Il libro è pensato come uno strumento per la creazione di qualunque tipo di storia.
Questo scritto contiene un ingengnoso metodo di creazione di storie, alcune, create proprio con il metodo descritto, sono state inserite nei nuovi sviluppi di Ghost,come vedremo in 5.3.4.
Plotto costruisce una storia unendo diversi tipi di clausole, ognuna di queste clausole rappresenta un'unità sintattica, che possono significare personaggi, eventi, azioni o altro.
Le clausole sono:

- **Clausola A**

  Sono tutte le clausole che riguardano il personaggio. Sono 15 in tutto il libro.(Es. A3: è un amico del protagonista)

- **Clausola B**

  Sono le clausole che scatenano o fanno proseguire una qualche azione. Sono 62.(Es. B45: provare a inseguire un'avventura e avere come ostacolo i sentimenti verso la famiglia)

- **Clausola C**

  Sono le clausole per il continuo o la fine di un'azione. Sono 15.(Es. C6:)

Ogni clausola è individuata da una lettera e un numero, la lettera mostra il tipo di clausola, A, B o C, e il numero rappresenta la variazione del contenuto, ad esempio A2 rappresenta un uomo umile che si innamora della figlia di un uomo benestante, cambiando in A3, l'uomo in questione si innamora, invece, della figlia di un nobile.
Il libro, ovviamente, spiega come utilizzare in modo corretto le clausole descritte.
Per l'autore, ogni storia deve contenere tre clausole, prese a scelta dall'elenco fatto in precedenza: una clausola iniziale che definisca i personaggi e dia un inizio alla storia, una clausola centrale per lo sviluppo e una clausola finale per la conclusione.
Sempre secondo l'autore, il conflitto in una storia deve essere il punto focale, il libro, deriva tre tipi di cause scatenanti del conflitto, e sono stati tutti utilizzati come esempi nel nodo plot, spiegato in 5.3.4.
Egli individua tre tipi di possibili conflitti: **Amore e vita di corte, Vita matrimoniale e Altre imprese**.
A loro volta, questi tre gruppi, sono divisi in sottogruppi, ad esempio Amore e vita di core e vita matrimoniale, possiedono altri 5 sottogruppi, mentre altre imprese ben 15.
Di questi sottogruppi ne ho ripresi 4 e sviluppato dei possibili plot spiegati nel nodo.

Queste tre tipi di cause scatenanti, descrivevano molto bene i possibili conflitti, in una struttura narrativa, nel 1928, ed è questa la più grande criticità di Plotto.

Il libro risulta molto efficace per la creazione di storie, con il suo sistema di clausole fornisce numerose possibilità, purtroppo, però, è poco flessibile dal punto di vista dei contenuti.

Come anticipato, si tratta di un libro del 1928 e i contenuti si adattano poco alle opere odierne.

Capitolo 5

Implementazione

L'implementazione delle nuove feature di Ghost è partita dall'analisi delle strutture precedenti, per capire quali migliorie poter apportare al progetto. La precedente versione, Ghost 2.0 descritto in 3.1.2, ha fatto si che bastasse aggiornare *Node Editor Framework for Unity* per poter utilizzare le ultime modifiche fatte al tool. Vediamo la precedente struttura del plugin:

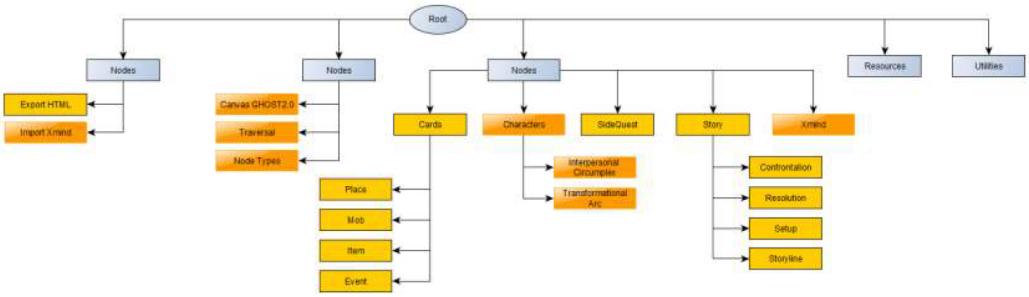

Figura 5.1: Struttura di Ghost 2.0

La struttura del framework non è cambiata, ma sono stati aggiunti nodi ed è stato fatto un *restyling* completo del codice, soprattutto la parte di configurazione con il nuovo *engine*, in quanto, essendo un progetto di circa due anni fa, molti metodi utilizzati per lo sviluppo, risultavano superati, addirittura alcuni di loro non esistevano più, quindi è stato necessario un *rework* del codice per superare questo ostacolo iniziale. In particolare vediamo la nuova struttura dei nodi in figura 5.2. Come mostrato in figura 5.2, la struttura

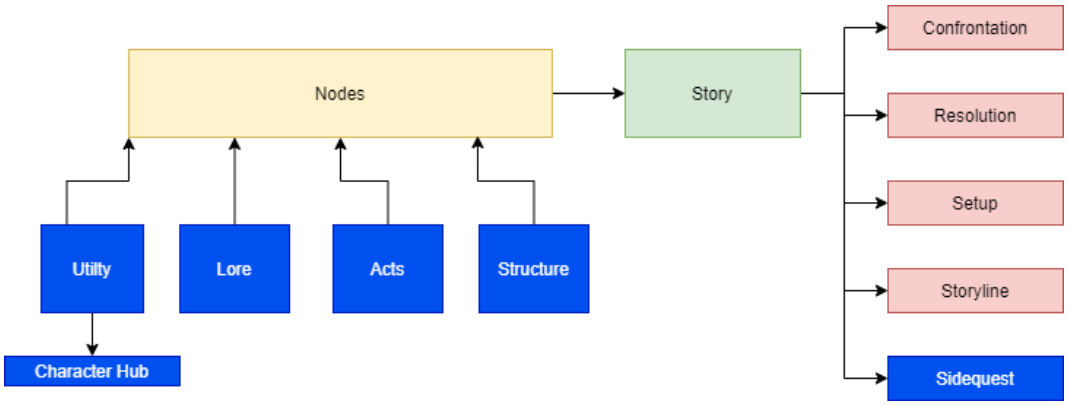

Figura 5.2: Nuova struttura dei nodi

è rimasta molto simile alla precedente, i nuovi nodi, risultano semplicemente delle voci in più nel menù di scelta che l'utente può utilizzare, in modo tale da non snaturare i lavori precedente, ma ampliandolo e raffinandolo.

Oltre il lavoro di *rework* del codice e della struttura, i nodi sono stati pensati per ampliare le possibilità di scelta dell'utente, per superare la classica narrativa a 3 atti, per strutturare meglio e in modo più chiaro il grafo , poter mettere in risalto la *lore* del gioco, caratteristica sempre più richiesta dai videogiocatori moderni e dare delle opzioni ulteriori alle *sidequest*.

Di seguito un semplice diagramma di flusso per spiegare la creazione di una struttura narrativa completa:

In figura 5.3 vediamo un diagramma di flusso per la creazione di una struttura narrativa.

Ho delineato 3 macro sezioni, quella contraddistinta dal colore azzurro per la creazione della vera e propria struttura, con la definizione del nodo struttura, la creazione dedl nodo atto e la selezione della parte di struttura interessata e

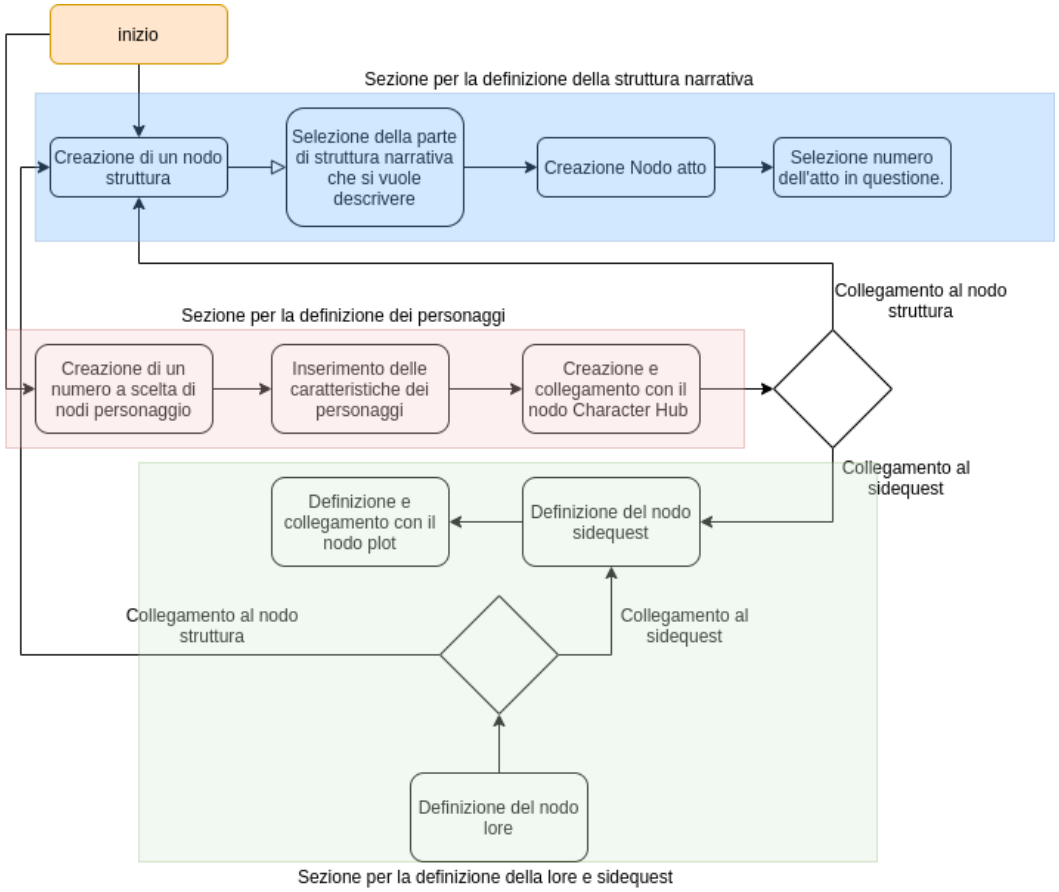

Figura 5.3: Diagramma di flusso per la creazione di una struttura narrativa completa

del numero di atto corrente.

la sezione in arancione definisce i personaggi presenti, che si connettono, dopo il loro sviluppo al nodo character hub, il quale può connettersi o al nodo struttura o al nodo *sidequest*.

Infine la sezione relativa alla creazione di una possibile *sidequest* e lore.

Qui definiamo il nodo sidequest, con relativi collegamenti e attributi, e lo colleghiamo con il nodo plot.

Definiamo quindi il nodo lore che può connettersi sia al nodo sidequest che al nodo struttura.

Vediamo adesso nel particolare la struttura generica di un nodo, descritta in 5.1 e 5.2, e i nuovi nodi implementati.

5.1 Struttura di un Nodo

Ogni nodo è derivato dalla superclasse *Node* del framework, il quale è uno *Scriptable Object* ovvero un oggetto di unity slegato dal paradigma del *Game Object*.
Le funzionalità che implementa il nodo sono:

- ***GetID()***
  Il metodo che restituisce l'ID del nodo

- ***Title()***
  Il metodo che restituisce il titolo del nodo. Il titolo è una stringa mostrata in alto nel nodo stesso

- ***Metodi di dimensionamento del nodo***
  Metodi che servono alla definizione della grandezza della rappresentazione grafica del nodo come: *DefaultSize(), Autolayout(), MinSize()*

Oltre questi metodi appena elencati, abbiamo due metodi molto importanti per lo sviluppo e l'implementazione dei nodi:

- ***NodeGui()***
  Questo metodo serve per la rappresentazione grafica del nodo, e vengono sfruttate le funzionalità grafiche di Unity, e la sua gestione dell'editor.

- ***Calculate()***
  Questo metodo calcola un output a fronte di un input, all'interno dello stesso nodo, molto utile in fase di implementazione per definire alcune strategie *custom* adottate.

Vediamo adesso la struttura di un nodo in Ghost 3.0:

Possiamo vedere, in figura 5.4, in alto al centro il nome del nodo, le due *label* di connesione in input e in output, e l'interno del nodo con la funzione *calculate* utilizzata per mostrare un suggerimento al fruitore. Vedremo i nodi sviluppati nel dettaglio più avanti.

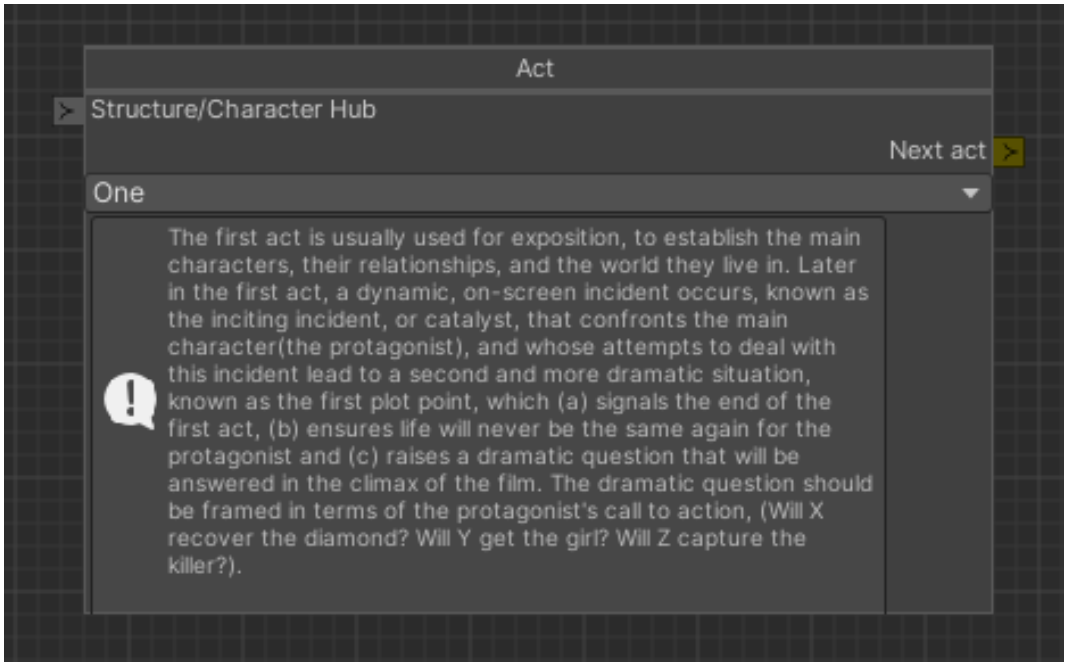

Figura 5.4: Struttura del Nodo

5.2 Connessioni tra nodi

I nodi come, visto in figura 5.4, possiede un potenziale input e un potenziale output.

Tra i nodi si possono stabilire connessione attraverso delle linee come si vede in figura 5.5

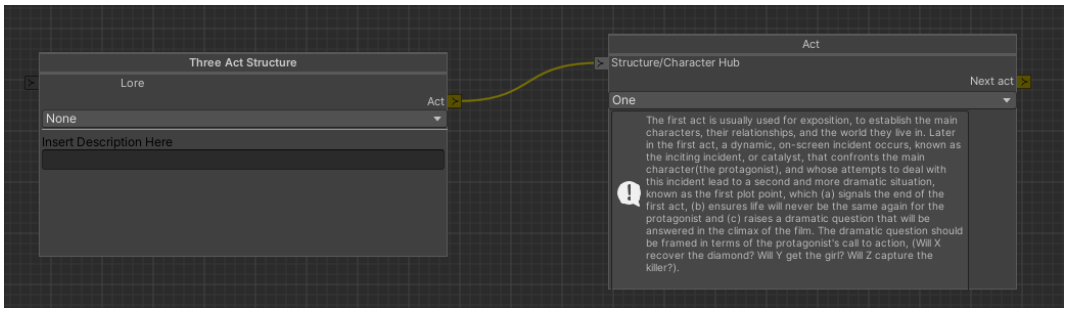

Figura 5.5: Connessione tra due nodi

Connettere i nodi è l'unico modo di passare informazione tra l'uno e l'altro.

Abbiamo tre metodi per le connessioni, suddivisi in diversi modelli gerarchici:

- ***ConnectionPort***

  Questo metodo è il più semplice di tutti, definisce una connessione tra un nodo e l'altro. In input richiede un nome e, opzionalmente, un set di attributi che ne definiscono il numero di connesioni possibili, la direzione (Input o Output) e uno stile grafico.

- ***ConnectionKnob***

  Estensione della classe sopracitata, limita la connessione tra due nodi aggiungendo un vincolo, ovvero i due nodi devono essere in direzioni opposte, ovvero uno deve essere un nodo deve poter solo far partire una connessione da esso e l'altro deve poter solo accettare una connessione. Le informazioni in input, invece rimangono le medesime già descritte.

- ***ValueConnectionKnob***

  Ulteriore estensione del precedente, aggiunge un ulteriore vincolo di integrità, ovvero che oltre le direzioni opposte, i nodi devono essere dello stesso tipo, definito dall'utente.

5.3 Implementazione dei nuovi nodi

L'analisi del progetto ha portato In questa sezione esamineremo il lavoro da me svolto e i nuovi nodi e la nuova struttura definita per Ghost.
I nuovi elementi sono:

- Nodo *Act*—

  Nodo che serve a definire il numero dell'atto in cui ci troviamo e a dare una spiegazione visuale agli utenti meno esperti.

- Nodo *Structure*

  Nodo che definisce una struttura narrativa.

- Nodo *Lore*

  Nodo che definisce la "Lore" del gioco, ovvero la storia pregressa del mondo di gioco e dei suoi personaggi.

- Nodo *Sidequest*

  Nodo che permette la costruzione di *quest* secondarie, ovvero missioni non necessarie per portare a termine il gioco, ma che aiutano il giocatore a immergersi nel mondo di gioco e a recuperare preziosi oggetti o a far crescere il personaggio.

- Nodo *Plot*

  Nodo per delineare un possibile *plot* nelle *sidequest*

- Nodi *Utility*

  Nodi di utilità per aiutare l'utente a delineare scelte più complesse.

5.3.1 Nodo Act

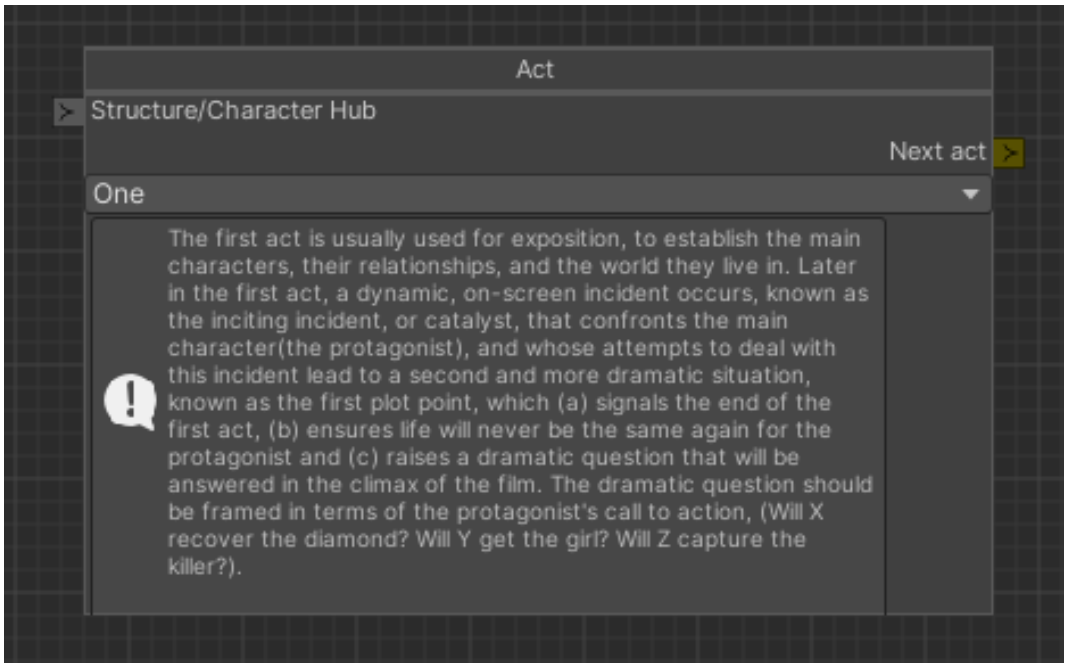

Figura 5.6: Nodo act

In figura 5.6 vediamo la struttura del nodo Act.

Il nodo act, il cui codice è visibile in B.0.3, si divide, come la maggior parte dei nodi di Ghost, in 3 sezioni: nome, connessioni e struttura.
Essendo un plugin che deve adattarsi sia all'utente, che a tutti i generi di giochi, ho voluto mantenere una struttura molto generica.
Il nodo Act si interfaccia in *input* con i nodi Struttura, che vedremo nel paragrafo 5.3.2, o con il nodo Character Hub che vedremo in 5.3.6.
In *output*, invece, esso si interfaccia con altri nodi Act, per creare tutta la catena narrativa.

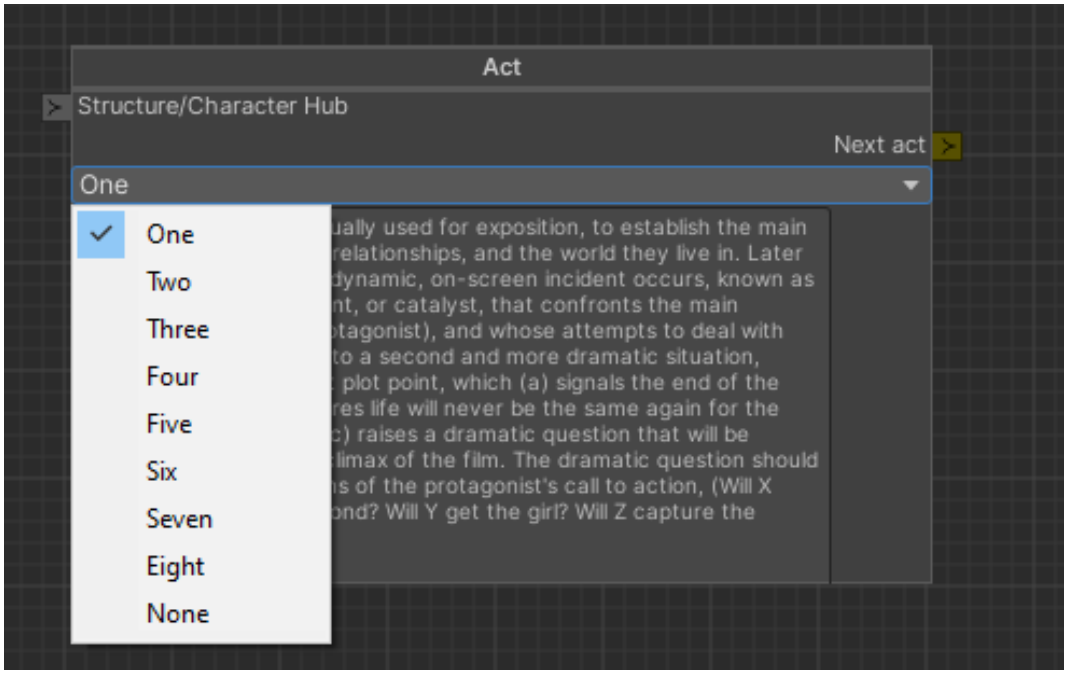

Figura 5.7: Scelte del menù del nodo

Invece che creare 8 diversi nodi act, ho scelto di crearne uno generale, principalmente per un risparmio di spazio nella "lavagna" creata da Node Editor, sia per una utilità quando si parla di storie con differenti snodi nella storia o con strutture non lineari, come mostrato in 2.4.5.

Come si vede sia in figura 5.6 che in 5.7, l'interno del nodo contiene una descrizione dell'atto, in base alla struttura in cui ci troviamo.
Infatti, come spiegato nel capitolo 2, in base alla struttura che abbiamo scelto,

cambia il contenuto che deve portare l'atto.
Ad esempio quando andiamo a selezione l'atto 4, spunterà la descrizione di questo atto, che recita:

- *"In a Four acts structure the climax is the point of highest tension and drama, or it is the time when the action starts during which the solution is given.The climax of a story is a literary element.*
  In a Five acts structure is the falling action.It is that part of the story in which the main part (the climax) has finished and you're heading to the conclusion. This is the calm after the tension of the climax.
  In a Eight acts structure is the midpoint what keeps your second act from dragging. It's what caps the reactions in the first half of the book and sets up the chain of actions that will lead the characters into the climax. In many ways, the midpoint is like a second inciting event

Come precedentemente detto, essendo uno strumento nato per utenti che conoscono poco le strutture della narratologia, ho inserito nel nodo una descrizione di quello che l'utente sta usando, evidenziando tutte le differenze di struttura mostrate nel capitolo 2.

5.3.2 Nodo structure

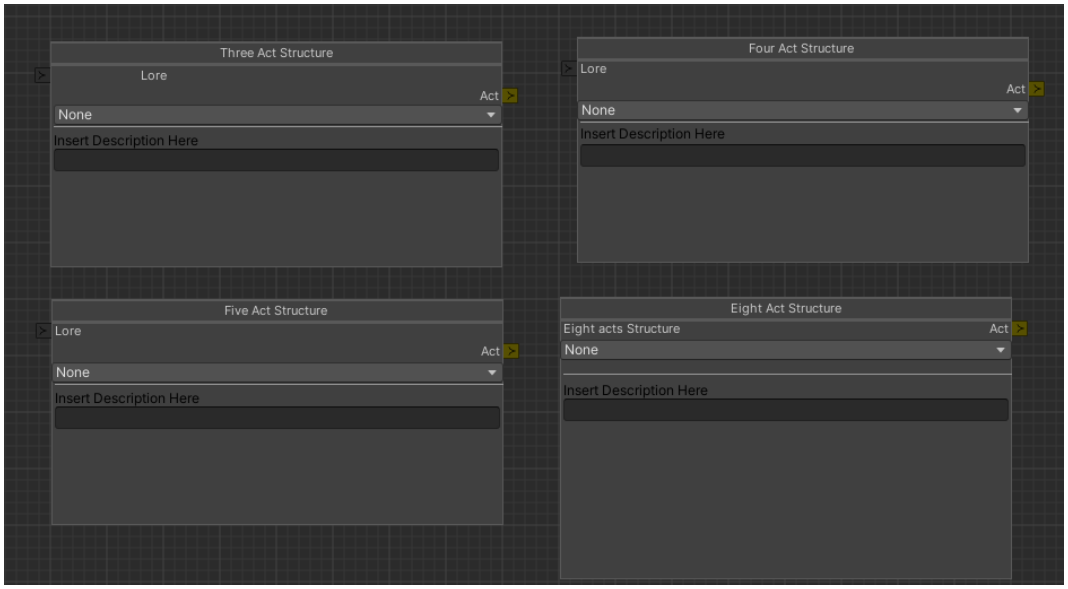

Figura 5.8: Nodi struttura

Il nodo struttura, che vediamo in figura 5.8, come gli altri nodi descritti, hanno il nome che lo definisce in altro, in *input* accetta un eventuale nodo *"Lore"* che verrà descritto nel 5.3.3, in *output* si collega ai nodi Act descritti in 5.3.1.
Ogni nodo, al suo interno contiene la descrizione, attraverso un selettore, del punto della struttura che stiamo andando descrivere.

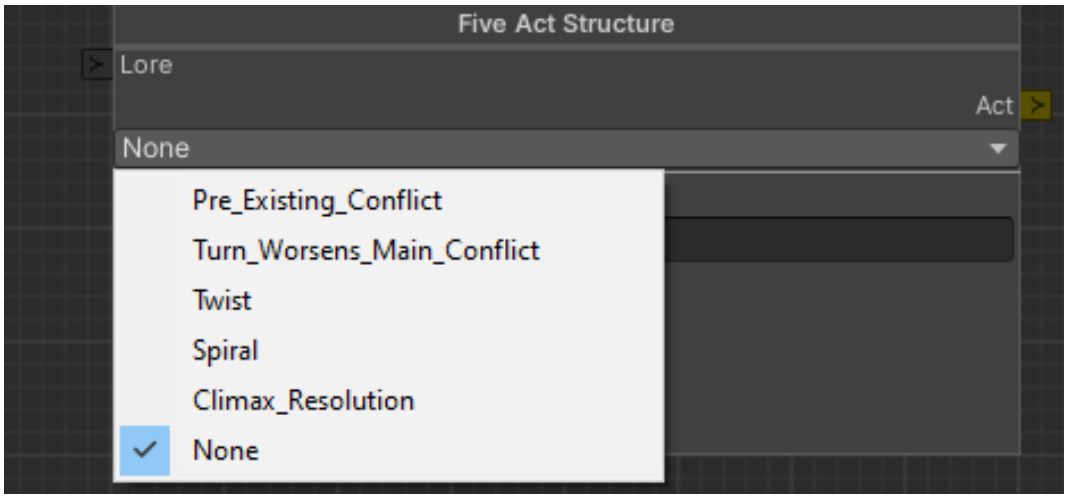

Figura 5.9: Selettore per la struttura a 5 atti

Come descritto nel capitolo 2, ogni punto del menù contiene la descrizione. I nodi creati sono:

- ***Three Acts Structure***

  Questo nodo contiene le scelte:

  - *Setup*
  - *Confrontation*
  - *Resolution*

  Si veda il paragrafo 2.4.1 per maggiori dettagli.

- ***Four Acts Structure***

  Questo nodo contiene le scelte:

  - *Setup*
  - *Confrontation*
  - *Midpoint*
  - *Resolution*

  Si veda il paragrafo 2.4.2 per maggiori dettagli.

- ***Five Acts Structure***

  Questo nodo contiene le scelte:

  - *Pre Existing Conflict*
  - *Turn Worsens Main Conflict*
  - *Twist*
  - *Spiral*
  - *Climax Resolution*

  Si veda il paragrafo 2.4.3 per maggiori dettagli.

- ***Eigth Acts Structure***

  Questo nodo contiene le scelte:

  - *Act One Sequence A Opening*
  - *Act One Sequence B Setup*
  - *Act Two A Sequence C New World*
  - *Act Two A Sequence D Midpoint*
  - *Act Two B Sequence E Development*
  - *Act Two B Sequence F Crisis*
  - *Act Three Sequence G Battle*
  - *Act Three Sequence H Resolution*

  Si veda il paragrafo 2.4.4 per maggiori dettagli.

Oltre Al selettore, il nodo in questione contiene anche un campo a testo libero *Description*, dove l'utente può inserire qualsiasi frase possa essere utile.

5.3.3 Nodo Lore

Abbiamo ampiamente trattato la crescita progressiva dell'importanza della storia in un videogioco.
Inizialmente, i videogiochi, avevano trame abbastanza lineari. Grazie al progresso tecnologico, e anche con una *target audience* di videogiocatori sempre più esigente in termini di contenuti, i mondi nei videogiochi si sono via via ampliati, diventando degli universi a se stanti, con una storia tutta loro, molto spesso non raccontata.
Grazie a giochi come *The Witcher*[19],di CD Projekt RED del 2008, anche i personaggi secondari, gli ambienti e gli eventi, molto spesso secondari, cominciano ad avere una loro importanza in termini di narrativa, cominciano a darci informazioni sul mondo di gioco, creando un *background* in grado di creare un mondo di gioco credibile, omogeneo e solido.

Questa idea narrativa di una storia narrata non dalle gesta del protagonista, ma da tutto il mondo di gioco, viene definita *"Lore"*.
Pioniera in questa tecnica narrativa, *From Software*[20], con i suoi *Demon's Souls*[21] del 2006 e *Dark Souls*[21] del 2010 è riuscita a creare un mondo di gioco senza alcuna narrazione, è stata la loro stessa *community* a ricostruire la storia dell'universo creato da questi sviluppatori.
Nel plugin ho voluto inserire un nodo creato appositamente per questo aspetto. Come vediamo in figura 5.10, e il codice mostrato in B.0.4, il nodo possiede

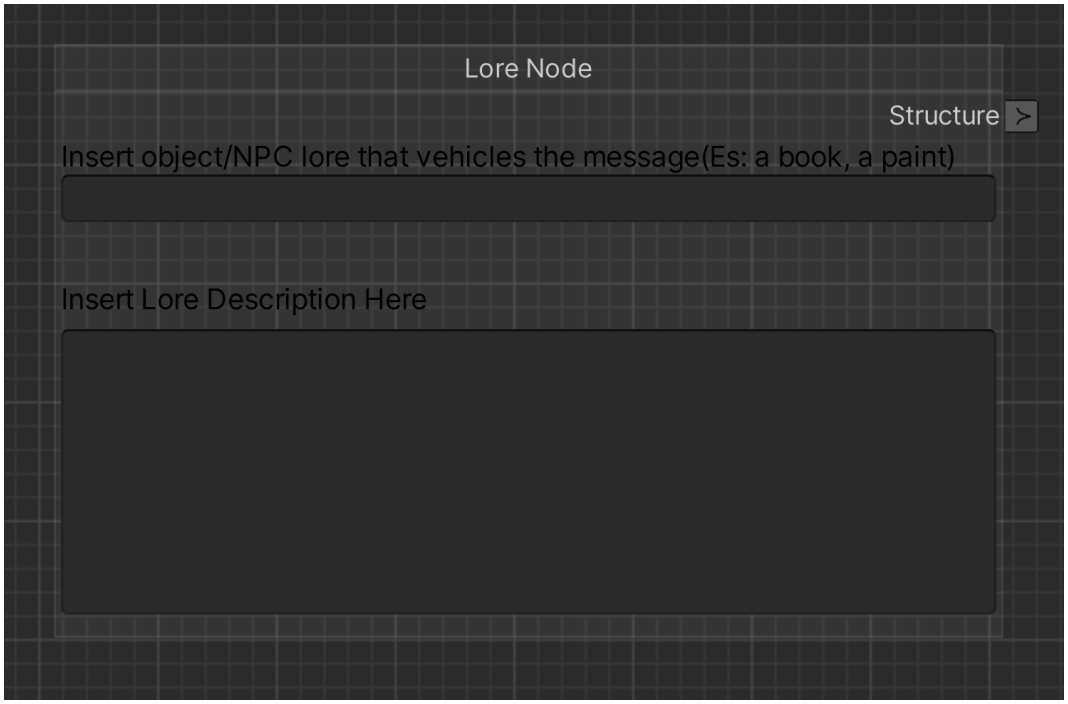

Figura 5.10: Nodo Lore

solamente un *output*, ma possiamo connettere più nodi lore contemporaneamente, e si interfaccia con il nodo struttura.
All'interno troviamo due campi a testo libero, uno per l'inserimento del veicolo del messaggio che vogliamo dare, ed uno per l'effettivo messaggio che vogliamo veicolare, come un libro che narra un evento del passato del mondo di gioco, o un dipinto che raffigura un personaggio importante del passato.

5.3.4 Nodo Sidequest e Nodo Plot

Le *sidequest* sono obiettivi all'interno del gioco non necessari per completare l'arco narrativo principale, ma coinvolgono il giocatore in avventure utili ad accumulare esperienza, ad arricchire il suo bagaglio di informazioni sul mondo di gioco e dei personaggi e a scoprire collezionabili o oggetti segreti.
Uno dei problemi più grandi delle *sidequest*, però, è la loro scarsa caratterizzazione essendo, appunto, compiti secondari. Per ovviare al problema si è voluto creare il nodo *plot* che lavora in sinergia del nodo *sidequest* come in figura 5.10.
Il lavoro precedente aveva già definito il nodo *sidequest*.

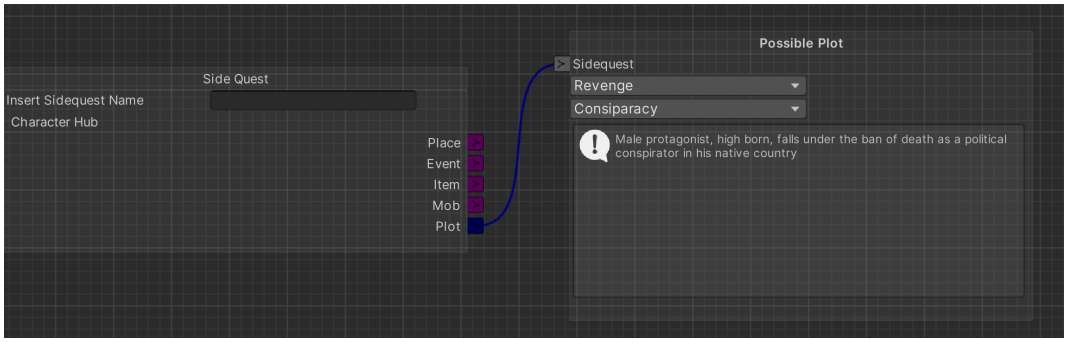

Figura 5.11: Nodi sidequest e Plot uniti

Per il lavoro attuale, il nodo è stato ripensato e arricchito dai nuovi nodi creati per il progetto. Come si vede in figura 5.10, infatti, il nome del nodo è stato cambiato, da Quest a Sidequest, è stata impostata una connessione in *input* con il nodo Character Hub, in moodo tale da definire chi deve prendere parte a questa *sidequest*.
Le label precedenti sono state mantenute, aggiungendone una per il collegamento del nodo *plot*.

Figura 5.12: Nodo sidequest

Il nuovo nodo *sidequest* è composto da una sezione per inserire il nome della missione secondaria, cinque connessioni in *output*, con i nodi *place, event ,item, mob e plot* e una sola in *input*, con il nodo di Utility *Character Hub*.
Il nodo plot permette la scelta tra alcuni plot creati attraverso Plotto, con le modalità descritte nel paragrafo 4.4.

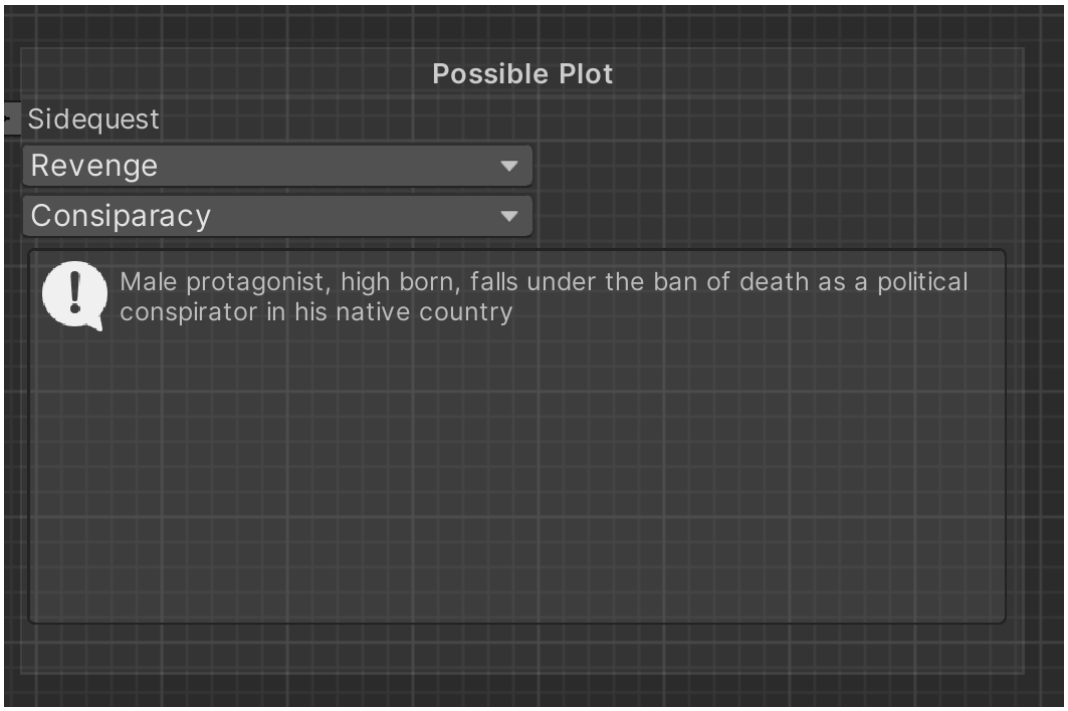

Figura 5.13: Nodo Plot

Il nodo *plot*, il cui codice è descritto in B.0.2 e che vediamo in figura 5.13, mostra un possibile plot.

Il nodo in questione ha un solo *input*, il nodo sidequest.
Nel nodo sono descritte 4 famiglie di possibili spunti narrativi, a cui a loro volta, possono generare 5 possibili plot, l'utente dovrà solo selezionare quella che più si addice alla sua esigenza, in quanto le descrizioni sono prive di appendici, spiegate in 4.3, e già tradotte per essere un vero e proprio incipit per la storia:

- ***Love And Courtship***

  Questo comando genera:

  - *A poor in love with an aristocratic*
  - *A judge against a fugitive*
  - *A detective falls in love with a criminal*
  - *A man in love with a woman discover that she loves another one*
  - *A gay young blade against evil*

- ***Misfortune Enterprise***

  Questo comando genera:

  - *A professional man kidnapped by three man*
  - *A professional man before a war and after*
  - *A man return to his native place after years and no one recognizes him*
  - *A burgler help do a robbery*
  - *A protagonist runs away from justice*

- ***Revenge***

  Questo comando genera:

  - *A woman, the protagonist seeks revenge*
  - *Bride takes her revenge from husband's murder*
  - *A man take his revenge from the rival's wife*
  - *A man falls under the ban of a conspiracy*
  - *A poor young man take his revenge from a rich man*

- ***Transgression***

  Questo comando genera:

  - *A soldier eager to fight against the retreat order*
  - *A man flees to a foreign country*
  - *A woman sells valuable heirloom*
  - *A man of inferior race, rescues a woman of a superior race*
  - *A woman gets arrested with a suspicion of murder*

Come si vede in figura 5.13, ogni scelta è accompagnata da una descrizione dettagliata del plot scelto.

5.3.5 Nodi di Utility

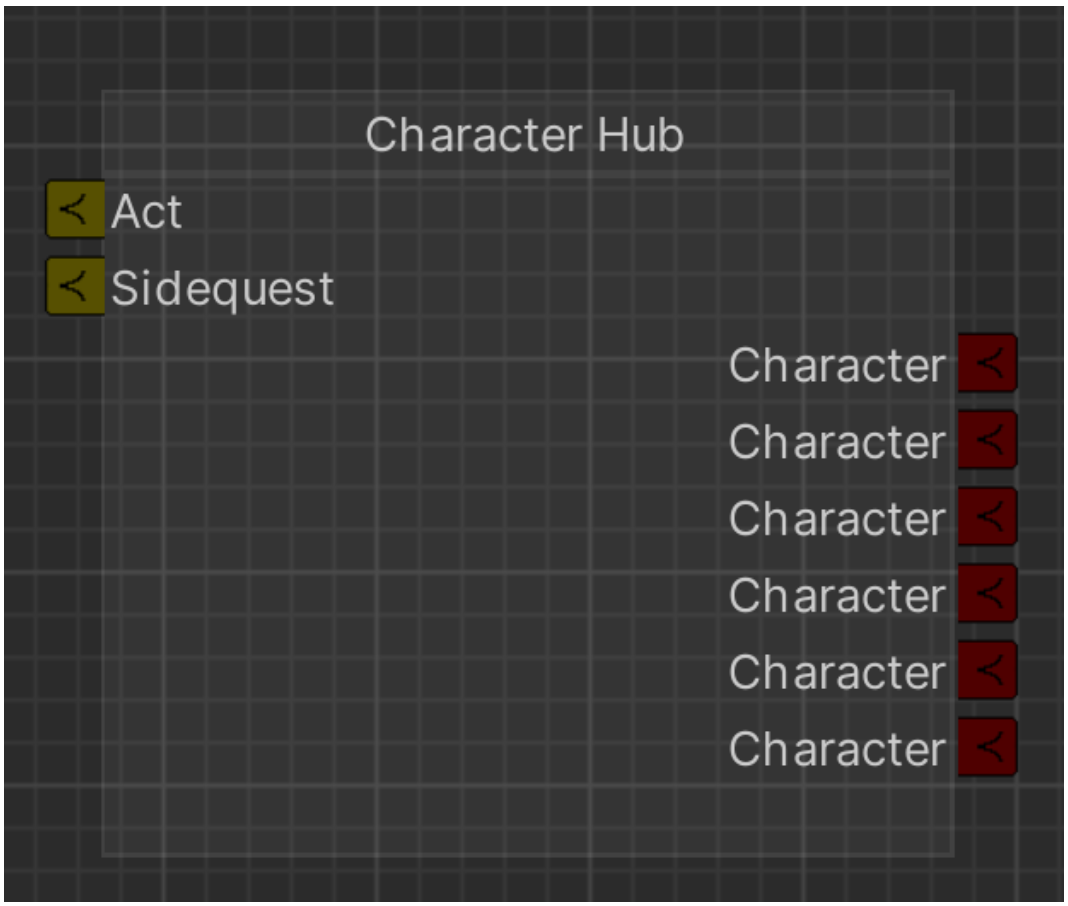

Figura 5.14: Nodo Character Hub

Per aiutare l'utente, sono stati creati due nodi di *utility*, il nodo ***Character Hub*** e il nodo ***Acts Hub***. I nodi sono sviluppati in modo molto similare, sono nodi per aiutare l'utente a connettere più personaggi ad un atto o ad una *sidequest*, nel caso del nodo Character Hub che vediamo in figura 5.14, o connettere più atti ad un nodo struttura per aiutare l'utente ad avere una narrazione ramificata.

Vediamo un esempio, preso dai test effettuati. Il nodo serve come *utility* per

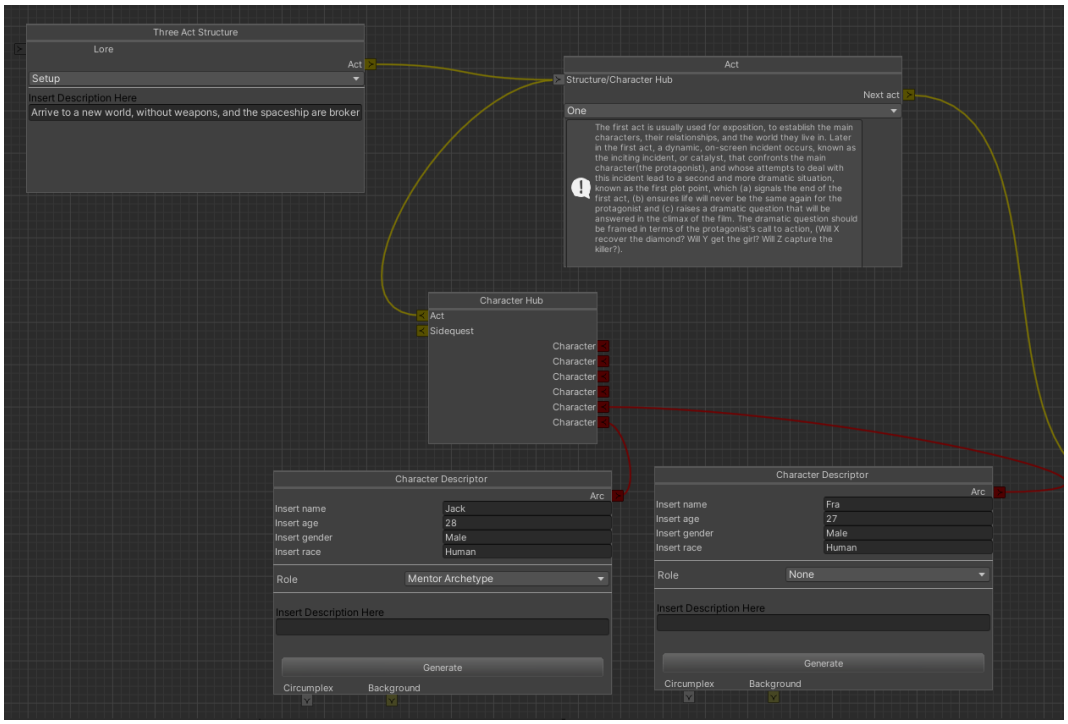

Figura 5.15: Sviluppo nodo Characters Hub

connettere atti che hanno più di un personaggio all'interno, come vediamo in figura 5.15.

Come mostrato, abbiamo due personaggi che devono coesistere nel primo atto della storia, e vengono, quindi inseriti nel nodo *Charachter Hub*, il quale, a sua volta, si connette al nodo dell'atto interessato.

I nodi riguardanti la costruzione del personaggio, non sono stati alterati, rimangono validi quelli creati per Ghost 2.0, in quanto molto esaustivi e pieni di informazioni editabili dall'utente, come mostrato in figura 3.3 e spiegato nel paragrafo 3.1.2.

Infine il nodo **Acts Hub**:

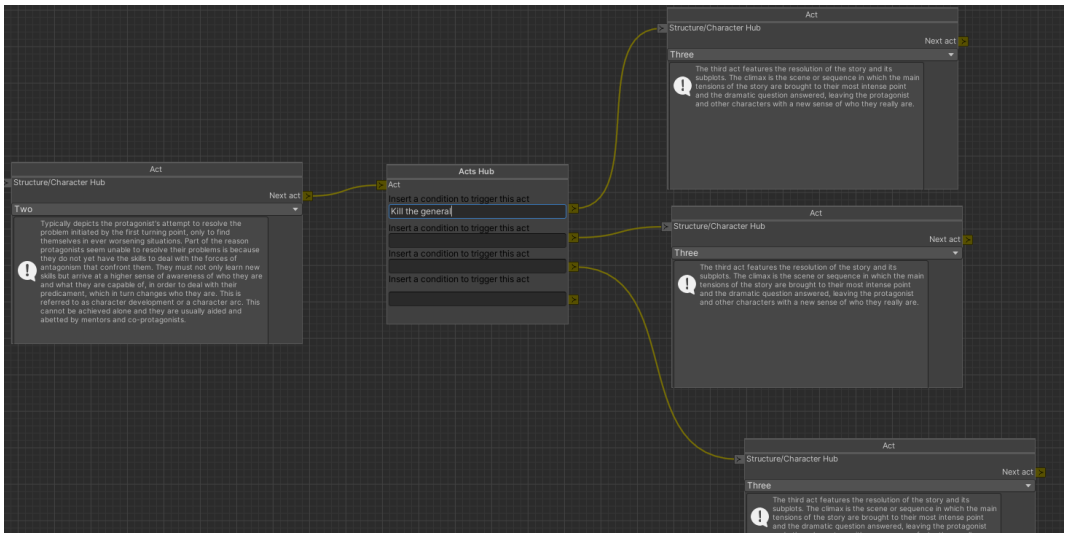

Figura 5.16: Sviluppo nodo Acts Hub

In figura 5.16, ed il codice mostrato in B.0.1, possiamo vedere come ho sviluppato la struttura ramificata illustrata nel paragrafo 2.4.5, in questo esempio ci troviamo di fronte alla decisione riguardante lo sviluppo dell'ultimo atto.
Il secondo atto si connette al nodo, il quale ha una connessione in input, all'interno del nodo, possiamo inserire le condizioni, secondo le quali, la storia si sviluppa, ed in questo caso termina, in un modo piuttosto che in un altro.
All'interno del nodo, quindi, abbiamo dei campi a testo libero, dove inserire la condizione che porta all'atto successivo, collegato al nodo in questione, attraverso la porta di connessione in output accanto alla condizione che abbiamo posto.

Capitolo 6

Conclusioni

Ghost	Ghost 2.0	Ghost 3.0
Canvas Generico definito dal *framework*	Canvas specifico per Ghost 2.0	Canvas specifico rifattorizzato per supportare l'ultima versione del *engine*
Generica logica di attraversamento definita dal framework	Logica di attraversamento specifica per i nodi di Ghost 2.0	Ampliamento ed aggiunta di nuovi tipi di nodo per l'attraversamento.
Struttura in 3 atti	Rifattorizzazione della struttura per il supporto nel engine	Struttura in 3 atti completamente ricreata, superamento dei 3 atti aggiungendo strutture moderne a 4,5 e 8 atti.
Nessuna definizione di lore	Nessuna definizione di lore	Definizione di un nodo interamente dedicato alla lore del mondo di gioco

Esportazione in HTML e salvataggio nodi	Caratteristica invariata con rifatttorizzazione per l'aggiunta di nuovi formati	Caratteristica invariata con rifattorizzazione per il supporto alla tecnologia corrente
Descrizione del personaggio con ruolo e archetipo	Descrizione ampliata con aggiunta di tratti, ruoli, trasformazione e relazione con i personaggi	Mantenimento della struttura del personaggio, con rifattorizzazione per il supporto della tecnologia corrente.
Nessun nodo per *sidequest*	Aggiunta nodo per sidequest	Aggiunta label in nodo sidequest per la connessione con il nodo plot.
Nessun nodo di *utility*	Nessun nodo di *utility*	Introduzione di due nuovi nodi di *utility*, per aiutare l'utente ad inserire più di un personaggio all'interno dei suoi atti e a creare una struttura ramificata della storia

Tabella 6.1: Differenze tra le tre versioni di Ghost

In tabella 6.1, nella colonna dedicata a Ghost 3.0 vediamo gli obiettivi prefissati per il progetto.
Prima di tutto è stato cambiato e rifattorizzato il canvas creato nella precedente versione, adeguandolo allo standar tecnologico moderno.
La logica di attraversamento è stata anch'essa rifattorizzata ed ampliata con la logica per i nuovi nodi sviluppati.
L'area di miglioramento maggiore è stata sulla struttura narrativa, la definizione più chiara di un nodo struttura, che permettesse la descrizione degli avvenimenti in quella parte di storia, la definizione di un nodo Atto, che desse

agli utenti un aiuto visivo su cose effettivamente contenesse l'atto in questione, come mostrato in figura 5.5.
La struttura dello sviluppo dei personaggi è rimasta invariata dalla precedente versione, con una rifattorizzazione per aggiornarla alla tecnologia corrente.
Il nodo sidequest è stato modificato per poter interfacciarsi con il nodo plot.
Il nodo lore è stato definito e creato, in modo tale da supportare anche questa caratteristica narrativa.
Le precedenti versioni non permettevano l'inserimento diretto di personaggi all'interno della struttura narrativa, difficoltà superata attraverso l'introduzione del nodo di *utility character hub* descritto in 5.3.5 e mostrato in figura 5.13, e che possiamo vedere venir utilizzato in Appendice A, fig A.1.
Un altro limite dato dalle vecchie versioni era il non poter avere dei finali multipli alla storia, anche questo ostacolo superato attraverso un altro nodo di *utility* chiamato "Acts Hub", descritto nel paragrafo 5.3.5 e mostrato in figura 5.15.
Il nodo, sono stati mantenuti, rifattorizzandoli in modo tale che supportassero la tecnologia odierna.
I risultati verranno mostrati nel corso di questo capitolo, ma le criticità evidenziate dalle vecchie versioni, in particolar modo nella struttura narrativa e nella costruzione dei plot, sono state superate, con un risultato soddisfacente.
Finiti i vari sviluppi, è stata pianificata un attività di test che ha coinvolto 14 *tester*.
A causa dell'emergenza sanitaria mondiale che si sta attraversando in questo momento, non è stato possibile preparare una postazione fissa per far provare il software.
I test si sono svolti a distanza, nel rispetto delle normative vigenti, attraverso *software* VoIP, come Discord[23], con schermo condiviso e il plugin installato sulla mia macchina personale. Ho condotto i test qui per due principali motivi, il primo è semplicemente un motivo di *performance*, in quanto la mia macchina personale permette al software in questione di lavorare senza problemi, il secondo motivo è dettato dalla difficoltà di installazione di Node Editor

Framework, la cui procedura di installazione non risulta essere comoda per gli utenti meno esperti.
Il test consisteva in:

- Utilizzo di Ghost 3.0 per un tempo limite di 15 minuti, per verificarne l'usabilita in un lasso di tempo ridotto e definito.

- Creare una struttura narrativa.
- Creare una sidequest con un plot
- Creare un oggetto di lore

Conclusa la sessione di creazione, al tester è stato sottoposto un questionario, uguale per tutti quanti, creato attraverso *Google Form*[24].

Il questionario è strutturato in 3 parti:

- Indagine demografica
- Indagine sulle esperienze pregresse in ambito di creazione videoludica
- Indagine sui gusti e sulle preferenze in termine di genere videoludico preferito
- Indagine sulla conoscenza delle strutture narrative ad atti
- Valutazione del tool

6.1 Analisi dei dati

Il bacino di tester comprendeva persone con differenti *background* culturali, non solo quindi persone con un' impostazione e una formazione tecnico-scientifica, ma anche *tester* con una formazione umanistica o con nessuna formazione specifica, ma con la passione per i videogiochi. Vediamo la loro distribuzione demografica in figura 6.1

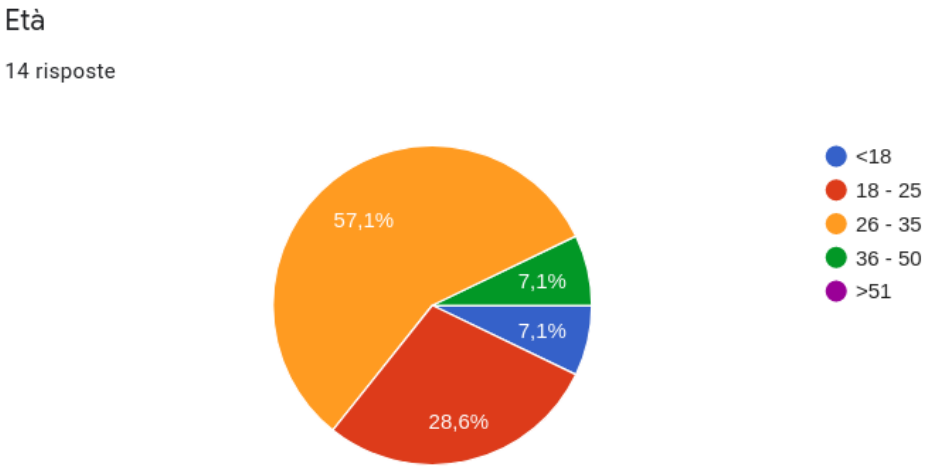

Figura 6.1: Struttura Demografica

6.1.1 Esperienze pregresse

Il questionario aveva diverse domande per la valutazione delle esperienze pregresse in ambito di creazione videoludica, sui ruoli svolti nei vari progetti e sulla conoscenza degli strumenti per la produzione di videgiochi, nel caso specifico Unity3D.
La prima domanda riguardava appunto se si avessero esperienze, professionali o non, in ambito di sviluppo di videogiochi.

Figura 6.2: Esperienze pregresse

Come possiamo vedere in figura 6.2 11 persone su 14 degli interrogati ha avuto esperienze professionali o amatoriali in ambito videoludico.

Dopo questa prima domanda sulle esperienze pregresse, si è sceso più in profondità, chiedendo a chi avesse risposto in modo affermativo, quali fossero stati i ruoli svolti nei vari progetti in cui avessero partecipato.

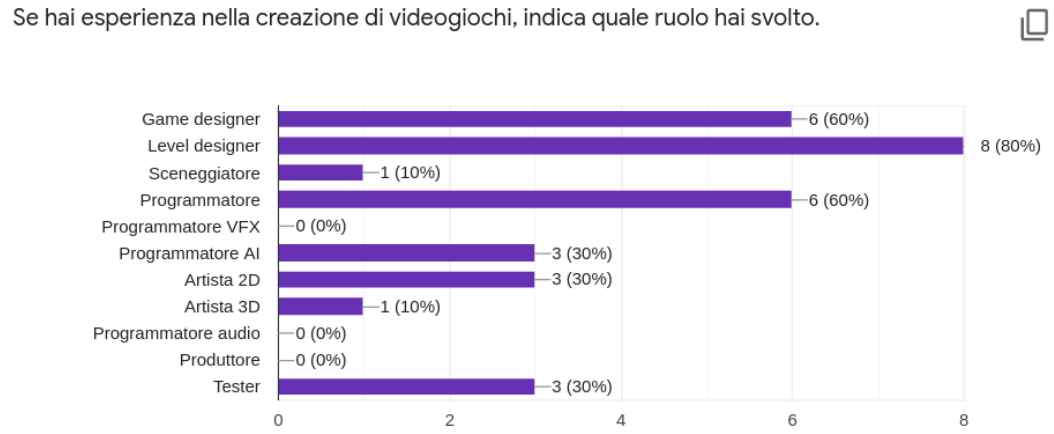

Figura 6.3: Distribuzione dei ruoli svolti

La maggior parte degli intervistati ha svolto il ruolo di *game designer*(6 su 14), *level designer*(8 su 14) o programmatore(6 su 14). Solamente una persona ha svolto il ruolo di sceneggiatore, con il compito di creare una struttura narrativa, una storia e un plot per il suo progetto.
Infine è stato chiesto se avessero mai avuto esperienze con Unity3D e se si in che termini temporali.

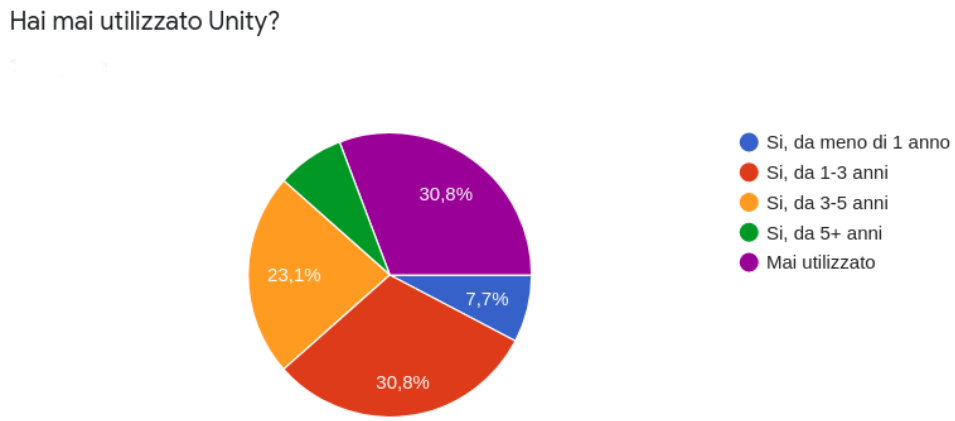

Figura 6.4: Utilizzo di Unity da parte degli intervistati

Come si evince dalla figura 6.4, 10 su 14 degli intervistati aveva già utilizzato Unity per i loro progetti, e la maggioranza per un periodo superiore ad un anno, segno che, l'engine è molto popolare e di facile utilizzo.

6.1.2 Abitudini videoludiche

Subito dopo le esperienze pregresse in ambito creativo, sono state chieste ai *tester* quali fossero le loro abitudini in campo videoludico.
La prima domanda è stata sul numero di ore che l'intervistato passa durante la settimana a giocare con qualsiasi tipo di videogioco.

Figura 6.5: Frequenza utilizzo dei videogiochi

Interessante notare, in figura 6.5, che nessuno ha risposto di non passare tempo davanti ad un gioco, tutti gli intervistati sono dei videogiocatori infatti, e la maggior parte di loro passa almeno più di 10 ore davanti ad un videogioco.

Subito dopo si è chiesto quali fossero i generi preferiti dagli intervistati

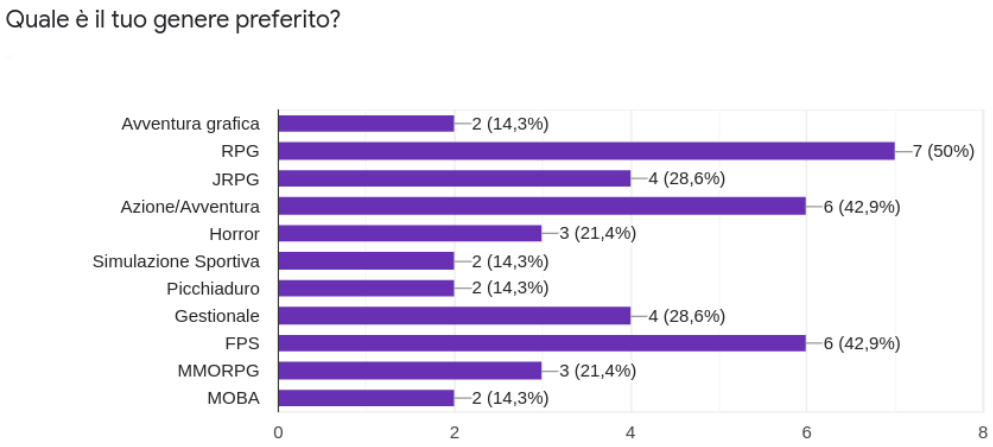

Figura 6.6: Preferenze di genere degli intervistati

Come possiamo vedere in figura 6.6 la maggior parte di loro ha risposto con RPG (7 su 14), Azione/avventura(6 su 14) o FPS (6 su 14), generi che ben si prestano ad una componente narrativa di impatto, come già spiegato in 2.2.

Infine è stato chiesto se avessero avuto esperienze in giochi di ruolo *pen & paper*.

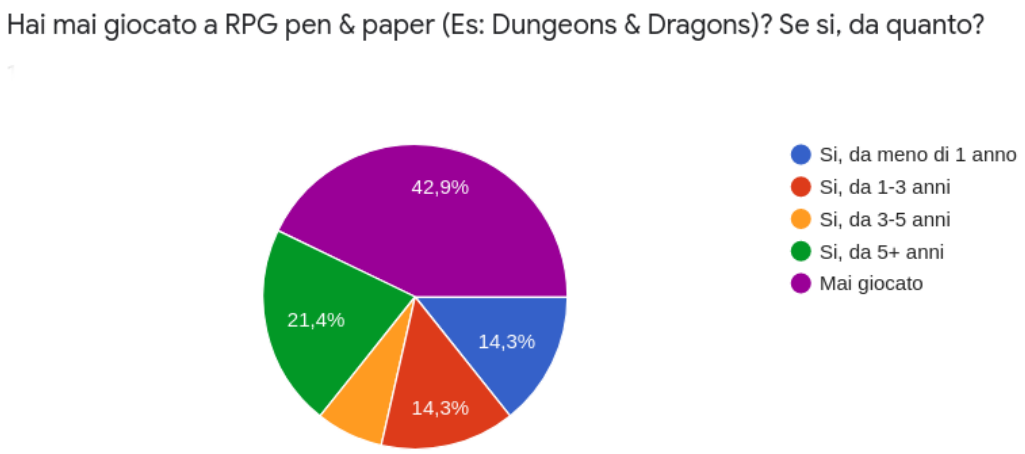

Figura 6.7: Distribuzione giochi di ruolo

Come si vede in figura 6.7 è quasi pari il numero di persone che hanno svolto questo genere di attività, 8 di loro, infatti, avevano preso parte a sessioni

di giochi di ruole, e 6, invece, non ne ha mai preso parte.
I giochi di questo tipo si prestano moltissimo alla creazione di storie, in quanto è il *game master* a dover creare una narrazione credibile e coinvolgente, per i giocatori che si cimentano in questo tipo di giochi.
La domanda seguente, quindi, è stata la conseguenza logica della precedente, ovvero in che ruolo avessero giocato.

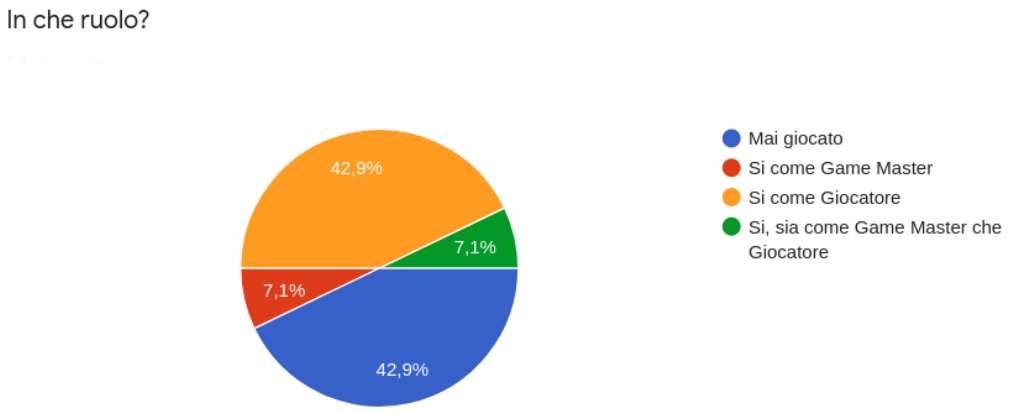

Figura 6.8: Ruoli nei giochi *pen & paper*

Come vediamo in figura 6.8 più della metà degli intervistati, 8, ha giocato come *game master* ovvero ha avuto esperienze, e un ruolo attivo, nella creazione di una struttura narrativa per un'attività ludica.

6.1.3 Conoscenza delle strutture narrative

Prima di passare alle valutazioni sul tool, è stato chiesto agli utenti se avessero una conoscenza pregressa delle strutture narrative o se avessero mai scritto delle storie, non per forza in ambito videoludico. Come possiamo vedere

Figura 6.9: Percentuale esperienze di scrittura

in figura 6.9 più della metà degli utenti aveva già scritto delle storie, quindi non erano del tutto nuovi all'argomento.
La domanda successiva è stata posta per capire l'effettiva conoscenza, in una scala da 1 a 5, dove 1 è nessuna conoscenza e 5 conoscenza totale, della struttura ad atti di una storia.

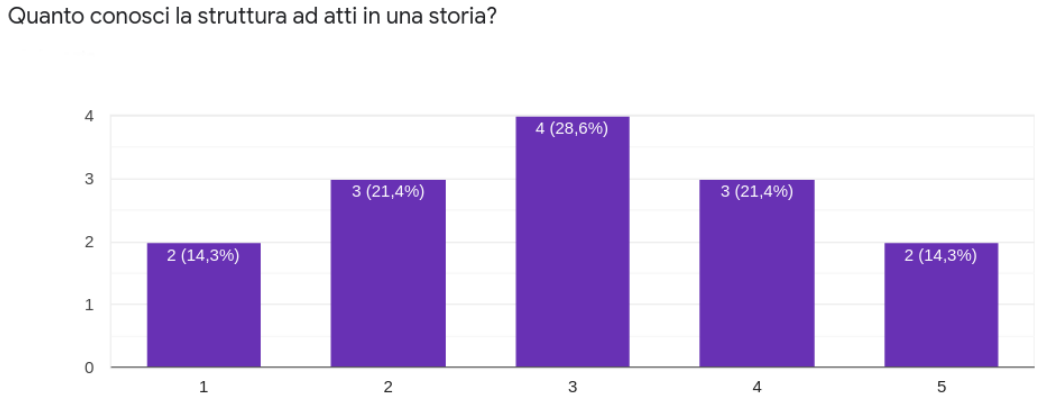

Figura 6.10: Conoscenza struttura ad atti

Come vediamo in figura 6.10, la distrubuzione della conoscenza di questo argomento è abbastanza omogenea, con la maggioranza che ha una conoscenza intermedia delle strutture narrative ad atti.
Infine è stato chiesto quanto fosse importante una buona componente narrativa in un videogioco, con una scala da 1 a 5, dove 1 era un totale disinteresse per questo aspetto e 5 il massimo interesse. Come vediamo in figura 6.11, per gli

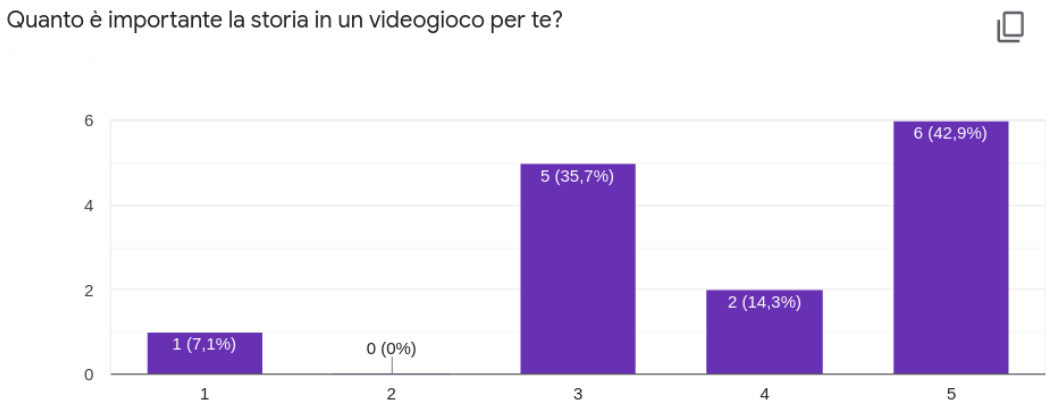

Figura 6.11: Importanza della storia secondo gli intervistati

intervistati la storia gioca un ruolo chiave nelle preferenze, con solamente un *tester* non interessato completamente alla componente narrativa.

6.1.4 Valutazione del tool

Infine sono state poste delle domande sulla soddisfazione complessiva e specifica del tool. In figura 6.12 viene mostrata una soddisfazione generale da

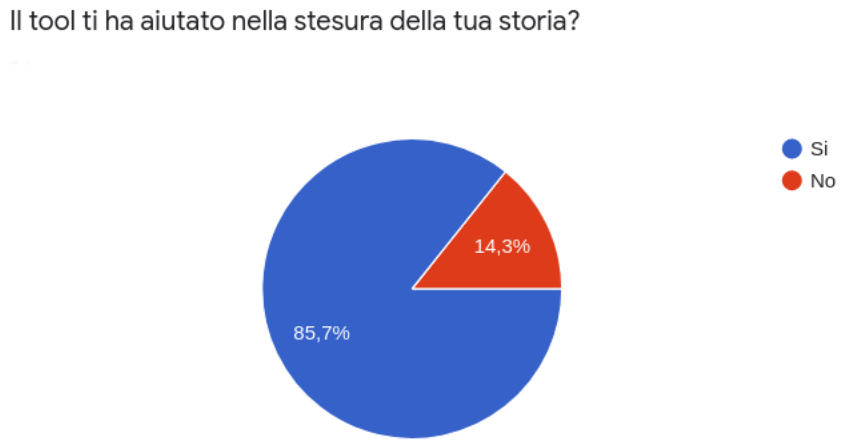

Figura 6.12: Soddisfazione generale

parte degli utenti per il tool nel suo complesso.
A questo punto, agli utenti è stato chiesto se, nel loro genere preferito, il progetto sarebbe stato utile In figura 6.13, sempre in una scala da 1 a 5, con 1

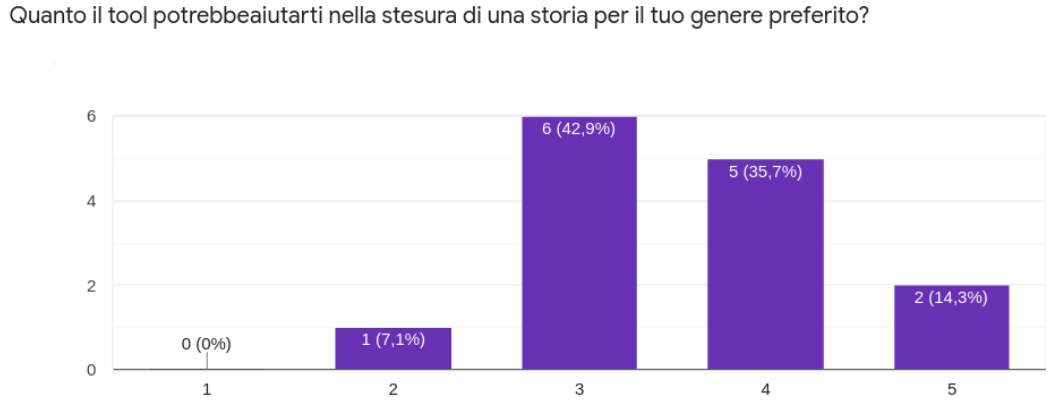

Figura 6.13: Utilità nel particolare genere

completamente in disaccordo e 5 completamente d'accordo, la maggioranza ha dimostrato una buona risposta nell'utilizzo del tool, in accordo con la figura precendere.

Infine sono state poste due domande più tecniche, sull'usabilità del tool e dei nodi in particolare:

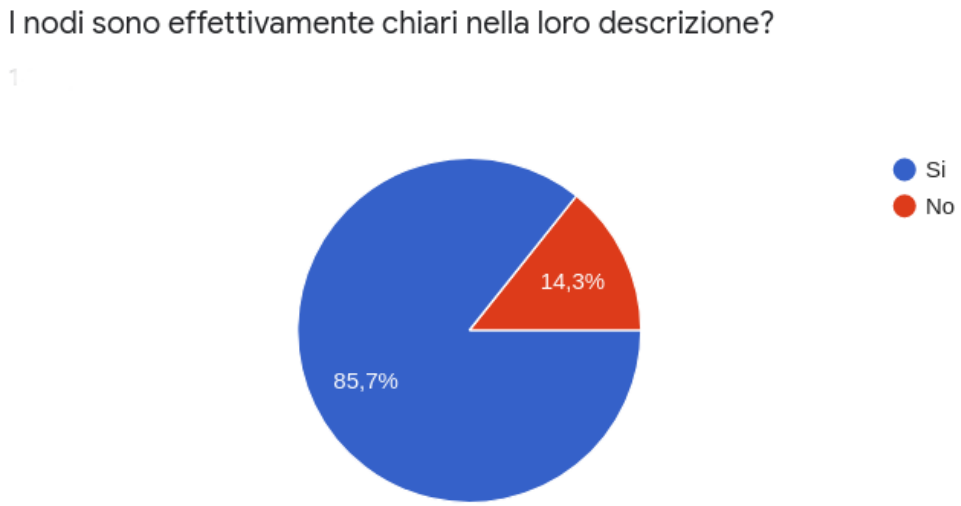

Figura 6.14: Soddisfazione sui nodi

In figura 6.14 vediamo una soddisfazione generale sulla chiarezza e l'esposizione dei nodi. E, come per i nodi, possiamo vedere in figura 6.15, un

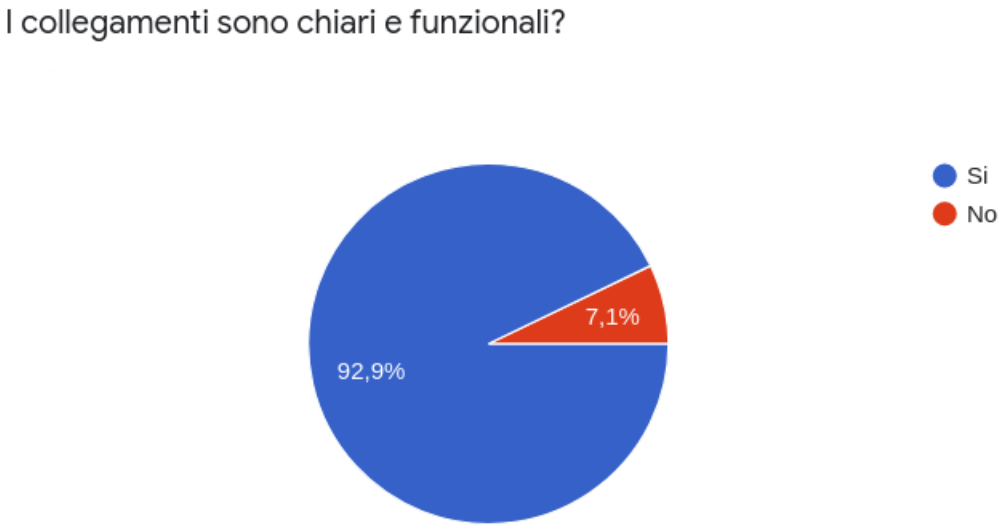

Figura 6.15: Soddisfazione sui collegamenti

ottimo risultato per quanto riguarda la soddisfazione sui collegamenti.

6.2 Sviluppi Futuri

La maggior parte delle criticità, che i tester hanno riferito durante la fase di test, risiede nella parte di *User experience*.
Infatti la maggior parte ha riscontrato e riferito, difficoltà nel navigare la canvas di Node editor, in quanto non vi è alcun riferimo visuale sulla posizione dei nodi, oltre vederli direttamente nello schermo (nessuna freccia che ne indichi la posizione, nessuna mappa che mostri dove trovare il nodo, nessuna ricerca testuale). Criticato, anche, il fatto che non si possa navigare la finestra con i tasti direzionali della tastiera, ma solo utilizzando il mouse.
Per migliorare l'esperienza dei buoni spunti potrebbero essere:

- Creazione di strutture narrative già preimpostate
- Riferimenti visuali nella canvas per la posizione dei nodi, che troppo spesso risultano lontani tra loro
- Crezione di *template* di personaggi basati su storie pre esistenti
- Ulteriori tooltip per la spiegazione e comprensione dei nodi, soprattutto per i nodi precedenti a Ghost 3.0,ad esempio per spiegare i vari archetipi inseriti, il loro ruolo e cosa comporta una scelta rispetto che un' altra nel nodo *Character descriptor*.

Restano valide le considerazioni fatte nei precedenti lavori di tesi, ovvero

- Creazione di una mini mappa dove mostrare la posizione dei nodi
- Integrazione con la scena di Unity3D
- Più possibilità di personalizzazione da parte dell'utente

6.3 Considerazioni Finali

La fase di test del progetto ha dato un buon riscontro, con pareri ampiamente positivi per quanto rigiarda il lavoro svolto.
Gli obiettivi prefissati, descritti in 1.3, ovvero

- Ampliamento della struttura a 3 atti, con l'introduzione di una struttura più moderna che potesse accettare più atti
- L'implemtenazione di un nodo completamente dedito alla *lore* del gioco, molto apprezzata dai tester, che hanno trovato il suo inserimento non fondamentale, ma sicuramente da un valore aggiunto al lavoro, in termini di completezza della struttura narrativa.
- Ampliamento e valorizzazione delle *sidequest* con l'implementazione del nodo *plot*

I margini di miglioramento sono ampi, soprattutto per quanto riguarda l'esperienza d'uso, ma le valutazioni mostrate in ambito di test sono decisamente positive.

Appendice A

Di seguito riportate alcune prove effettuate dai tester:

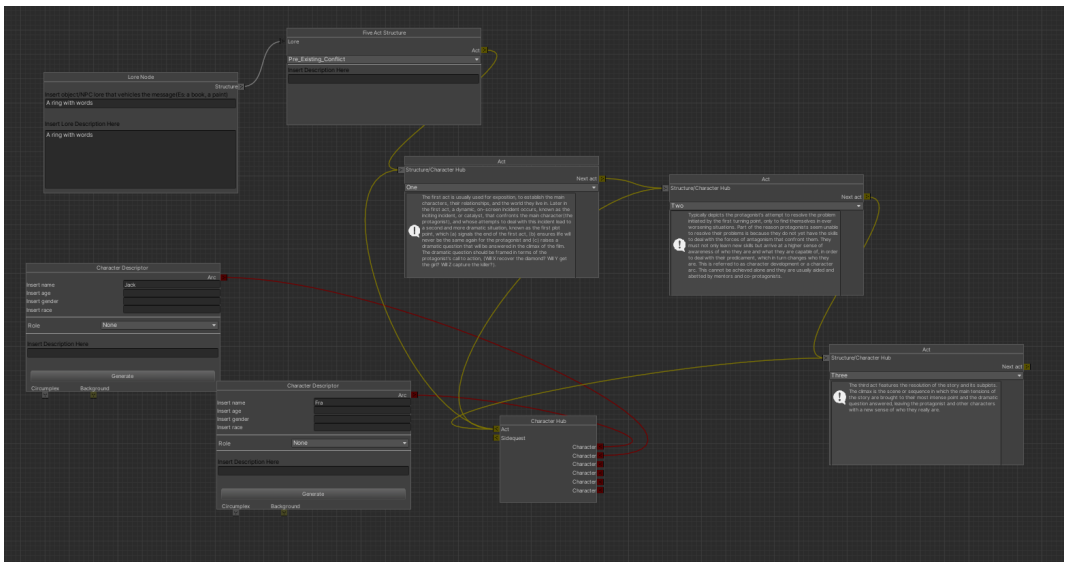

Figura A.1: Struttura a 5 atti con personaggi e lore

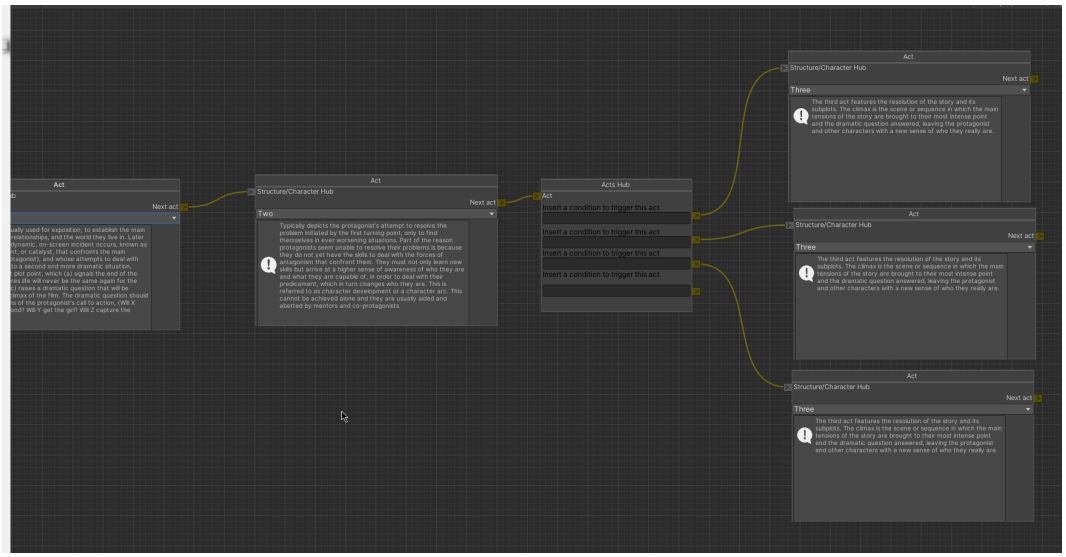

Figura A.2: Struttura a 3 atti, finali multipli

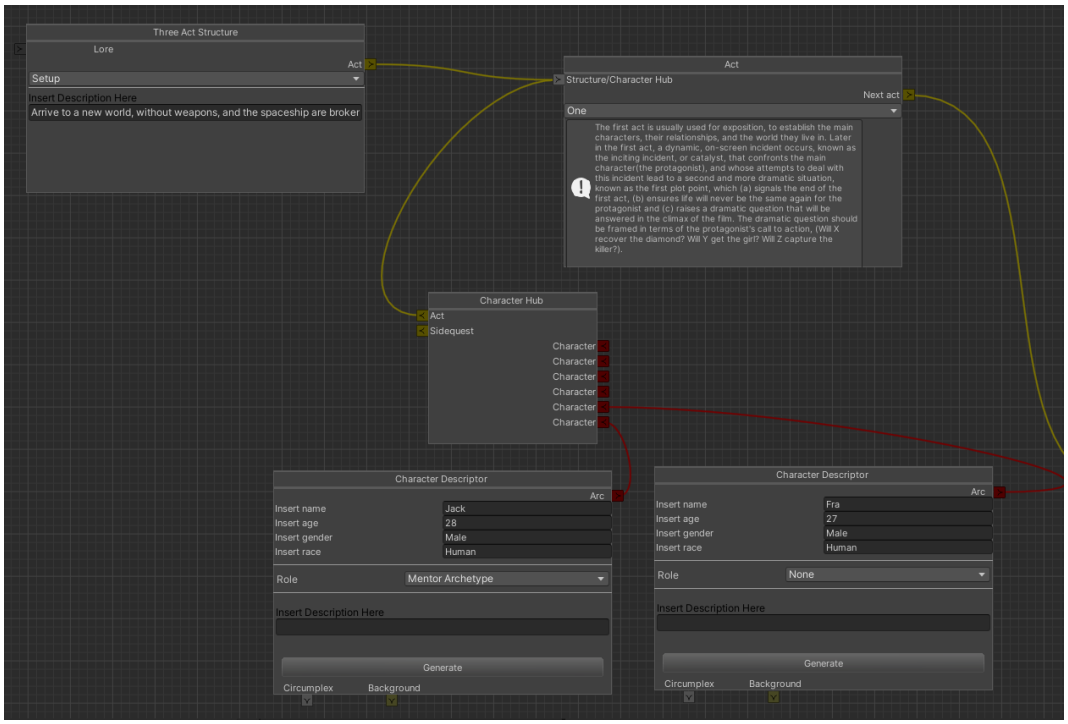

Figura A.3: Struttura a 3 atti con inserimento personaggi da Hub

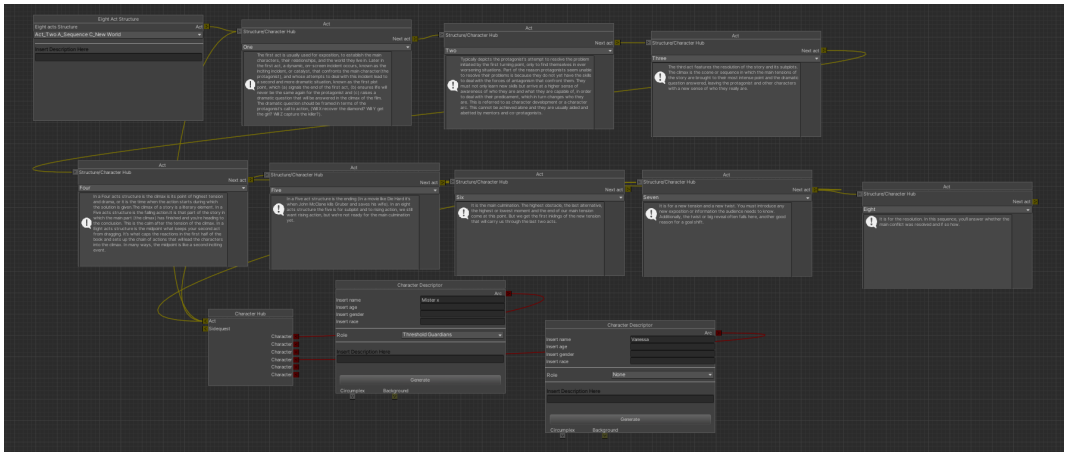

Figura A.4: Struttura a 8 atti con inserimento personaggi da Hub

Appendice B

B.0.1 Acts Hub Code

```
using UnityEngine;
using NodeEditorFramework;
using NodeEditorFramework.Utilities;
using UnityEditor;
[Node(false, "GHOST/Utility/ActsHub")]

public class ActsHub : Node
{
    public const string ID = "ghost_acts_hub";
    public override string GetID { get { return ID; } }
    GUIStyle guiStyle = new GUIStyle();
    public override string Title { get { return "Acts Hub"; } }

    public override Vector2 DefaultSize { get { return new Vector2(250, 220); } }

    public string text = "";
    public string condOne = "";
    public string condTwo = "";
    public string condThree = "";
    public string condFour = "";
```

```
[ValueConnectionKnob("Act", Direction.In, "ActsType", NodeSide.Left, 20)]
public ValueConnectionKnob structureConnection;
[ValueConnectionKnob("Character", Direction.Out, "ActsType",
NodeSide.Right, 60)]
public ValueConnectionKnob firstAct;
[ValueConnectionKnob("Character", Direction.Out, "ActsType",
NodeSide.Right, 100)]
public ValueConnectionKnob secondAct;
[ValueConnectionKnob("Character", Direction.Out, "ActsType",
NodeSide.Right, 140)]
public ValueConnectionKnob thirdAct;
[ValueConnectionKnob("Character", Direction.Out, "ActsType",
NodeSide.Right, 185)]
public ValueConnectionKnob fourthAct;

public override void NodeGUI()
{
    GUILayout.BeginHorizontal();
    GUILayout.BeginVertical();
    structureConnection.DisplayLayout();
    RTEditorGUI.PrefixLabel(new Rect(3, 20, 100, 50), new GUIContent
    ("Insert a condition to trigger this act"), new GUIStyle());
    condOne = EditorGUI.TextArea(new Rect(3, 35, size.x - 6, 20), text);
    RTEditorGUI.PrefixLabel(new Rect(3, 60, 100, 50), new GUIContent
    ("Insert a condition to trigger this act"), new GUIStyle());
    condTwo = EditorGUI.TextArea(new Rect(3, 75, size.x - 6, 20), text);
    RTEditorGUI.PrefixLabel(new Rect(3, 95, 100, 50), new GUIContent
    ("Insert a condition to trigger this act"), new GUIStyle());
    condTwo = EditorGUI.TextArea(new Rect(3, 110, size.x - 6, 20), text);
    RTEditorGUI.PrefixLabel(new Rect(3, 130, 100, 50), new GUIContent
```

```
        ("Insert a condition to trigger this act"), new GUIStyle());
        condTwo = EditorGUI.TextArea(new Rect(3, 155, size.x - 6, 20), text);
        GUILayout.EndVertical();
        GUILayout.EndHorizontal();
    }
}
```

B.0.2 Plot Code

```
using UnityEngine;
using NodeEditorFramework;
using NodeEditorFramework.Utilities;
using UnityEditor;

[Node(false, "GHOST/Story/SideQuest/Plot")]
public class Plot : Node
{
    public const string ID = "ghost_plot";
    public override string GetID { get { return ID; } }
    GUIStyle guiStyle = new GUIStyle();



    public enum PossiblePlot
    {
        LoveAndCourtship, MisfortunEnterprise, Revenge, Transgression,
        Mistery, None
    }
    public PossiblePlot lev = PossiblePlot.None;

    public enum LoveAndCourtship
    {
        APoorIsInLoveWithWealthyAndAristocraticB, AIsAJudgeAndBIsaFugitive, AADet
```

```
    AInLoveWithBDiscoversThatBIsInLoveWithB2, AXAGayYoungBlade, None
}
public LoveAndCourtship love = LoveAndCourtship.None;
public enum Enterprise { AAProfessionalManIsCapturedInHisOffice,
AWasInTheWorldWar, AAfterAMysteriousAbsenceOfManyYears, AABurglarSeeksToAidB,
AAfugitiveFromJustice, None }
public Enterprise misfortune = Enterprise.None;
public enum Revenge
{
    SeeksRevenge, BrideRevenge, MaleAgainstFemaleRival, Consiparacy
    , PoorYoungRevenge, None
}
public Revenge revenge = Revenge.None;
public enum Transgression
{
    FightTemptation, ForeignFugitive, SellHeirloom,HoundDog,Murder, None
}
public Transgression transgression = Transgression.None;
public enum Mistery {
    None
}
public Mistery mistery = Mistery.None;

public override string Title { get { return "Possible Plot"; } }
public override Vector2 DefaultSize { get { return new Vector2(400, 250); } }


[ValueConnectionKnob("Sidequest", Direction.In, "StructureType", ConnectionCo
public ValueConnectionKnob structureConnection;

public override void NodeGUI()
```

```
{
    GUILayout.BeginHorizontal();
    GUILayout.BeginVertical();
    structureConnection.DisplayLayout();
    lev = (PossiblePlot)UnityEditor.EditorGUILayout.
    EnumPopup(new GUIContent("", ""), lev);
    switch (lev)
    {
        case PossiblePlot.LoveAndCourtship:
            love = (LoveAndCourtship)UnityEditor.EditorGUILayout.
            EnumPopup(new GUIContent("", ""), love);
            EditorGUI.HelpBox(new Rect(3, 60, 390, 150),
            getLoveText(), MessageType.Info);
            break;
        case PossiblePlot.MisfortunEnterprise:
            misfortune = (Enterprise)UnityEditor.EditorGUILayout.
            EnumPopup(new GUIContent("", ""), misfortune);
            EditorGUI.HelpBox(new Rect(3, 60, 390, 150),
            getMisfortuneText(), MessageType.Info);
            break;
        case PossiblePlot.Revenge:
            revenge = (Revenge)UnityEditor.EditorGUILayout.
            EnumPopup(new GUIContent("", ""), revenge);
            EditorGUI.HelpBox(new Rect(3, 60, 390, 150),
            getRevengeText(), MessageType.Info);
            break;
        case PossiblePlot.Transgression:
            transgression = (Transgression)UnityEditor.
            EditorGUILayout.EnumPopup(new GUIContent("", ""), transgression);
            EditorGUI.HelpBox(new Rect(3, 60, 390, 150), getRevengeText(), Me
            break;
```

```
            case PossiblePlot.Mistery:
                mistery = (Mistery)UnityEditor.EditorGUILayout.
                EnumPopup(new GUIContent("", ""), mistery);
                EditorGUI.HelpBox(new Rect(3, 60, 390, 150),
                misteryText(), MessageType.Info);
                break;
        }

    }
    public string getLoveText()
    {
        string helpBox = "";
        switch (this.love)
        {
            case LoveAndCourtship.APoorIsInLoveWithWealthyAndAristocraticB:
                helpBox = "Male Protagonist, poor, is in love with wealthy
                and aristocratic female protagonist * A, pretends to be a man of
                break;
            case LoveAndCourtship.AIsAJudgeAndBIsaFugitive:
                helpBox = "Male protagonist is a judge, and Female Protagonist
                is a fugitive from justice posing as a woman of wealth andfashion
                break;
            case LoveAndCourtship.AADetectiveFallsInLoveWithB:
                helpBox = "Male protagonist, a detective, falls in love with
                female protagonist, the criminal he has arrested and is returning
                to the scene of her crime for trial and punishment";
                break;
            case LoveAndCourtship.AInLoveWithBDiscoversThatBIsInLoveWithB2:
                helpBox = "Male Protagonist, in love with Female Protagonist.
                Discovers that female protagonist is in love with a her female fr
                break;
```

```
            case LoveAndCourtship.AXAGayYoungBlade:
                helpBox = "A misterious male person, a gay young blade
                traveling through
                the country, takes refuge from a storm " +
                    "in a rural church.To his astonishment, he is hailed
                    at once as a bridegroom, and " +
                    "is hurried to the altar where a pretty girl, in an
                    exhausted condition, " +
                    "seems waiting for him. In a spirit of recklessness
                    , he allows himself " +
                    "to be married to her; and when she, after the ceremony
                    , seems to realize that " +
                    "he is not the man she thought he was, he hurriedly makes his
                break;
            case LoveAndCourtship.None:
                break;
        }
        return helpBox;
    }
    public string getMisfortuneText()
    {
        string helpBox = "";
        switch (this.misfortune)
        {
            case Enterprise.AAProfessionalManIsCapturedInHisOffice:
                helpBox = "Male protagonist, a professional man, is captured in
                his oflfice at night by three mysterious strangers, " +
                    "blindfolded and taken to a secret place. The male protagonis
                    spirited away by this three strangers,
                    is compelled to perfom a professional service";
                break;
```

```
case Enterprise.AWasInTheWorldWar:
    helpBox = "Male protagonist was in the World War
    Before the World War, was a successful business man; after the wa
    a physical wreck, and a bankrupt";
    break;
case Enterprise.AAfterAMysteriousAbsenceOfManyYears:
    helpBox = "Male, after a mysterious absence of many years,
    returns to his old home town." +
        "Returning as an Unknown to his native place,
        discovers that no one recognizes him";
    break;
case Enterprise.AABurglarSeeksToAidB:
    helpBox = "Male protagonist, a burglar, seeks to aid
    female protagonist, who was his friend before he
    'went to the bad'
    male protagonist," +
        "friend of female protagonist, breaks into a
        building for the
        purpose of committing a robbery, and" +
        "finds a trusted employee, male  criminal,
        female protagonist's
        husband, dead at his desk, a defaulter and a" +
        "suicide. Male criminal has left a note
        explaining his guilt,
        in order" +
        "to save his friend female protagonist from disgrace,
        destroys a letter that would have proved her" +
        "husband,male criminal, a defaulter and a suicide,
        'blows' a safe and pretends to have committed a robber";
    break;
case Enterprise.AAfugitiveFromJustice:
```

```
                helpBox = "Male protagonist, a fugitive from justice,
                disguises himself and, as an Unknown, risks discovery and
                arrest to carry out a romantic enterprise";
                break;
            case Enterprise.None:
                break;
        }
        return helpBox;
    }
    public string getRevengeText()
    {
        string helpBox = "";
        switch (this.revenge)
        {
            case Revenge.SeeksRevenge:
                helpBox = "Female protagonist seeks revenge as a lofty
                conception of duty|and comes to her death while seeking it";
                break;
            case Revenge.BrideRevenge:
                helpBox = "Female protagonist's husband, male protagonist,
                is killed by male criminal." +
                    "Male criminal through the law's delay and technicalities,
                    escapes with only a light sentence female protagonist invokes
                    "the Mosaic law in seeking revenge upon male criminal
                    for the murder of her husband.";
                break;
            case Revenge.MaleAgainstFemaleRival:
                helpBox = "Male protagonist, seeking revenge against female
                rival for a wrong committed by her husband, male rival, who " +
                    "is dead, finds that female rival treasures male rival's
                    memory most sacredly, unaware of his evil" +
```

```
                    " character. Male protagonist could destroy the beautiful
                    love and devotion of female rival, for her dead" +
                    " husband, male rival, by telling her the sort of man male
                    rival was male protagonist , in a spiritual victory, " +
                    "decides to forego a cherished enterprise and spare an innoce
                    woman her happy" +
                    "but mistaken ideals";
                break;
            case Revenge.Consiparacy:
                helpBox = "Male protagonist, high born, falls under
                the ban of death as a political conspirator in his native country
                break;
            case Revenge.PoorYoungRevenge:
                helpBox = "Male protagonist, a poor young man,
                inspired by anger and a desire for revenge, seeks to ruin " +
                    "wealthy male utility symbol, a powerful captain of
                    industry 1317 A commits an act of reprisal against male
                    utility sym with more serious results than he had" +
                    "intended";
                break;
            case Revenge.None:
                break;



        }
        return helpBox;
    }
    public string getTransgressionText() {
        string helpBox = "";
        switch (this.transgression) {
            case Transgression.FightTemptation:
```

```
                helpBox = "Male protagonist is a soldier, eager to
                fight but who is commanded to retreat before a superior force of
                break;
            case Transgression.ForeignFugitive:
                helpBox = "Male protagonist flees to a foreign country
                to escape the consequences of a transgression";
                break;
            case Transgression.SellHeirloom:
                helpBox = "Female protagonist, in order to carry
                out a certain enterprise, sells a valuable
                heirloom, object, confided to her for safe - keeping by male prot
                break;
            case Transgression.HoundDog:
                helpBox = "Male protagonist, of an inferior race,
                rescues female protagonist, of a superior race, from accident.";
                break;
            case Transgression.Murder:
                helpBox= "Female protagonist's friend, male friend,
                mysteriously disappears while in female protagonist's
                company so she gets arrested on suspicion of having
                murdered male friend";
                break;
            case Transgression.None:
                break;
        }
        return helpBox;
    }
    public string misteryText() {
        string helpbox = "";
        switch (this.mistery) { }
        return helpbox;
```

```
    }

}
```

B.0.3 Acts node Code

```
using UnityEngine;
using NodeEditorFramework;
using NodeEditorFramework.Utilities;
using UnityEditor;

[Node(false, "GHOST/Acts/Act")]
public class Act : Node
{
    public const string ID = "ghost_act";
    public override string GetID { get { return ID; } }
    GUIStyle guiStyle = new GUIStyle();



    public enum Level { One, Two, Three, Four, Five, Six, Seven, Eight, None }
    public Level lev = Level.None;

    public override string Title { get { return "Act"; } }
    public override Vector2 DefaultSize { get { return new Vector2(400, 250); } }



    [ValueConnectionKnob("Structure/Character Hub", Direction.In, "StructureType"
    public ValueConnectionKnob structureConnection;

    [ValueConnectionKnob("Next act", Direction.Out, "ActsType")]
    public ValueConnectionKnob actOutConnection;
    public override void NodeGUI()
```

```
{
    GUILayout.BeginHorizontal();
    GUILayout.BeginVertical();
    structureConnection.DisplayLayout();
    actOutConnection.DisplayLayout();
    RTEditorGUI.Seperator(new Rect(3, 48, size.x - 3, 1));
    EditorGUI.HelpBox(new Rect(3, 55, 350, 248), getHelpBox(), MessageType.In
    GUILayout.EndVertical();
    GUILayout.EndHorizontal();
    lev = (Level)UnityEditor.EditorGUILayout.EnumPopup(new GUIContent("", "")
}

public override bool Calculate()
{
    switch (lev)
    {
        case Level.One:
            actOutConnection.SetValue<string>("One");
            break;
        case Level.Two:
            actOutConnection.SetValue<string>("Two");
            break;
        case Level.Three:
            actOutConnection.SetValue<string>("Three");
            break;
        case Level.Four:
            actOutConnection.SetValue<string>("Four");
            break;
        case Level.Five:
            actOutConnection.SetValue<string>("Five");
            break;
```

```
            case Level.Six:
                actOutConnection.SetValue<string>("Six");
                break;
            case Level.Seven:
                actOutConnection.SetValue<string>("Seven");
                break;
            case Level.Eight:
                actOutConnection.SetValue<string>("Eight");
                break;
            case Level.None:
                actOutConnection.SetValue<string>("None");
                break;

        }

        return true;
    }

    public string getValue()
    {
        string ris = "";
        switch (this.lev)
        {

            case Level.One:
                ris = "One";
                break;
            case Level.Two:
                ris = "Two";
                break;
            case Level.Three:
```

```
                ris = "Three";
                break;
            case Level.Four:
                ris = "Four";
                break;
            case Level.Five:
                ris = "Five";
                break;
            case Level.Six:
                ris = "Six";
                break;
            case Level.Seven:
                ris = "Seven";
                break;
            case Level.Eight:
                ris = "Eight";
                break;
            case Level.None:
                ris = "One";
                break;

        }
        return ris;
    }
    public string getHelpBox()
    {
        string helpBox = "";
        switch (this.lev)
        {

            case Level.One:
```

```
            helpBox = "The first act is usually used for exposition,
            to establish the main characters,
            their relationships, and the world they live in.
            Later in the first act, a dynamic, on-screen incident occurs,
            known as the inciting incident, or catalyst,
            that confronts the main character(the protagonist),
            and whose attempts to deal with this incident
            lead to a second and more dramatic situation,
            known as the first plot point, which (a) signals
            the end of the first act, (b) ensures life will never
            be the same again for the protagonist and (c) raises
            a dramatic question that will be answered in the
            climax of the film. The dramatic question should
            be framed in terms of the protagonist's call to action,
            (Will X recover the diamond? Will Y get the girl? Will Z capture
            break;
        case Level.Two:
            helpBox = "Typically depicts the protagonist's
            attempt to resolve the problem initiated by the
            first turning point, only to find themselves in
            ever worsening situations. Part of the reason protagonists
            seem unable to resolve their problems is because they do
            not yet have the skills to deal with the forces of antagonism
            that confront them. They must not only learn new skills
            but arrive at a higher sense of awareness of who they
            are and what they are capable of, in order to deal
            with their predicament, which in turn changes who
            they are. This is referred to as character development
            or a character arc. This cannot be achieved alone and
            they are usually aided and abetted by mentors and co-protagonists
            break;
```

```
case Level.Three:
    helpBox = "The third act features the resolution
    of the story and its subplots. The climax is the
    scene or sequence in which the main tensions
    of the story are brought to their most intense
    point and the dramatic question answered,
    leaving the protagonist and other characters
    with a new sense of who they really are." +
        "";
    break;
case Level.Four:
    helpBox = "In a Four acts structure is the
    climax is its point of highest tension and
    drama, or it is the time when the action
    starts during which the solution is given.
    The climax of a story is a literary element." +
        " In a Five acts structure is the falling
        action.It is that part of the story in which
        the main part (the climax) has finished and
        you're heading to the conclusion.
        This is the calm after the tension of the climax. " +
        "In a Eight acts structure is the midpoint
        what keeps your second act from dragging.
        It's what caps the reactions in the first half
        of the book and sets up the chain of actions that
        will lead the characters into the climax.
        In many ways, the midpoint is like a second
        inciting event.";
    break;
case Level.Five:
    helpBox = "In a Five act structure is the ending
```

```
                (In a movie like Die Hard it's when John McClane
                kills Gruber and saves his wife). In an eight acts
                structure the five is for subplot and to rising action,
                we still want rising action, but we're not ready
                for the main culmination yet.";
                break;
            case Level.Six:
                helpBox = "It is the main culmination. The highest obstacle,
                the last alternative, the highest or lowest moment and
                the end of our main tension come at this point.
                But we get the first inklings of the new tension
                that will carry us through the last two acts.";
                break;
            case Level.Seven:
                helpBox = "It is for a new tension and a new twist
                . You must introduce any new exposition or information
                the audience needs to know. Additionally,
                the twist or big reveal often falls here, another
                good reason for a goal shift.";
                break;
            case Level.Eight:
                helpBox = "It is for the resolution.
                In this sequence, you'll answer whether
                the main conflict was resolved and if so how.";
                break;
            case Level.None:
                break;

        }
        return helpBox;
    }
```

```
}
```

B.0.4 Lore node Code

```
using UnityEngine;
using NodeEditorFramework;
using NodeEditorFramework.Utilities;
using UnityEditor;

[Node(false, "GHOST/Lore/Lore")]
public class LoreNode : Node
{
    public const string ID = "ghost_lorenode";
    public override string GetID { get { return ID; } }

    public override string Title { get { return "Lore Node"; } }
    public override Vector2 DefaultSize { get { return new Vector2(400, 250); } }
    public enum Messages { Collectionable, Paint, Book, Letter, Photo, None }
    public Messages messages = Messages.None;

    [ValueConnectionKnob("Structure", Direction.Out, "StructureType")]
    public ValueConnectionKnob firstLevelOutputKnob;


    public string text = "";
    public string loreVechicles = "";


    public override void NodeGUI()
    {
```

```
        GUILayout.BeginHorizontal();
        GUILayout.BeginVertical();

        GUILayout.EndVertical();
        GUILayout.BeginVertical();

        firstLevelOutputKnob.DisplayLayout();

        GUILayout.EndVertical();
        GUILayout.EndHorizontal();
        RTEditorGUI.PrefixLabel(new Rect(3, 20, 100, 50),
        new GUIContent("Insert object/NPC lore that vehicles the message(Es: a bo
        new GUIStyle());
        GUIStyle styleVehicles = GUI.skin.box;
        loreVechicles = EditorGUI.TextArea(new Rect(3, 35, size.x - 6, 20), text)

        RTEditorGUI.PrefixLabel(new Rect(3, 80, 100, 50),
        new GUIContent("Insert Lore Description Here"),
        new GUIStyle());

        GUIStyle style = GUI.skin.box;
        text = EditorGUI.TextArea(new Rect(3, 100, size.x - 6, 120), text);

        if (GUI.changed)
            NodeEditor.curNodeCanvas.OnNodeChange(this);
    }
}
```

Bibliografia

[1] Videogames da attività per nerd a industria:
https://www.ilfattoquotidiano.it/2020/11/22/videogames-da-attivita-per-nerd-a-industria-e-anche-litalia-conquista-la-sua-fetta-di-mercato/6007822/ [15/02/2021]

[2] Tomb Raider Official Site:
https://tombraider.square-enix-games.com/en-us [15/02/2021]

[3] Uncharted Official Site:
https://www.unchartedthegame.com/it-it/[15/02/2021]

[4] God of War official site:
https://godofwar.playstation.com/[15/02/2021]

[5] Naughty Dog Official site:
https://www.naughtydog.com/[15/02/2021]

[6] Starcraft official site:
https://starcraft.com/it-it/[15/02/2021]

[7] Command and Conquer official site
https://www.ea.com/it-it/games/command-and-conquer[15/02/2021]

[8] Robert Denton Bryant & Keith Giglio. *Slay the dragon writing great videogames*. Michael Wiese Productions ,2015

[9] Struttura Waterfall:
https://luigiatauro.com/2015/04/12/waterfall-incrementale-o-agile/[15/02/2021]

[10] Twine official site:
https://twinery.org/[15/02/2021]

[11] Xmind official site:
https://www.xmind.net/[15/02/2021]

[12] Fungus official site:
https://fungusgames.com/[15/02/2021]

[13] Scrivener overview:
https://www.literatureandlatte.com/scrivener/overview[15/02/2021]

[14] Final draft official site:
https://www.finaldraft.com/[15/02/2021]

[15] http://inform7.com/[11/02/2021]

[16] Unity Engine official site:
https://unity.com/[11/02/2021]

[17] Node Editor repository ad documentation:
https://github.com/Seneral/Node_Editor_Framework[12/02/2021]

[18] Cook W., Plotto: The Master Book of All Plots, Portland: Tin House Books, 2011

[19] The witcher official site:
https://thewitcher.com/it[15/02/2021]

[20] From software official site:
https://www.fromsoftware.jp/ww/[15/02/2021][15/02/2021]

[21] Detail page of Demon's souls from official site:
https://www.fromsoftware.jp/ww/detail.html?csm=070[15/02/2021]

[22] Detail page of Dark souls from official site:
https://www.fromsoftware.jp/ww/detail.html?csm=086[15/02/2021]

[23] Discord official site:
https://discord.com/[15/02/2021]

[24] Google Forms Documentation:
https://www.google.com/forms/about/[15/02/2021]

[25] Andrew Stockdale. ClueGen: An Exploration of Procedural Storytelling in the Format of Murder Mystery Games. *Experimental AI in Games: Papers from the AIIDE Workshop AAAI Technical Report WS-16-2293* :
https://ojs.aaai.org/index.php/AIIDE/article/view/12896/12744[15/02/2021]

[26] Marc Cavazza, Fred Charles. Dialogue Generation in Character-based Interactive Storytelling
https://www.aaai.org/Papers/AIIDE/2005/AIIDE05-004.pdf[15/02/2021]

[27] Unity's expansion detailed data:
https://unity.com/our-company[15/02/2021]

[28] Unity's data in VR and AR:
https://unity3d.com/unity/features/multiplatform/vr-ar#: :text=Unity[15/02/2021

[29] A. Guarneri, L. A. Ripamonti, F. Tissoni, M. Trubian, D. Maggiorini e D. Gadia, «GHOST: a GHOst STory-writer,» in Proceedings of ACM CHItaly '17, Cagliari, 2017

[30] R. Fine, «UnityScript's long ride off into the sunset,» 11 agosto 2017.
https://blogs.unity3d.com/2017/08/11/unityscripts-long-ride-off-into-the-sunset/. [16/02/2021].

[31] Aristotele, Poetica, Bompiani, 2000.

[32] T. Fullerton, Game Design Workshop: A Playcentric Approach to Creating Innovative Games, 3êdizione a cura di, CBC Press, 2014.

[33] J. Gregory, Game Engine Architecture, 2êdizione a cura di, A K Peters/CRC Press, 2014.

[34] D. Holmes, A Mind Forever Voyaging: A History of Storytelling in Video Games, CreateSpace Independent Publishing Platform, 2012.

[35] J.Juul, A Clash Between Game and Narrative, in Digital Arts and Culture, Bergen, Norway, 1998.

[36] D. Marks, L'arco di trasformazione del personaggio, Audino, 2007.