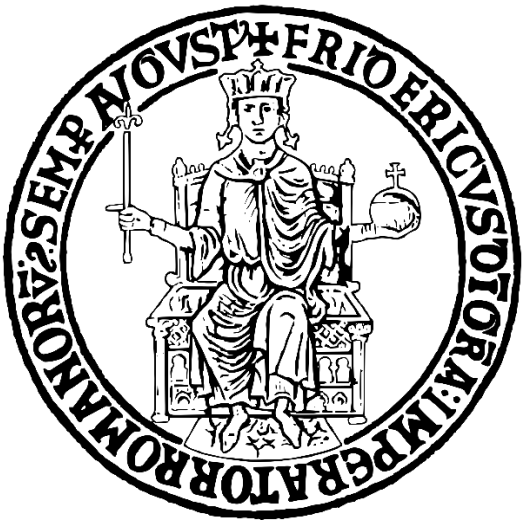
UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

DIPARTIMENTO DI INGEGNERIA ELETTRICA E DELLE TECNOLOGIE DELL'INFORMAZIONE



CORSO DI LAUREA MAGISTRALE IN INFORMATICA

GAME ENGINES AND INTERACTIVE EXPERIENCE

Deep-X-Pression: design document

Studente:
Danilo Esposito
Matricola:
N97000376

Docente:
Prof. Antonio Origlia

Anno Accademico 2022/23

Sommario

Bozza di progetto

Messaggio da comunicare

La depressione è un disturbo che influenza (non soltanto) la sfera emotiva della persona. Si può manifestare, tra gli altri, con sintomi:

- cognitivi (diminuzione della concentrazione, incapacità di prendere decisioni, etc.)
- affettivi (tristezza, malumori, disinteresse nelle attività quotidiane, etc.)
- comportamentali (ipersonnia o insonnia, rallentamento motorio, etc.)
- motivazionali (spossatezza, stanchezza costante, etc.)
- fisici (tachicardia, mal di testa, dolori muscolari e articolari, etc.)

Questa esperienza di gioco intende trasmettere al giocatore le sensazioni che prova una persona affetta da questi disturbi negli aspetti più comuni della quotidianità: la "serenità" della propria casa, l'ambiente lavorativo, i rapporti sociali.

Narrative design: show, don't tell

Per "show, don't tell" si intende un principio, ereditato dall'industria del cinema, mediante cui si vuole far comprendere situazioni dell'esperienza interattiva mostrando immagini anziché comunicarle chiaramente.
Per trasmettere al giocatore sensazioni legate alla depressione, come accennato nel precedente paragrafo, verrà utilizzato questo principio. Tutto verrà basato su un singolo parametro, l'umore del personaggio, che può influenzare:

- La scena di gioco, attraverso la manipolazione dei *materiali* di cui è costituita. Ad esempio: se il personaggio è di buonumore, la scena si mostrerà del suo colore reale; se il personaggio è di pessimo umore, la scena si mostrerà in bianco e nero.
- Il comportamento del personaggio, il quale può muoversi più o meno velocemente (o non muoversi del tutto, se si sente sfinito), può o meno saltare, e così via.
- Le motivazioni del personaggio, il quale può o meno svolgere le attività quotidiane (in ambito lavorativo: comunicare con i clienti o i colleghi; nel tempo libero: fare una passeggiata anziché guardare serie/film; nei rapporti sociali: rispondere agli amici che gli propongono di svolgere delle attività).
- I *suoni* che il personaggio avverte, che possono risultare ovattati nel caso in cui egli sia di cattivo umore.

Tuttavia, per alcuni elementi del videogioco, come i pensieri del personaggio o i dialoghi con gli altri personaggi, potrebbe essere necessario utilizzare del testo.

Narrative design: play, don't show

Il fulcro dell'esperienza interattiva è l'interazione tra il personaggio e ciò che lo circonda, principalmente strumenti lavorativi o per svago personale (telefono, computer, cellulare, TV, etc.) e gli altri personaggi della scena (colleghi, amici, etc.). Il giocatore può compiere varie azioni e prendere decisioni che influenzano l'umore del personaggio, ma vale anche il contrario: a seconda di questo "parametro", il personaggio sarà più o meno preposto a svolgere le attività quotidiane e a compiere determinate azioni.
Oltre alle attività, il giocatore può dialogare con altri personaggi, sia direttamente (in ufficio) sia indirettamente (a casa, mediante il cellulare).
Inoltre, cosa succede quando l'umore raggiunge il punto più basso? Si potrebbe pensare ad un game over immediato, oppure il giocatore – quando nessuno tra i suoi colleghi/amici lo fa – può motivare egli stesso il personaggio a continuare le sue attività, sebbene non possa farlo sempre e ripetutamente.

Stato interno del gioco

L'intero gameplay è basato su un parametro: l'umore del personaggio. Questo parametro influenza il gameplay, come le azioni che può eseguire il giocatore e le attività che può svolgere il personaggio, ma anche la scena di gioco: infatti, esso influenza la "vivacità" dei colori dei materiali; i colori saranno vivi se il grado d'umore è alto, altrimenti la scena sarà in bianco e nero: chiaramente, questo sarà graduale a seconda del valore di questo parametro, da gestire come float. Oltre al sottosistema dei materiali, l'umore influenza anche il modo in cui il personaggio avverte gli effetti sonori e la musica in

sottofondo: i suoni verranno riprodotti naturalmente se il grado d'umore è alto, altrimenti risulteranno ovattati al decrescere del valore di questo parametro.
Il personaggio principale può dialogare con gli altri personaggi: conseguentemente ad alcune azioni e alle scelte di dialogo effettuate, verrà assegnato – alla prima interazione con il personaggio X – e modificato – alle interazioni successive – un punteggio relativo al rapporto che il personaggio principale ha con tale personaggio, in una scala da 1 a 10: il giocatore dovrebbe mantenere sempre alto questo punteggio, ma avrà un certo grado di libertà nelle scelte. Il rapporto con i vari personaggi influisce sui dialoghi a cui si può prender parte.
Per la realizzazione dell'esperienza di gioco, verrà utilizzato uno dei seguenti template di Unreal Engine: *Prima persona*, se il giocatore deve immedesimarsi nella storia, giocando come se fosse egli stesso il personaggio, o *Terza persona*, per far sì che il giocatore si renda meglio conto delle azioni che compie il personaggio in base al suo umore.
Fondamentale per l'avanzamento sarà la gestione dell'input. Il personaggio può svolgere varie attività sia in ambito lavorativo sia nel tempo libero trascorso in casa, avvicinandosi ad uno dei vari oggetti con cui il giocatore può interagire alla pressione di un pulsante. Oltre a poter camminare, correre, saltare, e così via, può risultare necessario motivare il personaggio nel caso in cui non voglia svolgere le varie attività o perda la concentrazione per svolgerle, o colorare il mondo intorno a lui nel caso in cui l'umore scenda al valore più basso: per farlo, il giocatore dovrà premere ripetutamente un pulsante e, se il suo intervento va a buon fine, l'umore verrà impattato positivamente. Sarà importante anche l'input per la scelta nei dialoghi con gli altri personaggi.

Riepilogo della struttura del videogioco

- **Template di UE:** Prima persona / Terza persona
- **Personaggio principale:** Uomo in stato depressivo
- **Personaggi secondari:** Amici, colleghi di lavoro
- **Livello/i:** Casa, Ufficio
- **Genere:** Simulatore di vita (stile The Sims)
- **Obiettivi di gioco:** Interagire con l'ambiente circostante, prendere decisioni
- **Conseguenze alle azioni intraprese:** Alterazione dell'umore del personaggio principale
- **Effetti dell'umore sulla scena e sul gameplay:** Modifica dei *materiali*, modifica dei *suoni*, predisposizione del personaggio a svolgere le attività quotidiane, alterazione motoria, comportamentale e cognitiva, possibilità di avere *dialoghi* con altri personaggi
- **Effetti delle scelte di dialogo:** modifica del rapporto tra i personaggi

Nota. Resta da valutare la realizzazione di alcuni elementi di questa esperienza interattiva:
- Quante attività e quali scelte sono disponibili per tali attività?
- Aggiungere al gameplay dei quick time event per mantenere alta la concentrazione – e quindi l'umore – del personaggio?
- ...

Per un'esperienza di questo tipo, saranno estremamente importanti gli aspetti del narrative design.

1. Narrative design: una introduzione

Deep-X-Pression (leggi "deep expression") è un videogioco simulatore di vita di una persona che convive con la depressione, Emil. Questa esperienza interattiva intende trasmettere al giocatore le sensazioni che prova una persona affetta da questi disturbi: la storia è ambientata in un ufficio in cui Emil, oltre a svolgere attività prettamente lavorative, interagisce con i suoi colleghi per cercare di migliorare i rapporti con essi, aspetto che maggiormente influisce sul suo stato d'animo.

1.1 Elementi chiave dell'esperienza

Per "**show, don't tell**" si intende un principio, ereditato dall'industria del cinema, mediante cui si vuole far comprendere situazioni dell'esperienza interattiva mostrando immagini anziché comunicarle chiaramente. Per far sì che il giocatore empatizzi con Emil, si presenteranno nel corso dell'esperienza diverse situazioni visive che muteranno in base allo stato d'animo del personaggio: la scena di gioco, il comportamento del personaggio, le sue motivazioni, i suoni udibili, saranno tutti influenzati da questo aspetto, ed i cambiamenti saranno visibili al giocatore. Tuttavia, per alcuni elementi del videogioco, come i pensieri del personaggio o i dialoghi con gli altri personaggi, sarà necessario utilizzare del testo.

"**Play, don't show**" è un'espressione, derivata dalla precedente, che racchiude l'essenza del videogioco: sebbene ultimamente siano sempre più presenti cinematiche e sequenze narrative all'interno di questi, la componente interattiva è quella che lo ha sempre contraddistinto. Il fulcro di questa esperienza interattiva è l'interazione tra il personaggio e ciò che lo circonda, ovvero i più comuni oggetti che si trovano negli ambienti lavorativi (scrivania, WC, finestre, distributori, etc.) e gli altri personaggi della scena (colleghi). Il giocatore potrà compiere varie azioni e prendere decisioni che influenzano l'umore del personaggio, ma vale anche il contrario: a seconda di questo "parametro", il personaggio sarà più o meno preposto a svolgere le attività quotidiane e a compiere determinate azioni. Oltre alle attività, il giocatore potrà dialogare con altri personaggi: in base alle scelte effettuate e alle azioni che si compiono durante la giornata, il rapporto con gli altri personaggi potrà migliorare o peggiorare. ~~Quando l'umore del personaggio raggiungerà il punto più basso, per evitare un game over immediato, il giocatore – quando nessuno tra i suoi colleghi/amici lo fa – può motivare egli stesso il personaggio a continuare le sue attività, sebbene non possa farlo sempre e ripetutamente.~~

1.2 Motivazione degli elementi chiave e dettagli implementativi

Gli elementi appena introdotti sono stati scelti per la loro naturale predisposizione al narrative design: il giocatore dovrà prendere decisioni ragionate nelle attività e nei dialoghi, dal momento in cui queste influenzano l'umore del personaggio e il rapporto con gli altri personaggi, estremamente importante anche dal punto di vista implementativo. Infatti, come accennato nel precedente paragrafo, l'umore ha effetti su:

- La scena di gioco, attraverso la manipolazione dei *materiali* di cui è costituita. Ad esempio: se il personaggio è di buonumore, la scena si mostrerà del suo colore reale; se il personaggio è di pessimo umore, la scena si mostrerà in bianco e nero.
- Il *comportamento* del personaggio, il quale può muoversi più o meno velocemente (o non muoversi del tutto, se si sente sfinito), ad esempio.
- Le *motivazioni* del personaggio, il quale può o meno svolgere le attività quotidiane.
- I *suoni* che il personaggio avverte, che possono risultare ovattati nel caso in cui egli sia di cattivo umore.

Il videogioco verrà sviluppato utilizzando **UnrealEngine**. L'idea implementativa è utilizzare semplicemente il parametro dell'umore, proprio del personaggio, ed integrarlo con i sottosistemi dei Material, dei Sound ed eventualmente dell'AI per gestire le componenti visive, acustiche e di interazione con i personaggi dell'esperienza, rispettivamente. Mentre l'umore è proprio del personaggio principale, il rapporto con i personaggi secondari è gestito su questi ultimi, per cui X.relationship indica il rapporto tra Emil e il personaggio X.

1.3 Approccio allo storytelling digitale

La narrativa si svolge in un ufficio lavorativo, di cui Emil, il personaggio principale, è il responsabile, sebbene esegua comunque le stesse mansioni degli altri dipendenti. In ufficio sono disponibili diverse attività a cui Emil potrà prender parte semplicemente interagendovi: un distributore automatico di caffè, un divano con TV per rilassarsi durante l'ora di

spacco, una piccola stanza con servizi igienici, le scrivanie dei suoi colleghi per dialogarvi e la sua scrivania per il lavoro. Emil potrà interagire con gli elementi secondari dell'ambiente, come la finestra, l'orologio e la porta, per avere informazioni aggiuntive sul suo stato d'animo e per attivare varie meccaniche di gioco.

Nota. In base ad uno studio di fattibilità, verrà implementata o meno una sezione in cui Emil si trova nel proprio appartamento.

1.4 Deep-X-Pression: l'ambientazione

Per curare la narrativa del videogioco verrà utilizzato il software **articy:draft 3**. L'esperienza prende luogo ai giorni nostri in un ufficio lavorativo, di cui Emil, il personaggio principale, è il responsabile. L'ufficio è spazioso abbastanza da contenere:

- Cinque scrivanie; viste dall'entrata dell'ufficio, sulla sinistra:
  - in fondo, al centro, c'è quella di Emil;
  - immediatamente prima, a destra c'è la scrivania di Laura e a sinistra c'è la scrivania di Kevin;
  - le prime due visibili sono a destra quella di Jennifer e a sinistra quella di Harry.
- Un divano con TV, posti di fronte all'entrata, da utilizzare durante l'ora di spacco.
- Un distributore di bevande, al muro alla destra del divano e subito sulla destra all'ingresso.
- ~~Una stampante personale, posta accanto alla scrivania di ogni dipendente, Emil compreso.~~

Segue una piantina di esempio:

Questo paragrafo verrà approfondito nel capitolo successivo.

2. Narrative design: l'ambientazione

2.1 Deep-X-Pression: la location

10001 Consulting, Piano 3, Ufficio 3. *10001 Consulting è un'azienda di consulenza informatica con sede a New York, fondata nel 2008: negli anni si è espansa al punto da acquistare un edificio a cinque piani, ognuno dei quali possiede dieci uffici per le attività lavorative dei suoi dipendenti. L'ufficio numero 3, posto al terzo piano dell'edificio, è dotato di cinque scrivanie con stampante e computer personali, servizi igienici, due distributori di bevande e un comodo divano con TV, nel caso in cui i dipendenti vogliano trascorrere la loro ora di spacco in ufficio. In questo ufficio svolgono le loro mansioni Emil, Harry, Jennifer, Kevin e Laura.*

2.2 Deep-X-Pression: personaggio principale

Emil. *Responsabile di 10001 Consulting, svolge le sue mansioni nell'ufficio 3 del terzo piano dell'edificio aziendale, sito a New York. Deve molto all'azienda, a cui è legato da oltre dieci anni, che l'ha aiutato a superare un momento difficile della sua vita. Anche per questo motivo sente molto la responsabilità del suo impiego: la sua dedizione è d'esempio per gli altri dipendenti... Ma i sorrisi, a volte, nascondono ben altro. Ha un ottimo rapporto con Jennifer, sua collega di vecchia data.*

- Età: 35.
- Sesso: uomo.
- Personalità: molto dedito al lavoro, lo considera un ricambio di tutto l'affetto mostratogli dall'azienda, che l'ha aiutato a superare un momento difficile della sua vita.
- Obiettivi: superare le sue debolezze, evitare che i suoi conflitti interiori impattino negativamente sull'efficienza lavorativa del team.
- Conflitti interiori: lotta contro la depressione.
- Abilità: problem solving, mantenere il team coeso.
- Debolezze: rapporti sociali, sensibilità.
- Stranezze / Abitudini: fare spesso visita ai servizi igienici, spendere un ventesimo del suo stipendio ai distributori.

2.3 Deep-X-Pression: personaggi secondari

Harry. *Dipendente di 10001 Consulting, svolge le sue mansioni nell'ufficio 3 del terzo piano dell'edificio aziendale, sito a New York. Neolaureato, ha da poco superato il suo primo anno di contratto con l'azienda, che gli è valso un rinnovo biennale per meriti. Talento precoce, ha particolari ambizioni.*

- Età: 24.
- Sesso: uomo.
- Personalità: apparentemente simpatico, talento precoce.
- Obiettivi: diventare responsabile di 10001 Consulting.
- Conflitti interiori: -
- Abilità: problem solving, mantenersi aggiornato con le innovazioni tecnologiche.
- Debolezze: interazione con il suo responsabile.
- Stranezze / Abitudini: confabulare.

Jennifer. *Dipendente di 10001 Consulting, svolge le sue mansioni nell'ufficio 3 del terzo piano dell'edificio aziendale, sito a New York. Non ha avuto le possibilità economiche di frequentare l'università, ma la sua forte personalità le ha consentito di superare appieno uno stage con la sua attuale azienda, con cui lavora da ormai otto anni. Si è iscritta all'università due anni fa, per riscattarsi di ciò che non ha potuto fare in passato. È collega di vecchia data di Emil, cui è particolarmente legata, avendolo aiutato direttamente a superare un momento difficile della sua vita.*

- Età: 28.
- Sesso: donna.
- Personalità: gentilezza estrema, talento maturato negli anni, empatia fuori dal comune.
- Obiettivi: terminare gli studi.
- Conflitti interiori: -

- Abilità: immedesimarsi nei problemi del team, farsi carico del lavoro non svolto dagli altri.
- Debolezze: -
- Stranezze / Abitudini: -

Kevin. *Dipendente di 10001 Consulting, svolge le sue mansioni nell'ufficio 3 del terzo piano dell'edificio aziendale, sito a New York. Professionista di lunga data, ha accettato l'impiego tre anni fa, dopo aver trascorso quasi dodici anni in un'altra azienda di consulenza. Sebbene abbia le capacità per farlo, non ha particolari ambizioni legate alla sua carriera. Inoltre, è un accanito videogiocatore.*

- Età: 42.
- Sesso: uomo.
- Personalità: pigro, appassionato di videogiochi
- Obiettivi: -
- Conflitti interiori: -
- Abilità: trovare la soluzione ideale nel minor tempo possibile.
- Debolezze: pigrizia.
- Stranezze / Abitudini: -

Laura. *Dipendente di 10001 Consulting, svolge le sue mansioni nell'ufficio 3 del terzo piano dell'edificio aziendale, sito a New York. Molto simile a Emil per sensibilità, ha stretto da subito un forte legame con Jennifer, che l'ha aiutata ad ambientarsi quando ha accettato l'impiego quattro anni or sono. Pur non essendo particolarmente capace di lavorare in singolo, forma una gran squadra con Kevin nelle riunioni con i clienti dell'azienda. Ammira le capacità lavorative di Harry.*

- Età: 31.
- Sesso: donna.
- Personalità: simpatica, buon rapporto con ogni membro del team.
- Obiettivi: -
- Conflitti interiori: -
- Abilità: tenere alto il morale dei clienti.
- Debolezze: sensibilità.
- Stranezze / Abitudini: -

2.4 Deep-X-Pression: oggetti interagibili

Clock. *Oggetto interagibile importante ai fini dell'esperienza. Orologio presente nell'ufficio 3, piano 3 dell'edificio in cui ha sede l'azienda 10001 Consulting. Emil lo può utilizzare per far trascorrere il tempo di un'ora, ma questo riduce di poco il suo umore.*

CouchAndTV. *Oggetto interagibile importante ai fini dell'esperienza. Divano con TV frontale usati dai dipendenti dell'ufficio 3, piano 3 dell'edificio in cui ha sede l'azienda 10001 Consulting. Emil e i suoi colleghi lo possono usare durante l'ora di spacco nel caso vogliano trascorrerla insieme per migliorare i loro rapporti.*

Dispenser. *Oggetto interagibile importante ai fini dell'esperienza. Distributore per bevande calde e fredde costruito dall'azienda BLEH (acronimo sconosciuto), aggiunto in seguito all'ammodernamento dell'edificio avvenuto nel 2020. Utilizzabile in qualsiasi momento della giornata, vede in Emil un assiduo frequentatore per mantenere attiva la concentrazione (Focus): Emil può aiutare i colleghi solo quando è concentrato, ma l'utilizzo dei distributori riduce il suo umore.*

EmilDesk. *Oggetto interagibile importante ai fini dell'esperienza. Scrivania di Emil, che lavora nell'ufficio 3, piano 3 dell'edificio in cui ha sede l'azienda 10001 Consulting. Essendo molto dedito al lavoro e di esempio per i colleghi, è il luogo dell'ufficio ove normalmente Emil trascorre più tempo. Emil ritiene il lavoro uno svago che lo fa sentire gradualmente meglio, per cui ha la sensazione che il tempo passi più rapidamente.*

ExitDoor. *Oggetto interagibile importante ai fini dell'esperienza. Porta di ingresso e di uscita dell'ufficio 3, piano 3 dell'edificio in cui ha sede l'azienda 10001 Consulting. Il livello inizia con Emil all'ingresso della porta, all'avvio della partita nella prima ora della giornata lavorativa; Emil può utilizzarla in qualsiasi ora della giornata per uscire, ma farlo prima che l'ottava ora lavorativa sia terminata ha dei risvolti negativi sul gruppo di lavoro e su di lui.*

WC. *Oggetto interagibile importante ai fini dell'esperienza. Water closet usato dai dipendenti dell'ufficio 3, piano 3 dell'edificio in cui ha sede l'azienda 10001 Consulting. Emil lo utilizza anche per riflettere degli errori commessi durante la giornata o per nascondere agli altri le sofferenze con cui convive.*

Window. *Oggetto interagibile importante ai fini dell'esperienza. Una semplice finestra dell'ufficio dell'ufficio 3, piano 3 dell'edificio in cui ha sede l'azienda 10001 Consulting. Posta in varie zone della stanza per mantenere l'ambiente soleggiato, Emil ama guardare fuori per prendere decisioni che ritiene importanti.*

Segue la piantina vista nel Paragrafo 1.4 aggiornata con tutte le entità introdotte in questo Capitolo:

2.5 Deep-X-Pression: variabili importanti

ClockVariables. Gruppo di variabili del Clock:

- **CanInteract**, bool (default value: true). *Indica se Emil può interagire con questa entità. Emil non può interagire con la stessa entità due volte nella stessa ora o, in condizioni particolari, giornata. Avanzando di un'ora, non tutte le entità sono nuovamente interagibili. All'utilizzo, diminuisce l'umore di Emil di 5.*
- **Hours**, integer (default value: 8)**.** *Ore dell'orologio: aumentano di 1 quando Minutes raggiunge i 60, oppure quando Emil interagisce con l'orologio. La giornata lavorativa inizia alle 8:00 di mattina e termina alle 17:00; dalle 12:00 alle 13:00 è prevista un'ora di spacco.*
- **Minutes**, integer (default value: 0). *Minuti dell'orologio: aumentano di 1 ogni 10 secondi di tempo reale, si resettano a 60.*

CouchAndTVVariables. Gruppo di variabili di CouchAndTV:

- **CanInteract**, bool (default value: true). *Indica se Emil può interagire con questa entità. Emil non può interagire con la stessa entità due volte nella stessa ora o, in condizioni particolari, giornata.*

DispenserVariables. Gruppo di variabili del Dispenser:

- **CanInteract**, bool (default value: true). *Indica se Emil può interagire con questa entità. Emil non può interagire con la stessa entità due volte nella stessa ora o, in condizioni particolari, giornata. All'utilizzo, Emil si concentra, consentendogli di aiutare i colleghi al lavoro; tuttavia, viene ridotto il suo umore di 25. Si può interagire con il dispenser una volta al giorno.*

EmilDeskVariables. Gruppo di variabili di EmilDesk:

- **CanInteract**, bool (default value: true). *Indica se Emil può interagire con questa entità. Emil può sempre interagire con la sua scrivania per lavorare. All'utilizzo, il tempo avanza più rapidamente (un secondo di tempo reale equivale a un minuto di tempo in-game), ed ogni ora di tempo in-game fa migliorare leggermente il morale di Emil (+5).*
- **WorkToDo**, string**.** *Lavoro da svolgere per gli altri colleghi. Contiene il nome del collega da aiutare: per completare il lavoro è necessaria un'ora in-game. Non si possono prendere in carico più lavori allo stesso tempo.*

EmilVariables. Gruppo di variabili di Emil:

- **Focus**, bool (default value: false)**.** *Concentrazione di Emil: se è true, egli è concentrato; non lo è, altrimenti. Emil può prendere in carico i lavori dei suoi colleghi solo quando è concentrato. La concentrazione persiste fino al termine della giornata.*
- **Mood**, integer/float (default value: 50)**.** *Umore di Emil, parametro che influisce sulle attività che egli può svolgere, e sulla percezione che Emil ha dell'ambiente circostante. Molti suggeriscono che rappresenti un grado di convivenza e capacità di relazionamento con sé stessi.*

ExitDoorVariables. Gruppo di variabili della ExitDoor:

- **CanInteract**, bool (default value: true). *Indica se Emil può interagire con questa entità. Emil non può interagire con la stessa entità due volte nella stessa ora o, in condizioni particolari, giornata.*
- **CanLeave**, bool (default value: false)**.** *Regola la possibilità di lasciare l'ufficio o meno: se true, Emil può lasciare l'ufficio senza penalità; se è false, Emil può lasciare l'ufficio con penalità in base al tempo rimanente. Diventa true quando trascorrono le otto ore lavorative. Le penalità riguardano i rapporti con i colleghi e il suo umore, secondo la seguente formula:* TimePenalty = (17 - ClockVariables.Hours) * 5 + (60 - ClockVariables.Minutes) * (5/60). *Interagendo con la porta, si può terminare la giornata ed iniziare quella successiva, oppure terminare il gioco.*
- **TimePenalty**, integer/float**.** La penalità sull'umore e sui rapporti con gli altri personaggi in base alla quantità di tempo di gioco rimanente.

HarryVariables. Gruppo di variabili di Harry:

- **CanConverse**, bool (default value: true)**.** *Regola la possibilità per Emil di avere un dialogo con Harry. ~~Quando è true, viene mostrato un punto esclamativo sul personaggio in questione.~~*
- **CanInteract**, bool (default value: true). *Indica se Emil può interagire con questa entità. Emil non può interagire con la stessa entità due volte nella stessa ora o, in condizioni particolari, giornata.*
- **DialogueSuspended**, bool (default value: false). *Indica se il dialogo è stato sospeso per riflettere alla finestra.*
- **MoodDifference**, integer/float (default value: 0)**.** *Indica la variazione dell'umore di Emil causata dall'interazione con questa entità. Se positivo, Emil ha giovato dall'interazione; altrimenti, Emil non ne ha giovato. Se MoodDifference è uguale a x, l'umore di Emil è cambiato di x. Range: [-100,+100].*
- **Relationship**, integer/float (default value: 25)**.** *Rappresenta il rapporto tra Emil e Harry.*
- **WorkToDo_Completed**, bool (default value: false)**.** *Indica se il lavoro da svolgere per conto di Harry è stato completato.*
- **WorkToDo**, bool (default value: false)**.** *Indica se è attivo un lavoro da svolgere per conto di Harry.*

JenniferVariables. Gruppo di variabili di Jennifer:

- **CanConverse**, bool (default value: true)**.** *Regola la possibilità per Emil di avere un dialogo con Jennifer. ~~Quando è true, viene mostrato un punto esclamativo sul personaggio in questione.~~*
- **CanInteract**, bool (default value: true). *Indica se Emil può interagire con questa entità. Emil non può interagire con la stessa entità due volte nella stessa ora o, in condizioni particolari, giornata.*
- **DialogueAboutPast**, bool (default value: false)**.** *Indica se è stato già attivato il dialogo con Jennifer sul passato di Emil.*
- **DialogueSuspended**, bool (default value: false)**.** *Indica se il dialogo è stato sospeso per riflettere alla finestra.*
- **MoodDifference**, integer/float (default value: 0)**.** *Indica la variazione dell'umore di Emil causata dall'interazione con questa entità. Se positivo, Emil ha giovato dall'interazione; altrimenti, Emil non ne ha giovato. Se MoodDifference è uguale a x, l'umore di Emil è cambiato di x. Range: [-100,+100].*

- **Relationship**, integer/float (default value: 100)**.** *Rappresenta il rapporto tra Emil e Jennifer.*
- **WorkToDo_Completed**, bool (default value: false)**.** *Indica se il lavoro da svolgere per conto di Jennifer è stato completato.*
- **WorkToDo**, bool (default value: false)**.** *Indica se è attivo un lavoro da svolgere per conto di Jennifer.*

KevinVariables. Gruppo di variabili di Kevin:

- **CanConverse**, bool (default value: true)**.** *Regola la possibilità per Emil di avere un dialogo con Kevin. ~~Quando è true, viene mostrato un punto esclamativo sul personaggio in questione.~~*
- **CanInteract**, bool (default value: true). *Indica se Emil può interagire con questa entità. Emil non può interagire con la stessa entità due volte nella stessa ora o, in condizioni particolari, giornata.*
- **DialogueAboutVideogames**, bool (default value: false)**.** *Indica se è stato già attivato il dialogo con Kevin sul suo videogioco preferito.*
- **DialogueSuspended**, bool (default value: false). *Indica se il dialogo è stato sospeso per riflettere alla finestra.*
- **MoodDifference**, integer/float (default value: 0)**.** *Indica la variazione dell'umore di Emil causata dall'interazione con questa entità. Se positivo, Emil ha giovato dall'interazione; altrimenti, Emil non ne ha giovato. Se MoodDifference è uguale a x, l'umore di Emil è cambiato di x. Range: [-100,+100].*
- **Relationship**, integer/float (default value: 50)**.** *Rappresenta il rapporto tra Emil e Kevin.*
- **WorkToDo_Completed**, bool (default value: false)**.** *Indica se il lavoro da svolgere per conto di Kevin è stato completato.*
- **WorkToDo**, bool (default value: false)**.** *Indica se è attivo un lavoro da svolgere per conto di Kevin.*

LauraVariables. Gruppo di variabili di Laura:

- **CanConverse**, bool (default value: true)**.** *Regola la possibilità per Emil di avere un dialogo con Laura. ~~Quando è true, viene mostrato un punto esclamativo sul personaggio in questione.~~*
- **CanInteract**, bool (default value: true). *Indica se Emil può interagire con questa entità. Emil non può interagire con la stessa entità due volte nella stessa ora o, in condizioni particolari, giornata.*
- **DialogueSuspended**, bool (default value: false). *Indica se il dialogo è stato sospeso per riflettere alla finestra.*
- **MoodDifference**, integer/float (default value: 0)**.** *Indica la variazione dell'umore di Emil causata dall'interazione con questa entità. Se positivo, Emil ha giovato dall'interazione; altrimenti, Emil non ne ha giovato. Se MoodDifference è uguale a x, l'umore di Emil è cambiato di x. Range: [-100,+100].*
- **Relationship**, integer/float (default value: 50)**.** *Rappresenta il rapporto tra Emil e Laura.*
- **WorkToDo_Completed**, bool (default value: false)**.** *Indica se il lavoro da svolgere per conto di Laura è stato completato.*
- **WorkToDo**, bool (default value: false)**.** *Indica se è attivo un lavoro da svolgere per conto di Laura.*

WCVariables. Gruppo di variabili di WC:

- **CanInteract**, bool (default value: true). *Indica se Emil può interagire con questa entità. Emil non può interagire con la stessa entità due volte nella stessa ora o, in condizioni particolari, giornata.*
- **RepeatableDialogues**, string array**.** *Insieme di dialoghi falliti per cui Emil può ritornare sui suoi passi.*

WindowVariables. Gruppo di variabili di Window:

- **CanInteract**, bool (default value: true). *Indica se Emil può interagire con questa entità. Emil non può interagire con la stessa entità due volte nella stessa ora o, in condizioni particolari, giornata.*
- **ImportantDialogue**, string**.** *In qualche dialogo con uno dei personaggi secondari, Emil può selezionare - se possibile - un'opzione per riflettere sulla decisione da prendere: questo attiverà, presso una delle finestre disponibili, un suggerimento sulla scelta migliore.*

3. Narrative design: story flowchart

Questo capitolo spiega dettagliatamente la narrativa di Deep-X-Pression, omettendo le caratteristiche del gameplay e dell'implementazione finale in UnrealEngine, aspetti posti nel Capitolo 4. Seguono alcune terminologie:

- "Diagramma di flusso": il diagramma che rappresenta la storia del videogioco con una rete di eventi, dialoghi e decisioni che contribuiscono alla sua creazione.
- "Frammento di flusso": un frammento del diagramma di flusso, che può rappresentare un punto dell'intreccio della storia, una missione, uno stato di gioco, etc. Viene utilizzato in Deep-X-Pression per gestire dei sottoflussi della storia indipendenti.
- "Dialogo": uno speciale frammento del diagramma di flusso che può contenere delle conversazioni. Questi oggetti non sono linee di dialogo, bensì contenitori di frammenti di dialogo.
- "Frammento di dialogo": linee di conversazione incluse in un dialogo.
- "Entità": le entità sono sostanzialmente tutti gli oggetti di gioco (personaggi, oggetti veri e propri, o oggetti astratti).

3.1 La struttura della storia

La storia di Deep-X-Pression prende vita in un ufficio lavorativo, di cui Emil è il responsabile. Al suo gruppo fanno parte anche Harry, Jennifer, Kevin e Laura, ognuno dei quali lavora nella propria postazione. Deep-X-Pression simula il classico giorno lavorativo di Emil: in qualsiasi momento, il giocatore può interagire con la porta per scegliere di passare al giorno successivo oppure per terminare la partita. In sostanza, l'esperienza è una reiterazione del giorno lavorativo. In questo paragrafo sono stati introdotti due frammenti di flusso della storia:

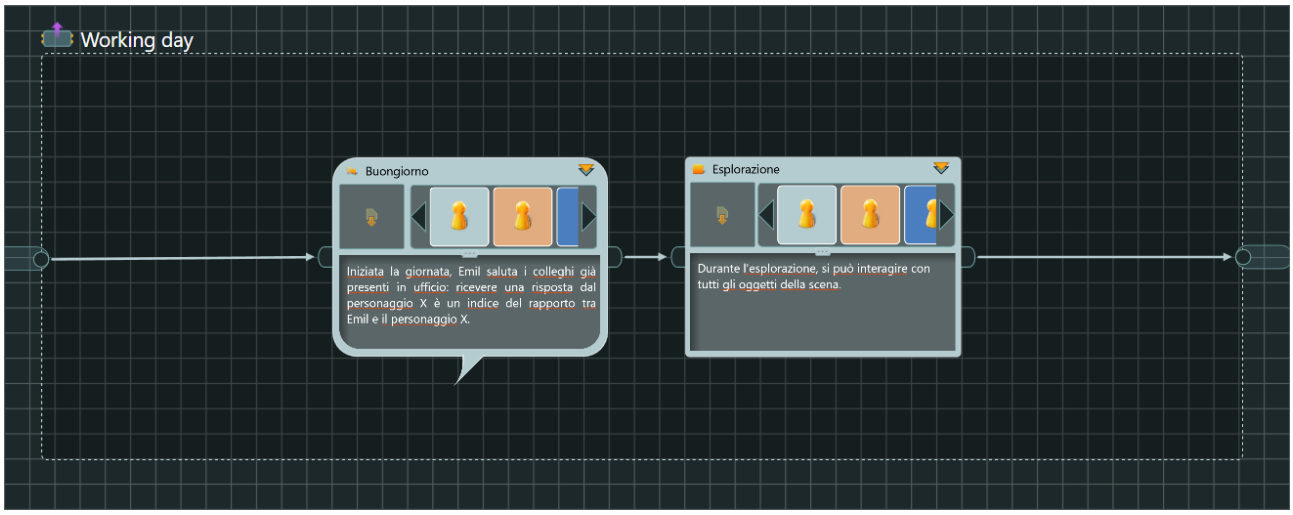
- **Working day.** "L'esperienza è una reiterazione dei giorni lavorativi di Emil: la giornata termina quando egli lascia l'ufficio e, dopo aver valutato eventuali cambiamenti nei rapporti con gli altri personaggi, inizia subito la giornata successiva, in cui vengono mantenuti i parametri di quella attuale. In sintesi, il videogioco ha la seguente struttura: Working day #1 → Working day #2 → Working day #3 → ..."
- **Summarising.** La fine dell'esperienza: dopo aver interagito con la porta, il giocatore ha selezionato l'opzione di terminare la partita. Vengono mostrati i risultati finali: umore di Emil, rapporti con i colleghi.

3.2 Il giorno lavorativo

La giornata inizia con un saluto ai colleghi; dopo questa breve interazione, può iniziare la fase di esplorazione:

- **Buongiorno.** "Iniziata la giornata, Emil saluta i colleghi già presenti in ufficio: ricevere una risposta dal personaggio X è un indice del rapporto tra Emil e il personaggio X."
- **Esplorazione.** "Durante l'esplorazione, si può interagire con tutti gli oggetti della scena."

Non ci si immergerà nel frammento di flusso del Buongiorno, in quanto è relativamente semplice: Emil saluta i colleghi e, se questi rispondono, il rapporto con essi è buono, altrimenti non lo è. Nel paragrafo 3.3 verrà analizzato, invece, il frammento di flusso dell'Esplorazione. Segue una figura che mostra la struttura del Working day.



3.3 L'esplorazione

Durante la fase di esplorazione, il giocatore può interagire con ogni oggetto della scena. Dopo ogni interazione, il giocatore è libero di interagire con qualsiasi altro personaggio entro la stessa ora: passata l'ora, ogni personaggio della scena torna ad essere interagibile; gli oggetti sono sempre interagibili, con qualche eccezione (ad esempio, non si può interagire con il dispenser più di una volta nella stessa giornata). Al centro della figura è posta l'interazione con la porta d'uscita, che è l'unico modo per passare al giorno successivo o per terminare la partita. Seguono tutte le interazioni:

- **Interazione con Clock.** Interagendo con l'orologio è possibile avanzare di un'ora istantaneamente, rendendo nuovamente disponibile l'interazione con i personaggi, al costo di un leggero peggioramento dell'umore di Emil.
- **Interazione con Couch&TV.** Interagendo con il divano, soltanto durante l'ora di spacco, Emil può scegliere di rilassarsi da solo o, se è in ottimi rapporti con tutti i suoi colleghi, di avere un dialogo riconciliante con essi, che massimizza il loro rapporto e l'umore di Emil.
- **Interazione con Dispenser.** Interagendo con il dispenser Emil si concentra: solo quando è concentrato, infatti, e se è dell'umore giusto, egli può prendere in carico il lavoro dei suoi colleghi, migliorando di molto il rapporto con essi; tuttavia, all'interazione viene ridotto di molto l'umore di Emil.
- **Interazione con Window.** Durante i dialoghi con i colleghi, Emil può scegliere, in certi casi, di sospendere il dialogo per rifletterci su: scelta questa opzione, diventa disponibile un momento di riflessione presso una delle finestre dell'ufficio, che consente di avere un suggerimento sulla scelta migliore da effettuare.
- **Interazione con WC.** Il water ha una doppia funzionalità: permette di ripetere l'ultimo dialogo avuto con uno dei suoi colleghi annullandone gli effetti (solo sul rapporto, non sull'umore di Emil), oppure per ripristinare l'umore di Emil se è estremamente basso.

- **Interazione con ExitDoor.** Interagendo con la porta d'uscita, si può passare direttamente al giorno successivo – calcolando le penalità nell'eventualità che l'orario lavorativo non sia ancora terminato – oppure terminare la partita, mostrando i risultati in Summarising.
- **Interazione con EmilDesk.** Interagendo con la scrivania, l'umore di Emil migliora con l'avanzare delle ore e, al contempo, se sono disponibili lavori per altri colleghi, Emil può completarli in un'ora. Mentre è seduto alla scrivania, l'utente può sospendere l'interazione alla pressione di qualsiasi tasto di movimento. Tuttavia, non può interagire con la scrivania se sono le 17:00.
- **Interazione con Harry.** Interagendo con Harry si può avviare un dialogo con quest'ultimo: dopo un saluto iniziale, se si riceve una risposta fredda allora il rapporto con questo personaggio non è dei migliori. Dopodiché, il giocatore può effettuare una delle seguenti scelte: offrire aiuto per il lavoro (+), invitarlo al distributore (+/-), allontanarsi, sospendere il dialogo. Il dialogo può migliorare il rapporto (+) o peggiorarlo (-).
- **Interazione con Jennifer.** Interagendo con Jennifer si può avviare un dialogo con quest'ultima: dopo un saluto iniziale, Jennifer noterà se Emil non è di buonumore, rincuorandolo. Dopodiché, il giocatore può effettuare una delle seguenti scelte: offrire aiuto per il lavoro (+), parlare del passato di Emil (+), allontanarsi. Il dialogo può migliorare il rapporto (+).
- **Interazione con Kevin.** Interagendo con Kevin si può avviare un dialogo con quest'ultimo: dopo un saluto iniziale, se si riceve una risposta fredda allora il rapporto con questo personaggio non è dei migliori. Dopodiché, il giocatore può effettuare una delle seguenti scelte: offrire aiuto per il lavoro (+), invitarlo al distributore (+), avviare un quiz sulla passione di Kevin per i videogiochi (+), allontanarsi, sospendere il dialogo. Il dialogo può migliorare il rapporto (+).
- **Interazione con Laura.** Interagendo con Laura si può avviare un dialogo con quest'ultima: dopo un saluto iniziale, Laura chiederà direttamente come sta Emil, segno della similarità tra i due. Dopodiché, il giocatore può effettuare una delle seguenti scelte: offrire aiuto per il lavoro (+), rimproverare Laura per la sua eccessiva affabilità (-), elogiare Laura per il suo lavoro (+) allontanarsi, sospendere il dialogo. Il dialogo può migliorare il rapporto (+) o peggiorarlo (-).

Quindi, l'unico modo per passare alla fase di Summarising è interagire con la porta e scegliere di terminare la partita. Qui, si mostrano i risultati della partita, in merito all'umore di Emil e al rapporto con i suoi colleghi.

3.4 Caratteristiche dello storytelling

Alla luce di quanto mostrato in questo capitolo, la storia di Deep-X-Pression non ha un "percorso" (**story path**) definito: interagisci con l'ambiente circostante e dialoga con i personaggi secondari per cambiare il rapporto con essi e l'umore di Emil (inteso come una sorta di rapporto con sé stesso); dalle decisioni prese, lo stato del gioco cambia a seconda di questi parametri, e idealmente si vuole invogliare il giocatore ad esplorare questi diversi stati. Come si può notare dalla fase di esplorazione, di cui c'è una figura esplicativa all'inizio del Paragrafo 3.3, il giocatore è libero di seguire il percorso che preferisce: non c'è un ordine preciso per l'avanzamento della storia. Sulla base del messaggio da veicolare, l'obiettivo è far comprendere al giocatore le sensazioni che prova una persona che convive con la depressione: il giocatore dovrebbe chiedersi "cosa succede se prendo la decisione 'bad' in vece della 'good'?". La presenza nel gioco di meccaniche che

1) Lo spronano continuamente a prendere le decisioni 'good' (riflettere alla finestra), e
2) Gli consentono di tornare sui propri passi per prendere una scelta diversa (usare il bagno)

dovrebbe alimentare nel giocatore questo senso di esplorazione: si **motiva estrinsecamente** il giocatore a prendere le decisioni buone, ma al contempo si alimenta la **curiosità** nel percorrere la scelta cattiva, dandogli addirittura lo strumento per farlo. Non è mai esistito, di conseguenza, uno story path critico mantenuto nella fase di progettazione.

Tuttavia, si può immaginare definito il grafico del **narrative arc** come segue, basato sull'umore di Emil e sul rapporto con gli altri personaggi:

1. All'inizio tutto risulta "normale", per cui si parte dal centro
2. Durante l'esplorazione, il giocatore può prendere decisioni per cambiare l'umore di Emil e i rapporti con i vari personaggi a suo piacimento, dunque mediamente si prosegue con una sorta di onda sinusoidale ad ampiezza variabile
3. Infine, siccome il gioco può terminare in qualsiasi stato, mediamente si termina al centro

Volendo centrare l'arco narrativo di Deep-X-Pression in uno dei sei "predefiniti", questo è una combinazione tra gli archi narrativi *Man in a hole* e *Icarus*, siccome il giocatore medio tende a scegliere il percorso estremamente buono o il percorso estremamente cattivo.

La storia del videogioco ruota intorno alle vicende lavorative di Emil: trattasi sostanzialmente di una reiterazione delle sue giornate al lavoro, sintomo anche della forte monotonia della vita di questo personaggio. Interagendo con l'ambiente

circostante e con gli altri personaggi della scena cambia il suo umore, specialmente nei dialoghi: per la sua forte sensibilità ed empatia, a seconda della scelta effettuata il suo umore migliora o peggiora. Ad ogni errore di Emil corrisponde una ripercussione sul suo umore, proprio perché le persone depresse sono estremamente sensibili ai propri errori, al punto da farle sentire inidonee alla vita sociale. Anche per curare quest'ultimo aspetto sono presenti nel videogioco le meccaniche di riflettere alla finestra ed usare il bagno, rispettivamente per: comprendere qual è la scelta migliore per Emil e per chi lo circonda, ed eventualmente tornare sui propri passi e rimediare agli errori (in realtà, si può anche ottenere l'effetto opposto, a seconda degli scopi prefissati dal giocatore – bad or good). Soltanto se Emil è di ottimo umore e se i rapporti con gli altri personaggi sono ottimi, viene sbloccato un dialogo durante l'ora di spacco in cui Emil si apre agli altri, che può essere visto come un finale positivo per la storia, in cui è evidente la sua **crescita** (intesa come la capacità del personaggio di far avanzare la storia, superando le difficoltà), sebbene non esista un vero e proprio finale. Inoltre, in un dialogo con un personaggio in particolare, si può scegliere di scoprire ulteriori dettagli sulla vita passata di Emil, legati al suo **sviluppo**. Quindi, in sintesi:

- La crescita di Emil viene evidenziata dalla sua capacità di socializzare con gli altri personaggi, aspetto che lo fa sentire di buonumore e che gli fa osservare il mondo che lo circonda in modo più vivo
- Lo sviluppo di Emil è caratterizzato dalla scoperta di fatti legati al suo passato che lo hanno reso così vulnerabile

I dialoghi consentono anche di rivelare circostanze sugli altri personaggi che ne fanno emergere il carattere.

4. Gameplay e sviluppo in UnrealEngine

Questo capitolo verrà scritto come una sorta di diario di sviluppo, spiegando le decisioni di implementazione impossibili da riprodurre in ArticyDraft ed eventuali deviazioni dalla progettazione originale. Per implementare i dialoghi verrà utilizzato il plugin open source **Dialogue System** (di Not Yet, gli sviluppatori che hanno realizzato il videogioco WarriOrb).

Nota. È stato fatto un tentativo di utilizzare il plugin Articy:draft Importer for Unreal, per importare direttamente nel progetto finale di Unreal Engine quanto realizzato per la fase di narrative design; tuttavia, sono state riscontrate difficoltà nell'importazione del contenuto di Articy:draft (non veniva generato alcun albero dello story flowchart), per cui è stato utilizzato DlgSystem.

Attenzione. Verranno lasciate delle print di debugging all'interno del gioco per far capire al tester i cambiamenti del game state che avvengono "dietro le quinte": il giocatore finale ovviamente non vedrà queste print, bensì dovrà capire i suddetti cambiamenti con l'ausilio di material e sound system oppure con i dialoghi presenti.

Deep-X-Pression è basato sull'interazione con i diversi oggetti e personaggi della scena. Il giocatore può muovere il personaggio (camminando di default, oppure correndo – se può – premendo Left Shift) liberamente durante la giornata lavorativa: avvicinandosi ad uno degli oggetti interagibili (orologi, divano, distributore, scrivania, porta di uscita, water, finestre) o ad uno dei personaggi secondari (Harry, Jennifer, Kevin, Laura), il giocatore può interagirvi alla pressione di un pulsante, e lo stato del gioco può evolvere in positivo o in negativo di conseguenza, condizionatamente alle azioni compiute dal giocatore. Seguono gli aspetti del gameplay condizionati dall'umore di Emil:

- **50 ≤ Umore di Emil ≤ 100**
    - Emil avverte un suono gradualmente più ovattato al decrescere dell'umore
    - La scena apparirà nel suo colore normale
    - Emil potrà correre o camminare liberamente
    - Tutte le azioni sono disponibili
- **26 ≤ Umore di Emil ≤ 49**
    - Emil avverte un suono gradualmente più ovattato al decrescere dell'umore
    - La scena apparirà gradualmente più grigia al decrescere dell'umore
    - Emil potrà correre o camminare liberamente
- **1 ≤ Umore di Emil ≤ 25**
    - Emil avverte un suono gradualmente più ovattato al decrescere dell'umore
    - La scena apparirà gradualmente più grigia al decrescere dell'umore
    - Emil potrà soltanto camminare
- **Umore di Emil = 0**
    - Il suono risulterà completamente ovattato, causando dei disturbi cognitivi: in questo stato, Emil non potrà conversare con i suoi colleghi
    - La scena sarà completamente in bianco e nero
    - Emil potrà soltanto camminare

Per aiutare Jennifer e Kevin (Harry e Laura non accetteranno il suo aiuto, ma i rapporti miglioreranno ugualmente), oltre ad essere dell'umore giusto (>25), Emil dev'essere concentrato, bevendo un drink al distributore (che, tuttavia, riduce il suo umore di 25). In sostanza, per aiutare i colleghi nei loro lavori – il modo ottimale per migliorare sia il rapporto con essi sia l'umore di Emil – bisogna avere un umore maggiore a 50, bere qualcosa al distributore per concentrarsi, e accettare gli incarichi dialogando con i colleghi.

Nel paragrafo 3.3 sono presenti le più importanti meccaniche di gioco, tutte mantenute nell'implementazione finale.

4.1 Gli oggetti di gioco in UnrealEngine: Blueprints interagibili

Gli oggetti di gioco, di cui si è parlato nei paragrafi 2.4 e 2.5, sono stati implementati nella versione finale del videogioco come Actor, quindi presenti e visibili nella scena di gioco. La maggior parte degli asset sono stati recuperati dal plugin Twinmotion Content, che include gratuitamente una quantità relativamente elevata di asset già pronti all'uso: ognuno degli asset utilizzati sono delle static mesh, che sono state successivamente convertite in Blueprint per gestire la logica di gioco. Non ci sono stati cambiamenti nella maggior parte delle variabili e della logica di questi oggetti, eccezion fatta per:

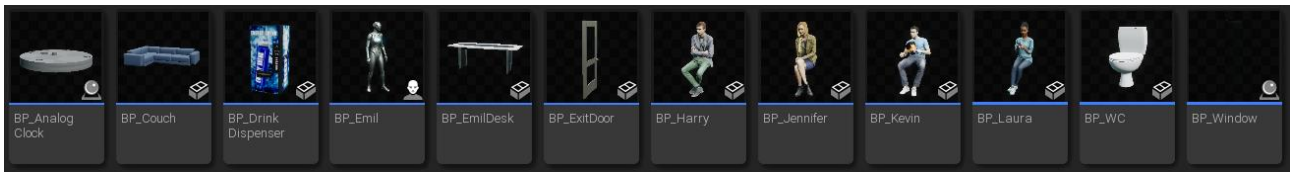
- **EmilDeskVariables.WorkToDo.** *Lavoro da svolgere per gli altri colleghi. Contiene il nome del collega da aiutare: per completare il lavoro è necessaria un'ora in-game. Non si possono prendere in carico più lavori allo*

stesso tempo. In fase di progettazione si è pensato di gestire tutti e quattro i possibili compiti con una stringa, poi soggetta a parsing all'interazione con la scrivania. Tuttavia, nell'implementazione finale sono state utilizzate, in sua vece, quattro variabili booleane, ognuna rappresentante un personaggio (WorkToDo_[Harry/Jennifer/Kevin/Laura]), per una gestione più efficiente e notevolmente semplificata.

- **WCVariables.RepeatableDialogues.** *Insieme di dialoghi falliti per cui Emil può ritornare sui suoi passi.* In fase di progettazione si è pensato di gestire tutti e quattro i possibili dialoghi ripetibili con un array di stringhe. Tuttavia, nell'implementazione finale sono state utilizzate, in sua vece, quattro variabili booleane, ognuna rappresentante un personaggio (RepeatableDialogues_[Harry/Jennifer/Kevin/Laura]), per una gestione più efficiente e notevolmente semplificata: si può ripetere soltanto l'ultimo dialogo avuto con il personaggio.
- **WindowVariables.ImportantDialogue.** *In qualche dialogo con uno dei personaggi secondari, Emil può selezionare - se possibile - un'opzione per riflettere sulla decisione da prendere: questo attiverà, presso una delle finestre disponibili, un suggerimento sulla scelta migliore.* In fase di progettazione si è pensato di gestire tutti e quattro i possibili dialoghi importanti con una stringa, poi soggetta a parsing all'interazione con la finestra. Tuttavia, nell'implementazione finale sono state utilizzate, in sua vece, quattro variabili booleane, ognuna rappresentante un personaggio (ImportantDialogue_[Harry/Jennifer/Kevin/Laura]), per una gestione più efficiente e notevolmente semplificata.

Oltre ai suddetti cambiamenti sulle variabili, è stata creata una classe Blueprint, denominata **BP_SecondaryCharacter**, sottoclasse di StaticMeshActor (quindi Actor e non Character, in quanto non è previsto che questi personaggi si muovano, per semplicità) che contiene tutte le variabili comuni a Harry, Jennifer, Kevin e Laura; tuttavia, siccome Jennifer e Kevin hanno una variabile in più rispetto agli altri, anche per ognuno di questi personaggi è stata creata una Blueprint ad-hoc (sebbene ne possa essere spawnato soltanto uno nel mondo di gioco), sottoclasse di BP_SecondaryCharacter.

Segue un'immagine che include tutte le Blueprint interagibili importanti presenti in Deep-X-Pression.



4.2 Gli oggetti di gioco in UnrealEngine: implementazione delle interazioni

Anzitutto, la logica dell'interazione è stata implementata come segue. È stata creata un'interfaccia, **`Interactable`**, che contiene un unico metodo, **`OnInteract()`**: tutti gli oggetti nella precedente immagine la implementano, e ognuno di essi implementa una propria logica in seguito all'interazione. Quando il personaggio entra nel range di collisione di uno di questi oggetti, comparirà a schermo il corrispondente prompt di interazione, contenente una breve descrizione dell'oggetto e della meccanica che gestisce. Per la maggior parte di queste blueprint, è stato deciso di ottenere un riferimento al player character (sicuramente di tipo BP_Emil, quindi possibile un cast) nell'evento BeginPlay e conservarlo in una variabile interna alle singole blueprint, per evitare di ripetere l'iter `GetPlayerController -> GetControlledPawn -> cast to BP_Emil` ogniqualvolta occorre accedere alle variabili di Emil.

4.2.1: BP_AnalogClock. *Interagendo con l'orologio è possibile avanzare di un'ora istantaneamente, rendendo nuovamente disponibile l'interazione con i personaggi, al costo di un leggero peggioramento dell'umore di Emil.*
In questa fase dell'implementazione è stato necessario far sì che ognuno dei due orologi avesse un riferimento interno all'altro orologio, per mantenerli sincronizzati: semplicemente, quando il giocatore interagisce con un orologio, viene invocato il metodo d'interazione di *questo* orologio e dell'*altro* orologio. Questa strategia di implementazione è necessaria per evitare di utilizzare le variabili del tempo sul player character, in quanto non è possibile utilizzarle sulla level blueprint e poi referenziarle all'interno delle blueprint dell'orologio. Chiaramente, la suddetta strategia non è l'ideale, dato che, se si volesse inserire un terzo orologio, bisognerebbe inserire un ulteriore riferimento interno; una soluzione robusta sarebbe creare un array di BP_AnalogClock (o utilizzare il nodo Get All Actors of Class), ma per le esigenze progettuali ciò è stato evitato per semplicità, in favore della soluzione con riferimento incrociato tra i due orologi.
Nota. Un piccolo dettaglio. È presente nel gioco il **ciclo temporale**: il sole cambia la sua posizione a seconda del tempo. Alle ore 8:00, il sole è perfettamente in alto al centro della scena (sì, come se fossero le 12:00...), mentre alle ore 17:00 il sole tramonta. Una volta capito il meccanismo, il giocatore sarà in grado per grandi linee di stabilire l'ora, anche senza guardare l'orologio: in particolare, saprà sicuramente quando è ora di lasciare l'ufficio, ovvero al tramonto.

4.2.2: BP_Couch. *Interagendo con il divano, soltanto durante l'ora di spacco, Emil può scegliere di rilassarsi da solo o, se è in ottimi rapporti con tutti i suoi colleghi, di avere un dialogo riconciliante con essi, che massimizza il loro rapporto e l'umore di Emil.*
Per implementare la funzione di rilassarsi da solo, con cui il giocatore può far sì che Emil trascorra l'ora di spacco sul divano (migliorando l'umore in base al tempo rimanente per le 13:00), è stato necessario modificare la logica del clock, in quanto l'avanzamento è gestito direttamente nella blueprint BP_AnalogClock: in particolare, è stata aggiunta una nuova variabile booleana nel clock, settata a true prima della chiamata a funzione dell'avanzamento, per avanzare fino alle ore 13:00 esatte anziché avanzare di un'ora.
In questa fase è stato implementato il sistema di dialogo. Per implementare il "dialogo importante", ovvero quello che coinvolge tutti i personaggi del gioco, è stato aggiunto un fade-in e un fade-out per nascondere il pop-up istantaneo dei personaggi sul divano e il conseguente spostamento alle loro postazioni al termine del dialogo.

4.2.3: BP_DrinkDispenser. *Interagendo con il dispenser Emil si concentra: solo quando è concentrato, infatti, e se è dell'umore giusto, egli può prendere in carico il lavoro dei suoi colleghi, migliorando di molto il rapporto con essi; tuttavia, all'interazione viene ridotto di molto l'umore di Emil.*
Implementazione semplice: all'interazione (possibile una volta al giorno), attiva la concentrazione di Emil e ne riduce l'umore di 25. Necessario conservare un riferimento all'altro distributore.

4.2.4: BP_EmilDesk. *Interagendo con la scrivania, l'umore di Emil migliora con l'avanzare delle ore e, al contempo, se sono disponibili lavori per altri colleghi, Emil può completarli in un'ora. Mentre è seduto alla scrivania, l'utente può sospendere l'interazione alla pressione di qualsiasi tasto di movimento. Tuttavia, non può interagire con la scrivania se sono le 17:00.*
Quando il personaggio interagisce con la scrivania, dopo un fade Emil sarà seduto e al lavoro, e il tempo sarà velocizzato (un secondo di tempo reale corrisponde ad un minuto di gioco: ogni minuto, l'umore di Emil aumenta di 5 punti); se interagisce con la scrivania mentre è al lavoro, dopo un fade Emil sarà nuovamente in piedi e potrà svolgere le restanti attività. Ogni ora di tempo in game seduto alla scrivania, se sono disponibili lavori da svolgere per i colleghi, questi verranno completati e ciò verrà notificato al giocatore: se sono disponibili x lavori su 4 (con $1 \leq x \leq 4$), saranno necessarie x ore in game per completarli tutti, quindi... Occhio all'orologio dietro di Emil!

4.2.5: BP_Harry. *Interagendo con Harry si può avviare un dialogo con quest'ultimo: dopo un saluto iniziale, se si riceve una risposta fredda allora il rapporto con questo personaggio non è dei migliori. Dopodiché, il giocatore può effettuare una delle seguenti scelte: offrire aiuto per il lavoro (+), invitarlo al distributore (+/-), allontanarsi, sospendere il dialogo. Il dialogo può migliorare il rapporto (+) o peggiorarlo (-).*
Tutto implementato secondo la progettazione in Articy:draft. Unico problema riguardante il sistema di dialoghi: apparentemente non riconosce le variabili floating point di alcune classi implementate, per cui è stato necessario in alcuni frangenti convertire da float a int per l'aggiornamento delle variabili in seguito alle scelte di dialogo.

4.2.6: BP_Jennifer. *Interagendo con Jennifer si può avviare un dialogo con quest'ultima: dopo un saluto iniziale, Jennifer noterà se Emil non è di buonumore, rincuorandolo. Dopodiché, il giocatore può effettuare una delle seguenti scelte: offrire aiuto per il lavoro (+), parlare del passato di Emil (+), allontanarsi. Il dialogo può migliorare il rapporto (+).*
Tutto implementato secondo la progettazione in Articy:draft. Unico problema riguardante il sistema di dialoghi: apparentemente non riconosce le variabili floating point di alcune classi implementate, per cui è stato necessario in alcuni frangenti convertire da float a int per l'aggiornamento delle variabili in seguito alle scelte di dialogo.

4.2.7: BP_Kevin. *Interagendo con Kevin si può avviare un dialogo con quest'ultimo: dopo un saluto iniziale, se si riceve una risposta fredda allora il rapporto con questo personaggio non è dei migliori. Dopodiché, il giocatore può effettuare una delle seguenti scelte: offrire aiuto per il lavoro (+), invitarlo al distributore (+), avviare un quiz sulla passione di Kevin per i videogiochi (+), allontanarsi, sospendere il dialogo. Il dialogo può migliorare il rapporto (+).*
Tutto implementato secondo la progettazione in Articy:draft. Unico problema riguardante il sistema di dialoghi: apparentemente non riconosce le variabili floating point di alcune classi implementate, per cui è stato necessario in alcuni frangenti convertire da float a int per l'aggiornamento delle variabili in seguito alle scelte di dialogo.

4.2.8: BP_Laura. *Interagendo con Laura si può avviare un dialogo con quest'ultima: dopo un saluto iniziale, Laura chiederà direttamente come sta Emil, segno della similarità tra i due. Dopodiché, il giocatore può effettuare una delle seguenti scelte: offrire aiuto per il lavoro (+), rimproverare Laura per la sua eccessiva affabilità (-), elogiare Laura per il suo lavoro (+) allontanarsi, sospendere il dialogo. Il dialogo può migliorare il rapporto (+) o peggiorarlo (-).*
Tutto implementato secondo la progettazione in Articy:draft. Unico problema riguardante il sistema di dialoghi: apparentemente non riconosce le variabili floating point di alcune classi implementate, per cui è stato necessario in alcuni frangenti convertire da float a int per l'aggiornamento delle variabili in seguito alle scelte di dialogo.

4.2.9: BP_Window. *Durante i dialoghi con i colleghi, Emil può scegliere, in certi casi, sospendere il dialogo per riflettere su: scelta questa opzione, diventa disponibile un momento di riflessione presso una delle finestre dell'ufficio, che consente di avere un suggerimento sulla scelta migliore da effettuare.*
Oltre ad aver implementato l'interazione, presso le finestre si può anche leggere una descrizione di ognuno dei personaggi.

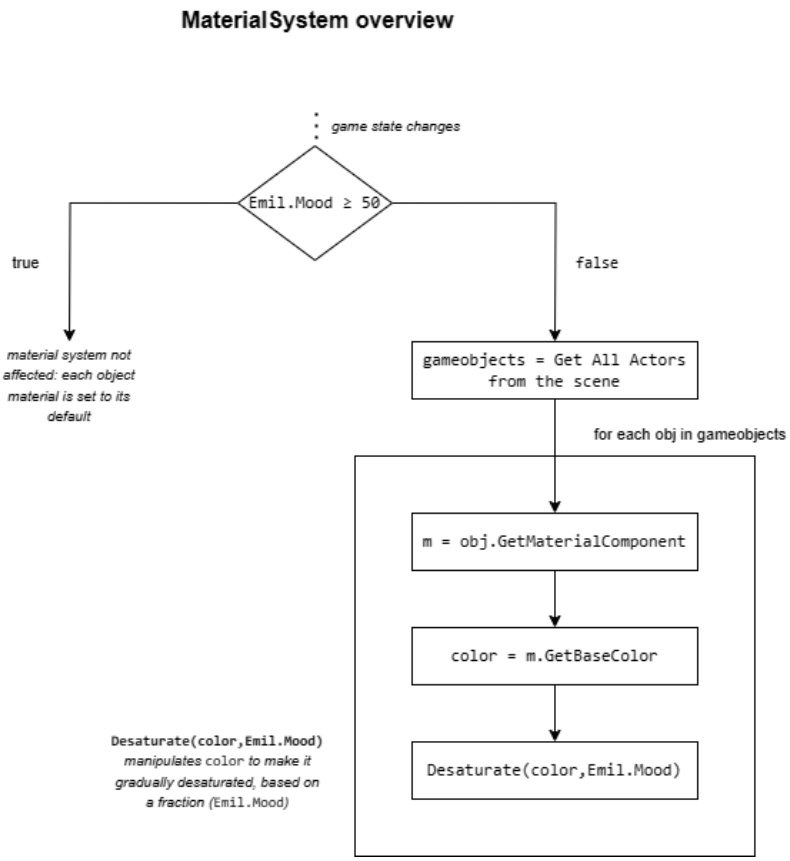
4.2.10: BP_WC. *Il water ha una doppia funzionalità: permette di ripetere l'ultimo dialogo avuto con uno dei suoi colleghi annullandone gli effetti (solo sul rapporto, non sull'umore di Emil), oppure per ripristinare l'umore di Emil se è estremamente basso.*
Oltre ad aver implementato l'interazione, presso i WC si possono anche consultare i parametri fondamentali del gioco: umore di Emil e rapporto con gli altri personaggi.

4.2.11: *Buongiorno.* *Iniziata la giornata, inizia un dialogo in cui Emil prima dà informazioni sul suo stato d'animo (Umore ≤ 25: 'Oggi sono di pessimo umore'; 25 < Umore ≤ 50: 'Oggi non sono di ottimo umore'; altrimenti: 'Oggi sono di buonumore'), poi saluta i colleghi già presenti in ufficio: ricevere una risposta dal personaggio X è indicatore di un buon rapporto tra Emil e il personaggio X (Rapporto > 50).*
Per avviare il dialogo a inizio partita, occorre attendere che tutti gli Actor siano spawnati nel livello: per questo, alla fine dell'evento BeginPlay del player character, è stato utilizzato un loop che ripete un controllo sulla callback GameModeBase.HasMatchStarted, che viene chiamata dopo l'invocazione degli eventi BeginPlay di tutti gli Actor.
Nota. Il loop è stato aggiunto per sicurezza, ma funziona anche senza, in quanto dai test effettuati la suddetta callback restituisce sempre true.

4.2.12: BP_ExitDoor. *Interagendo con la porta d'uscita, si può passare direttamente al giorno successivo – calcolando le penalità nell'eventualità che l'orario lavorativo non sia ancora terminato – oppure terminare la partita, mostrando i risultati in Summarising.*

4.2.13: *Summarising*. Nel dialogo che compare all'interazione con BP_ExitDoor, selezionando l'opzione "Termina partita" verranno mostrati i risultati finali dell'esperienza: giorni trascorsi, umore di Emil e rapporto con gli altri personaggi.

4.3 Material system



Come descritto all'inizio di questo capitolo, il sistema dei materiali è parte integrante delle meccaniche di gioco, per cui cambia al variare dello stato interno del gioco. In particolare, l'umore di Emil influenza i materiali di cui sono costituiti gli oggetti di gioco: quando questo parametro scende al di sotto dei 50, la scena apparirà gradualmente più grigia; questo consente al giocatore di capire quando l'umore di Emil non è ottimale.

Nel diagramma a lato è posta la logica del sistema dei materiali, che rafforza la breve descrizione precedente.

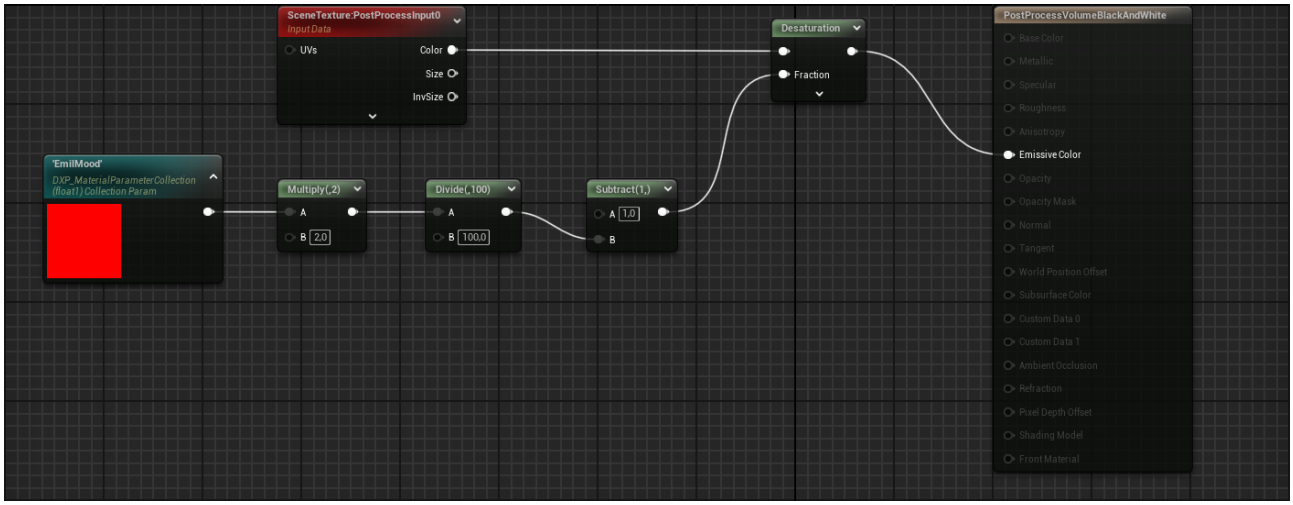
L'implementazione sarebbe notevolmente semplificata nel caso in cui UnrealEngine possedesse un nodo blueprint che consenta di ottenere un riferimento a tutti gli attori che compongono la scena di gioco: dopodiché, per ognuno di questi, basta ottenere un riferimento al relativo materiale, prenderne il colore di base e desaturarlo, in base all'umore di Emil.
Tuttavia, in UnrealEngine non esiste un metodo che consenta di ottenere un riferimento a tutti gli attori della scena: d'altronde, realizzare questo approccio sarebbe estremamente inefficiente in esperienze interattive 3D complesse. Dunque, per implementare la dinamica dei materiali discussa restano due alternative:

- Modificare ogni materiale utilizzato da ogni attore della scena (oneroso in termini di tempo)
- Manipolare il `PostProcessVolume`

È stato utilizzato il secondo metodo in seguito ad uno breve studio. Manipolare il `PostProcessVolume` richiede i seguenti passi per la realizzazione:

1. Creare un nuovo materiale (chiamato "`PostProcessVolumeBlackAndWhite`" nell'implementazione finale)
2. Aprire il file appena creato dal Content Browser e, nei Details del materiale, selezionare "`Post Process`" come Material Domain
3. Nell'Editor di UnrealEngine, selezionare il `PostProcessVolume` (utilizzato quello di default) e, tra le Rendering Features, aggiungere un nuovo elemento all'array dei Post Process Materials, selezionando il materiale creato al passo 1 ed impostando il valore associato a 1,0
4. Modificare la logica del materiale creato secondo le necessità. Resta comunque possibile, con un approccio più avanzato, evitare che singoli attori della scena siano oggetto di post processing

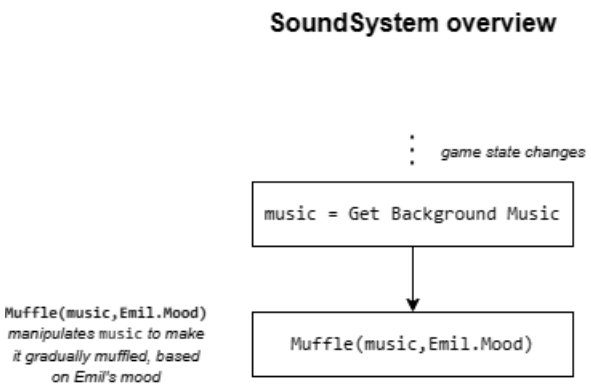
Nella seguente figura viene mostrata la logica del post processing grafico: in sintesi, il colore dell'immagine da riprodurre viene desaturato in base all'umore di Emil (tra 0 e 100), che viene moltiplicato per 2 (in quanto gli effetti devono essere riprodotti solo quando l'umore è < 50) e diviso per 100, onde normalizzarlo tra 0 e 1; quindi, viene sottratto a 1 il valore appena ottenuto, e il risultato della sottrazione viene usato come `Fraction` della desaturazione. Si immagini questo post processing come l'applicazione di un filtro B/N, in cui l'intensità del filtro equivale alla `Fraction`.
Di conseguenza: se l'umore di Emil è tra 50 e 100, la saturazione è al 100%; se l'umore di Emil è di 40, la saturazione è all'80%; se l'umore di Emil è di 30, la saturazione è al 60%; etc.
Nota. Il parametro "EmilMood" viene aggiornato nel metodo BP_Emil.UpdateMoodParameterValue, che viene invocato ogniqualvolta l'umore di Emil viene modificato nel corso del gioco.



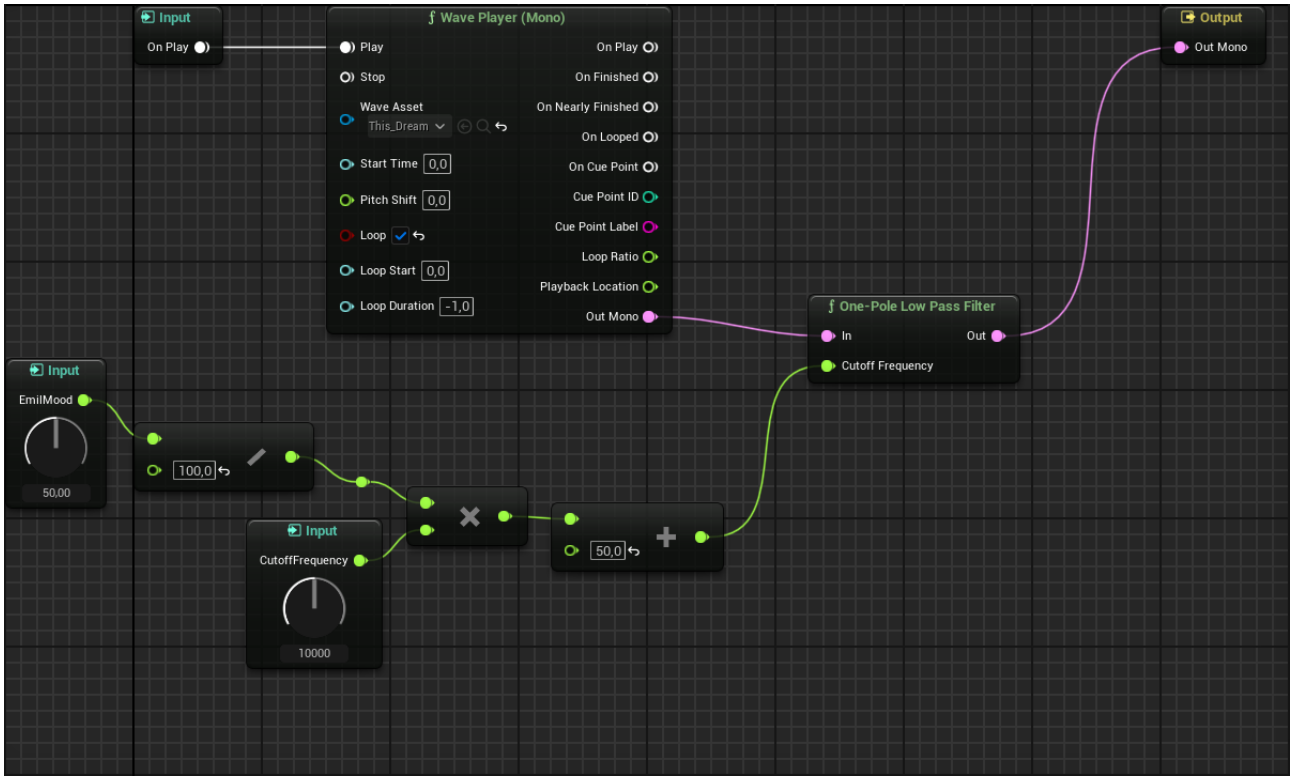
4.4 Sound system

Come per il sistema dei materiali, anche il sistema del suono è parte integrante dell'esperienza, e anch'esso cambia al variare dello stato interno del gioco. In particolare, l'umore di Emil influenza la colonna sonora del videogioco: man

mano che questo parametro scende, il suono risulterà gradualmente più ovattato, fino ad esserlo completamente quando l'umore raggiunge lo 0; anche questo sistema consente al giocatore di capire quando l'umore di Emil non è ottimale.



Lo pseudocodice `Muffle(music,Emil.Mood)` usa un MetaSoundSource per gestire la musica di sottofondo, avviandola semplicemente con un Wave Player (Mono): il risultato del player viene dato in input ad un filtro passa basso che, in base alla frequenza di taglio – calcolata secondo la formula $Cutoff\ Frequency = EmilMood * 100 + 50$ –, restituisce un suono più o meno ovattato. L'orecchio umano inizia ad avvertire una sostanziale differenza tra un suono ovattato e non quando la frequenza di taglio è al di sotto dei 500, in termini di umore di Emil quando quest'ultimo è ≤ 5. Sono stati scelti questi valori in modo da far capire al giocatore quando l'umore è estremamente basso, cioè prossimo allo 0. Diminuendo il tetto della frequenza di taglio al di sotto dei 10000 e regolando la percentuale di conseguenza, la qualità della musica sarebbe diminuita notevolmente.



Note per l'installazione

Verranno consegnati, allegati a questo documento, due archivi:

- `ArticyDraft3_Deep-X-Pression.zip` contiene la progettazione di Deep-X-Pression in ArticyDraft3
- `UnrealEngine_DeepXPression.zip` contiene il progetto UnrealEngine di Deep-X-Pression

Per il primo, è sufficiente estrarre la cartella dall'archivio nel percorso desiderato e, dalla finestra di benvenuto in seguito all'apertura di ArticyDraft3, utilizzare la funzione Apri da file e navigare fino a tale percorso, aprendo la cartella del progetto e selezionando il file `ProjectInfo.aph`.

Per il secondo, è già integrato nella cartella Plugins del progetto il plugin DialogueSystem utilizzato per il sistema di dialoghi. Pertanto, è sufficiente anche in questo caso estrarre la cartella dall'archivio nel percorso desiderato e, dal Project Browser di Unreal Engine, utilizzare la funzione Browse e navigare fino a tale percorso, aprendo la cartella del progetto e selezionando il file `DeepXPression.uproject`. Alla selezione comparirà una finestra in cui verrà chiesto di eseguire il rebuild di DeepXPression, DlgSystem e DlgSystemEditor, per cui è necessario installare Visual Studio (preferibilmente da UnrealEngine, che seleziona automaticamente le componenti necessarie) e .NET: se queste componenti sono installate, selezionare Sì e, in background, verrà avviata la fase di building del progetto, al termine della quale verrà aperto l'Unreal Editor con il progetto pronto all'uso.

Dall'Unreal Editor, utilizzare il pulsante Play per avviare la simulazione del videogioco. È preferibile avviarlo in modalità fullscreen premendo il pulsante F11 dopo aver selezionato la scena. Unica raccomandazione: all'avvio, il videogioco crea uno slot di salvataggio che viene sovrascritto ogniqualvolta l'umore di Emil viene aggiornato, conservando sia l'umore sia i rapporti con gli altri personaggi; premendo ESC per uscire dalla simulazione, questo slot viene mantenuto, dunque anche ai successivi Play. Per eliminare lo slot di salvataggio, premere il tasto Right Shift prima di premere ESC per uscire dalla simulazione, oppure selezionare Termina partita interagendo con la Exit Door.

Discorso presentazione

[~50s] Slide 2: Indice dei contenuti. Ho organizzato la relazione in quattro capitoli: i primi tre costituiscono la fase di progettazione, mentre la fase di implementazione è descritta nel Capitolo 4. Il primo capitolo è stato utile a mettere per iscritto l'idea del prodotto finale. Il secondo capitolo descrive l'ambientazione del videogioco, che include la location, i personaggi, gli oggetti di gioco e le loro variabili. Il terzo descrive la storia del videogioco, lo storytelling e le loro caratteristiche. Il capitolo conclusivo è stato scritto come una sorta di diario di sviluppo, ovvero man mano che venivano implementati i diversi sottosistemi.

[~50s] Slide 4: Il videogioco come mezzo di comunicazione. L'esperienza interattiva realizzata intende trasmettere al giocatore le sensazioni che prova una persona affetta da disturbi depressivi, che possono manifestarsi – tra gli altri – con sintomi cognitivi, affettivi, comportamentali, motivazionali e fisici. Per comunicare questo messaggio è stato deciso in fase di progettazione ciò che andava "mostrato, non detto" e ciò che andava "giocato, non mostrato" nel videogioco:

- Con il primo è stato scelto di rendere *visibili* dei cambiamenti nella scena di gioco, nei suoni e nei comportamenti e motivazioni del personaggio principale in base al suo umore
- Per il secondo principio, l'interazione con l'ambiente circostante e con gli altri personaggi è il fulcro dell'intera esperienza. Ed è così che viene raccontata la storia, senza mai dimenticare l'obiettivo del videogioco: comunicare il messaggio.

[~25s] Slide 5: Lo stato interno del gioco. L'intero gameplay è basato su un parametro, l'umore del personaggio, che influenza le azioni che può eseguire il giocatore. Inoltre, l'umore viene utilizzato nel sistema dei materiali e dei suoni dell'implementazione finale per riprodurre tali situazioni, di cui parlerò nelle ultime slide. Oltre ad interagire con gli oggetti presenti nella scena, il giocatore può prender parte a dei dialoghi con gli altri personaggi, un altro pilastro fondamentale del videogioco.

[~20s] Slide 6 + 7: La struttura del videogioco + Ufficio. Questa è una sintesi della struttura del videogioco, un simulatore di vita sviluppato in Unreal Engine utilizzando come base il 3rd Person Template. Le vicende di Emil, Harry, Jennifer, Kevin e Laura si svolgono in un ufficio lavorativo, di cui Emil, il personaggio principale, è il responsabile.

[~90s] Slide 9: Oggetti interagibili. Ho volutamente omesso dettagli superflui sui personaggi del gioco per dar più spazio agli oggetti interagibili.

- Gli orologi consentono di avanzare di un'ora, ma riducono di poco l'umore di Emil.
- L'area con divano e TV, utilizzabili durante l'ora di spacco, consentono ad Emil di recuperare energie fino alle 13:00 da solo, oppure insieme ai colleghi nel caso i loro rapporti siano buoni abbastanza.
- I distributori, utilizzabili una volta al giorno, consentono ad Emil di concentrarsi: egli può aiutare i colleghi solo quando è concentrato, ma l'utilizzo dei distributori riduce il suo umore.
- La scrivania, che migliora di ora in ora l'umore di Emil e fa trascorrere il tempo più rapidamente, consente di completare i lavori presi in carico dagli altri colleghi.
- La porta di uscita dell'ufficio può essere utilizzata in qualsiasi ora della giornata per terminare il gioco o per passare al giorno successivo; ma farlo prima che l'ottava ora lavorativa sia terminata ha dei risvolti negativi sul gruppo di lavoro e su Emil.
- I water consentono ad Emil di ripristinare l'ultimo dialogo avuto con i suoi colleghi, qualunque sia stato l'esito, e di migliorarne l'umore nel caso sia estremamente basso.
- Le finestre consentono ad Emil di avere suggerimenti sulle decisioni importanti da prendere nei dialoghi.

[~**20s**] **Slide 11: Struttura della storia.** Passiamo ora alla narrativa del videogioco. Deep-X-Pression simula il classico giorno lavorativo di Emil: in qualsiasi momento, il giocatore può interagire con la porta per scegliere di passare al giorno successivo oppure per terminare la partita. In sostanza, l'esperienza è una reiterazione del giorno lavorativo.

[~35s] Slide 12 + 13: Il giorno lavorativo + L'esplorazione. La giornata (dalle 8 alle 17, con un'ora di spacco tra le 12 e le 13) inizia con un saluto ai colleghi: se questi rispondono, il rapporto con essi è buono, altrimenti non lo è; dopo questa breve interazione inizia la fase di esplorazione. Dopo aver esaurito un dialogo con un personaggio, questi tornano disponibili solo dopo aver avanzato di un'ora presso l'orologio o la scrivania. Gli oggetti sono sempre interagibili, con qualche eccezione. Al centro della figura è posta l'interazione con la porta d'uscita, che è l'unico modo per passare al giorno successivo o per terminare la partita.

[~70s] Slide 14: Caratteristiche dello storytelling. Alla luce di quanto detto, la storia di Deep-X-Pression non ha uno **story path** definito. Sulla base del messaggio da veicolare, le meccaniche della finestra e del water **motivano estrinsecamente** il giocatore a prendere le decisioni buone, ma al contempo alimentano la **curiosità** nel percorrere la scelta cattiva. Non è mai esistito, di conseguenza, uno story path critico mantenuto nella fase di progettazione. Per questo motivo è arduo definire un **arco narrativo** per Deep-X-Pression; lo si può tuttavia immaginare come una combinazione tra gli archi narrativi *Man in a hole* e *Icarus*, siccome il giocatore medio tende a scegliere il percorso estremamente buono o il percorso estremamente cattivo. Infine:

- La **crescita** di Emil viene evidenziata dalla sua capacità di socializzare con gli altri personaggi, aspetto che lo fa sentire di buonumore e che gli fa osservare il mondo che lo circonda in modo più vivo
- Lo **sviluppo** di Emil è caratterizzato dalla scoperta di fatti legati al suo passato che lo hanno reso così vulnerabile

I dialoghi consentono anche di rivelare circostanze sugli altri personaggi che ne fanno emergere il carattere.

[~150s] Slide 16: Video gameplay del gioco. Nel documento di progettazione ho inserito questa sorta di diario di sviluppo, spiegando le decisioni di implementazione impossibili da riprodurre in Articy:Draft ed eventuali deviazioni dalla progettazione originale, che non sono state – per fortuna – rilevanti.

[~10s] Slide 17: Impatto dell'umore sul gioco. Ho inserito questa slide soltanto come riepilogo delle meccaniche relative ai principi "Show, don't tell" e "Play, don't show". Non c'è nulla da aggiungere, per cui andrei avanti.

[~55s] Slide 18: Meccaniche aggiuntive.

- Dal video si nota l'esistenza di un ciclo temporale: il sole cambia la sua posizione a seconda del tempo. Alle ore 8:00, cioè all'inizio della giornata lavorativa, il sole è perfettamente in alto al centro della scena (come se qui fossero le 12:00), mentre alle ore 17:00 il sole tramonta. Questo consente al giocatore di capire quando può lasciare l'ufficio senza penalità anche senza guardare l'orologio.
- È stato studiato e utilizzato un plugin esterno per il sistema di dialoghi: DialogueSystem di NotYet, utilizzato anche in diversi videogiochi indie.
- Interagendo con la finestra, si può anche leggere una breve descrizione dei personaggi.
- Interagendo con il water, si possono consultare in qualsiasi momento i parametri fondamentali del gioco, per far capire al giocatore con più facilità l'importanza dell'umore e quali sono le azioni che lo migliorano/peggiorano.
- Nella fase di summarising viene riepilogato lo score a fine partita.

[~60s] Slide 19: Material system. Il sistema dei materiali è parte integrante delle meccaniche di gioco: la componente visiva cambia al variare dell'umore di Emil. Quando questo parametro scende al di sotto dei 50, la scena apparirà gradualmente più grigia: questo consente al giocatore di capire quando l'umore di Emil non è ottimale. Per implementare questa dinamica sono state fatte due valutazioni:

- Modificare ogni materiale utilizzato da ogni attore della scena
- Manipolare il `PostProcessVolume`

È stato estremamente più semplice, più efficiente e meno oneroso in termini di tempo utilizzare il secondo metodo. Nella figura viene mostrata la logica del post processing grafico: se l'umore di Emil è tra 50 e 100, la saturazione è al 100% (in questo range, il parametro EmilMood dei materiali viene mantenuto uguale a 50); se invece è minore di 50, l'immagine risulterà gradualmente desaturata, utilizzando come frazione proprio la differenza tra il "100%" e l'umore di Emil.

[~40s] Slide 20: Sound system. Come per il sistema dei materiali, anche il sistema del suono è parte integrante dell'esperienza, e anch'esso cambia al variare dell'umore di Emil, che influenza la colonna sonora del videogioco: man mano che questo parametro scende, il suono risulta gradualmente più ovattato. Anche questo sistema consente al giocatore di capire quando l'umore di Emil non è ottimale. Questo è stato realizzato mediante MetaSoundSource per gestire la musica di sottofondo, avviandola semplicemente con un Wave Player (Mono): l'output del player viene dato in input ad un filtro passa basso che, con i parametri utilizzati, restituisce un suono più o meno ovattato.

TOTALE: 675s